2011

Experimental Simulation of Rapid Pressure Swing Adsorption for Medical Oxygen Concentrator and Numerical Simulation of the Critical Desorption-by-Purge Step

Siew Wah Chai
Lehigh University

Experimental Simulation of Rapid Pressure Swing Adsorption for Medical Oxygen Concentrator and Numerical Simulation of the Critical Desorption-by-Purge Step

by

Siew Wah Chai

Presented to the Graduate and Research Committee

of Lehigh University

in Candidacy for the Degree of

Doctor of Philosophy

in

Chemical Engineering

Lehigh University

September 2011

Copyright

Certificate of Approval

Approved and recommended for acceptance as a dissertation in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Dr. Mayuresh Kothare (Dissertation Advisor)

Accepted Date

Committee Members

Dr. Mayuresh V. Kothare (Committee Chairman) Date
Department of Chemical Engineering

Dr. Shivaji Sircar (Adjunct Professor) Date
Department of Chemical Engineering

Dr. William E. Schiesser (Professor Emeritus) Date
Department of Chemical Engineering

Dr. James T. Hsu Date
Department of Chemical Engineering

Dr. Hugo S. Caram Date
Department of Chemical Engineering

Acknowledgements

I would like to extend my sincere gratitude to:

Dr. Mayuresh Kothare, for serving as my advisor and putting absolute trust in my work progress. He has provided much freedom, support, and positive encouragement in my current research work and future career.

Dr. Shivaji Sircar, for pouring so much patience and kindness on me. His deep passion and seriousness about doing good research has inspired me and solidified my interest in adsorption separation. I am very privileged to have him as my teacher.

Dr. William Schiesser, Dr. Hugo Caram, Dr. James Hsu, for serving as my Ph.D. committee members, and Dr. Schiesser, for his generous guidance in numerical method of lines. I am grateful that each of them has also instilled in me deeper interest and appreciation of the research work.

Dr. Hoongsun Im, for mass spectrometer support; Dr. Michael Beaver, for project support; John Caffrey and Paul Bader, for practical support; Chemical Engineering Department of Lehigh University and Invacare Inc., for financial support.

My church: Pastor and Mrs. Ho, Daisy and Kwai, XiaoMing and HaiYan, Tony and Helen. They are like my family and have been upholding me in prayers.

My family: my father and mother, for putting absolute trust in me and giving me so much freedom and inexpressible love. My sisters, brother, and my aunt Mae Yip, she has been my wise counselor and my prayer warrior all the time.

Finally, I dedicate my upmost gratitude, honor, glory, and praise to my Lord Jesus Christ. I can do nothing good apart from Him and with His purpose in my life.

Table of Contents

List of Tables

List of Figures

Figure 7.10: Fraction of total N_2 desorbed vs amount of O_2 purge gas used. L/D = 1.39, d_p= 1500 µm, k_{N_2}= 5.3 s^{-1}, k_{O_2}= 5.1 s^{-1}, D_L ~ 24 cm^2/s, D_g ~ 0.22 cm^2/s, Nu ~ 20.4, Re ~ 144, ha ~ 0.21 $cal/cm^3/s/K$, $Q_{O_2,feed}$ = 5.35, 1.07 $mmoles/cm^2/s$, respectively, for t_{purge} = 2, 10 s. Legends: Table 7.6.

Abstract

As efforts are sought to achieve an overall compact and lightweight medical oxygen concentrator (MOC), rapid pressure swing adsorption (RPSA) plays an important role to lower the bed size factor (BSF). A mini experimental system supported by switching control and data acquisition was built to simulate a Skarstrom-like RPSA cycle. It was experimentally demonstrated that BSF cannot be indefinitely reduced by lowering total cycle time (t_c) and adsorbent particle size (d_p). However, an optimum process performance of BSF = 25 – 50 lbs/TPD$_c$ with an O_2 recovery (R) = 25 – 35% for production of ~ 90% O_2 could be achieved using a dry, CO_2-free air feed at adsorption pressure (P_A) of 3 – 4 atm, d_p of ~ 350 µm, and t_c of 3 – 5 seconds.

A rigorous mathematical model using finite volume method with Superbee flux limiter was developed and solved by numerical method of line (MOL) and Matlab's ode solver to simulate the critical O_2 back-purge step of a RPSA cycle. Finite adsorption kinetics, column pressure drop, and non-isothermality are the primary impediments to efficient purge in RPSA MOC of this study, while the effects of gas-phase mass and thermal axial dispersions and gas-solid heat transfer resistance are negligible. Very short purge time and small d_p in pancake adsorber is not practical. d_p of 300 – 500 µm yields optimum purge efficiency for bed size reduction of MOC.

This dissertation is written in three major parts with Chapter 1 the introduction, Chapters 2 to 4 the experimental work, and Chapters 5 to 7 the model simulation work; major research findings are reported in Chapters 4 and 7.

Chapter 1

Introduction, Motivations, and Research Goals

The general concept of rapid pressure swing adsorption (RPSA) is introduced for application in portable and small size medical oxygen concentrator (MOC) as envisaged by its current market strength and its growing potential. Though widely researched, most experimental and model simulation works of RPSA MOC failed to identify the constraints of RPSA technology in MOC application, particularly, adsorber size cannot be reduced indefinitely and a relaxed mathematical model cannot give a real picture of the separation performance. Adsorbent bed size reduction is the focus of this dissertation, which is investigated by experimentally finding the limits of operating and process conditions, as well as model simulation of desorption-by-purge step that critically governs the overall separation performance.

1.1 Medical Oxygen Concentrator via Pressure Swing Adsorption Technology

Pressure swing adsorption (PSA) is a widely used gas separation method for medium scale production of bulk gases and for trace removal like removing carbon dioxide from flue streams in chemical plants. It is a relatively safe, simple and cost effective operation compared to cryogenic separation. The application of PSA has extended from commercial plants at tonnage scale [~ 1 to 150 tons per day (TPD)] to onsite generation and even to mobile usage such as portable medical oxygen concentrator for respiratory therapy, with production rate of ≤ 5 liters per minute

(LPM) (~ 0.01 TPD). Due to the market demand of increasingly small size and lightweight MOC so as to increase mobility and to improve life quality, reduction of the adsorber size of a PSA-based MOC is an ongoing research and development activity.

1.1.1 Medical Oxygen Concentrator

Oxygen concentrator has the biggest market in medical use by patients that suffer chronic obstructive pulmonary disease (COPD). COPD is the 4th leading cause of death in America, claiming 122,283 lives in year 2003. Statistics shows that 11.4 million U.S. adults were estimated to have COPD, and close to 24 million U.S. adults have evidence of impaired lung function in year 2004. Though bronchodilator medications are helpful in controlling COPD, scientific studies demonstrate that appropriate use of supplemental oxygen (O_2) prolongs the lives of these patients (American Lung Association 2006).

Table 1.1 and Figure 1.1 provide the general information of home use and portable medical oxygen concentrators that produce 87 – 96% O_2 and are available in the current market. A home use O_2 concentrator that delivers 5 LPM of O_2 product is bulky and weights around 50 lbs. A number of manufacturers have introduced portable O_2 concentrators that weight around 10 lbs with production rate ~ 1 LPM, these MOCs pulse dose O_2 only when the patient is inhaling. Note that the price range for home use MOC is below $1000 but the sale price for a portable MOC can go as high as $5000. A search of U.S. Patents for portable MOCs was carried out and the general information is given by Table A.1 in Appendices. It shows that making compact O_2 concentrator dated back as early as 1970s and has become more prevalent in recent years.

In addition to portable MOC, other applications exploiting the reduced bed size of O_2 concentrator include distributed O_2 supply to replace the centralized source that depends on gas cylinders in hospitals (Dobson 1999; East Coast Radio Newswatch 2008), inside armored vehicles and military aircraft (Teague and Edgar 1999), onsite O_2 generation for aqua culture, O_2 batteries in portable electronic devices (Jiang and Abraham 2000), O_2 booster for improving health (Patterson 2008).

Table 1.1: Commercial home use and portable medical oxygen concentrators.

Device name	**Delivery mode**	**Weight (lbs)**	**Dimensions (Inches)**	**Noise (dBA)**	**Power consumption Battery lasting time**	**Maximum oxygen capacity per min (mL)**	**FAA-approved for in-flight use**	**Price (US$)**
Invacare Perfecto 2	Home use - continuous flow	43	23″ x 13″ x 11.5″	< 43	280 W (at 3 LPM continuous flow)	5000	NA	677 – 852.6
Invacare SOLO 2	Continuous /pulse dose	19.9	16.5″ x 11″ x 8″	< 39 (at all pulse dose settings) 42 (at 3 LPM continuous flow)	NA 4.5 hours	3000	Yes	3238 – 4499
DeVilbiss iGO	Continuous /pulse dose	19	15″ x 11″ x 8″	40 (at pulse dose setting 3) 47.8 (at 3 LPM continuous flow)	45 W (at pulse dose setting 1) 130 W (at 3 LPM continuous flow) 5.4 hours	3000	Yes	2800 – 4095
SeQual Eclipse 3	Continuous /pulse dose	18.4	19.3″ x 12.3″ x 7.1″	40 (at pulse dose setting 3) 48 (at 3 LPM continuous flow)	45 W (at pulse dose setting 1) 145 W (at 3 LPM continuous flow) 5.2 hours	3000	Yes	2639 – 3995
Respironics EverGo	Pulse dose	10	12″ x 8.5″ x 6″	43 (at setting 2)	70 W 8 hours	1050	Yes	4095

Device name	Delivery mode	Weight (lbs)	Dimensions (Inches)	Noise (dBA)	Power consumption Battery lasting time	Maximum oxygen capacity per min (mL)	FAA-approved for in-flight use	Price (US$)
Evo Central Air by Delphi	Pulse dose	9.8	11.6” x 7.4” x 4.6”	NA	NA 4 hours	925	Yes	3995 – 4495
Invacare XPO2	Pulse dose	7.3	10” x 7” x 4”	< 45 (at setting 2)	36 W 5 hours	900	Yes	3995
Inogen One G2	Pulse dose	7.25	10.7” x 9.5” x 3.9”	38 (at setting 2)	40 W 4 hours	900	Yes	3295 – 3995
FreeStyle by AirSep	Pulse dose	4.4	8.6” x 6.1” x 3.6”	41 (at setting 2)	28 W 2.5 hours	480	Yes	2400 – 3150

Note: Information was obtained from sales websites in December 2010

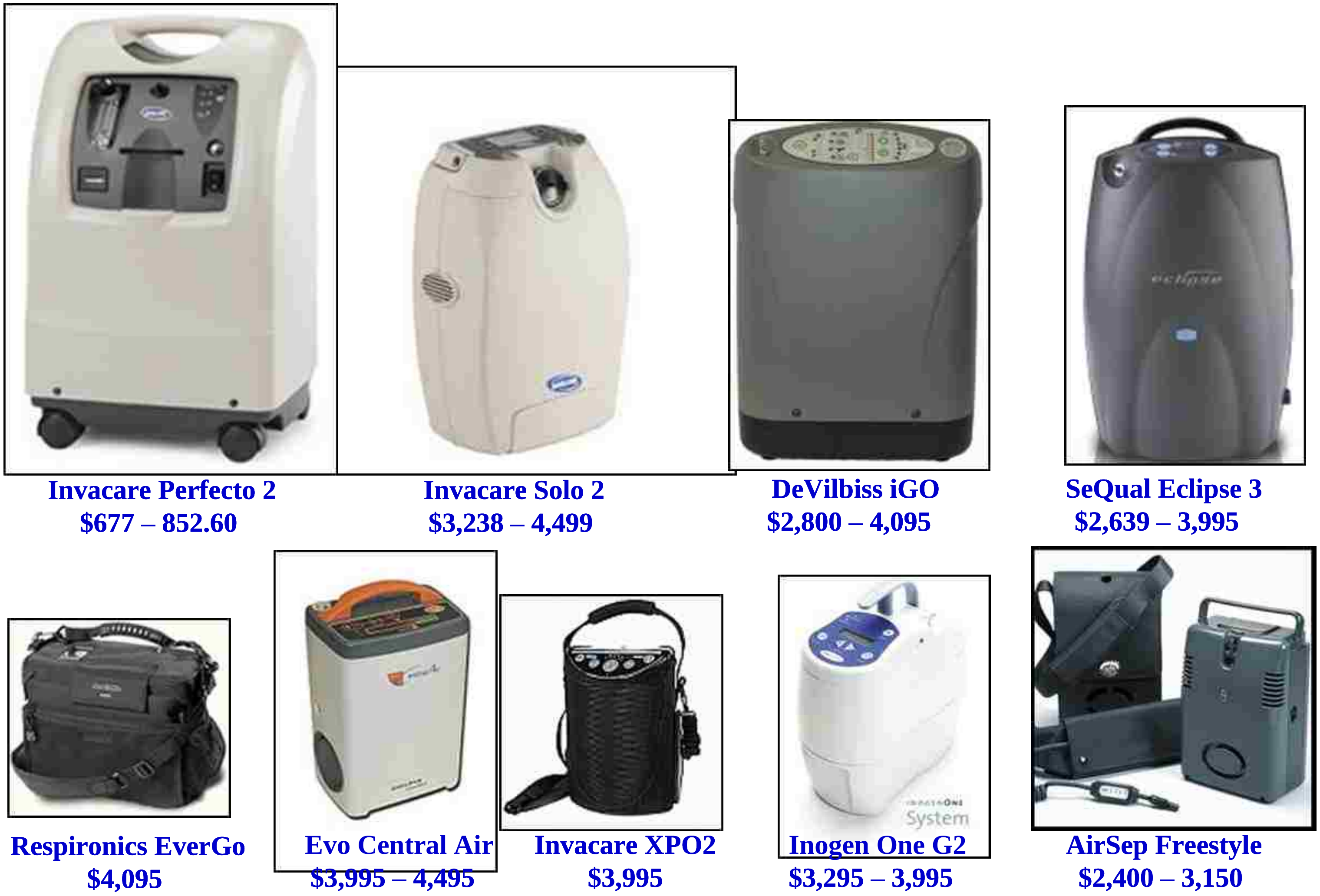


Figure 1.1: Price range of commercial home use and portable medical oxygen concentrators.

1.1.2 Pressure Swing Adsorption Technology

Many pressure swing adsorption processes have been patented and commercialized for production of ~ 90 – 95% O_2 enriched gas from ambient air by employing a zeolitic adsorbent (Sircar 1988; Sircar et al. 1999; Sircar and Myers 2003). Figure 1.2 shows the result of a patent search under the topic “air separation by adsorption” which plots a year by year tally of number of U.S. Patents on the subject during the period of 1980 to 2005. An overwhelming number of 452 patents were issued. The field is very crowded but still active (~ 7 patents issued per year).

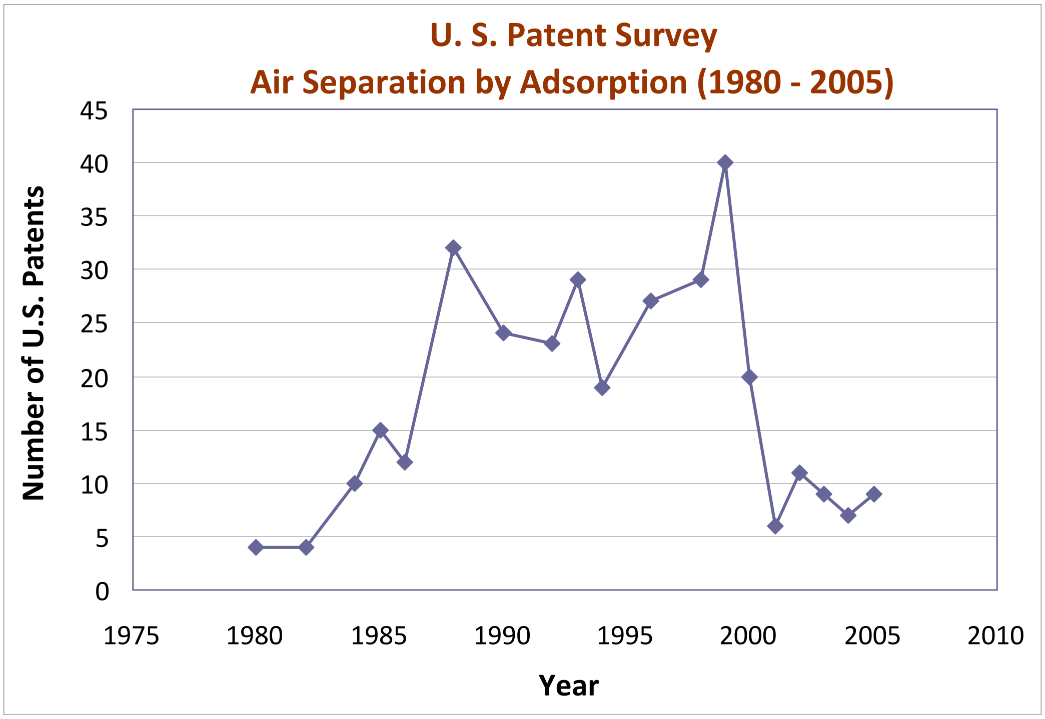


Figure 1.2: Result of patent search 1980 – 2005.

In principle, a generic PSA process for air separation cyclically carries out an adsorption step at a relatively higher pressure, where nitrogen (N_2) is selectively (thermodynamic) adsorbed from a dry and CO_2-free air stream on the zeolite, thereby, producing an O_2 enriched product gas, followed by N_2 desorption at relatively lower pressures where the zeolite is regenerated for reuse. Various complementary steps are often included in the process cycle to improve the product specifications and the process performances.

Three subclasses of the generic PSA technology based on the pressure levels of the adsorption step (P_A) and the ultimate desorption step (P_D) are common in the literature. They are pressure swing adsorption (PSA) where P_A is super-atmospheric and P_D is ambient, vacuum swing adsorption (VSA) where P_A is near ambient and P_D is sub-atmospheric, and pressure-vacuum swing adsorption (PVSA), which is a combination of the two. A VSA process employing a zeolite has also been commercialized where a N_2-rich byproduct gas (99+% N_2) is simultaneously produced (Sircar and Zondlo 2003).

1.1.3 Overall Performance for Adsorbent Bed Size Reduction

A common practice to decrease the bed size factor [BSF, lbs of adsorbent in the PSA unit per ton of contained O_2 per day production rate (TPD_c)] is by lowering the total cycle time (t_c, seconds) of the PSA process. Adsorbent columns packed with very small adsorbent particles are used to enhance the adsorption kinetics for rapid PSA (RPSA) cycle operation. In order to depict the relation of t_c to BSF and O_2 recovery, an overall performance analysis is given below.

Figure 1.3 is a block diagram depicting the overall performance of a generic PSA air separation system producing 92% O_2-enriched product gas. The total amount of zeolite adsorbent in the system is w (lbs) and the total cycle time of the PSA process is t_c (seconds). The variables F, P and W are, respectively, the amounts (lb moles) of air feed, O_2-enriched product gas, and N_2-rich waste gas from the system per cycle of operation.

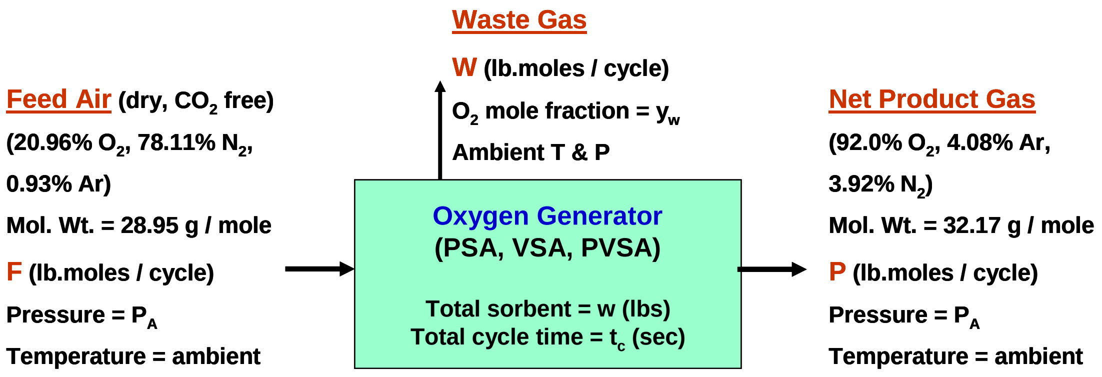


Figure 1.3: Block diagram of a generic PSA air separation system.

According to Figure 1.3, the net tonnage rate of production of contained O_2 per day (TPD_c) is equal to 1.272 x10^3 P/t_c, and

The bed size factor (BSF, lbs/TPD_c) is = 7.862 x 10^{-4} $w.t_c/P$ (1.1)

The O_2 recovery from air feed (R) = 0.92 P/0.2096 F (1.2)

The above-described unit of BSF is an industry standard.

For a given PSA process cycle (process steps and sequence, operating conditions, vessel design, system void, individual and total cycle times) and a given sorbent (sorptive and physical properties), one can define a specific O_2 productivity (N, lb moles/lb of zeolite/cycle = P/w) by the process. Consequently,

BSF = 7.862 x 10^{-4} t_c/N (1.3)

It can also be shown by rearranging equations (1.1) – (1.3) that,

Total amount of adsorbent in PSA system, w = BSF x TPD_c (1.4)

Air feed rate to PSA system (lb moles/sec) = F/t_c = 3.451 x 10^{-3} TPD_c/R (1.5)

Equations (1.4) and (1.5) show that BSF and R are two important performance variables for a generic PSA process for air separation because for a given plant size (TPD_c), the total amount of adsorbent inventory in the system is directly proportional to BSF, while the feed air flow rate for the process is inversely proportional to R.

Obviously lower BSF and higher R are preferred characteristics of a generic PSA O_2 concentrator. Equation (1.3) shows that BSF can be decreased by lowering t_c provided that N is not a function (or a weak function) of t_c. In other words, rapid cycles permit the adsorbent to be used more frequently and, therefore, increase the O_2 productivity rate provided that the performance is not affected by fast cycling of the process. This behavior is expected to be valid at relatively larger cycle times where the time scales of local adsorption kinetics and kinetics of heat transfer between the adsorbent and the gas phase, which are normally fast, are much smaller than the individual step times and the overall process cycle time. On the other hand, these kinetic impediments can be detrimental to the process performance if the characteristic time constants of these processes are comparable to the PSA process cycle time as it can be in the case of a very rapid cycle.

1.1.4 Bed Size Factor of Tonnage Scale Commercial Plants

Reduction of total cycle time to decrease the BSF of a PSA O_2 concentrator, however, has been implemented in design of many tonnage scale (~ 1 to 150 TPD_c) commercial plants. Table 1.2 provides several examples of progressive reduction of total cycle time and BSF over the years (Batta 1972; Collins 1976; Neill et al. 1998; Leavitt 1995; Sircar et al. 2000).

The PSA process developed by Batta (Table 1.2) is one of the earlier patented and commercialized processes for production of ~ 90% O_2 from air. The PVSA processes described in the second and third columns are more recent developments. The improvement in BSF due to faster cycle times is evident. The evolution of the preferred zeolite for the air separation application is also notable in the Table. These processes typically employ adsorbent particle diameters of 1 – 2 mm for low pressure drops during the process steps.

Table 1.2: Performance of commercial tonnage scale O_2 concentrators.

	U. S. Patents 3,636,679 (1972) 3,973,931 (1976) 5,711,787 (1998)	**U. S. Patent 5,415,683 (1995)**	**U. S. Patent 6,146,447 (2000)**
Authors	Batta Collins Neill, Leavitt, Figueiredo	Leavitt	Sircar, Naheiri, Fischer
Company	Union Carbide Corp., Praxair Technology	Praxair Technology	Air Products
Process Mode	PSA	PVSA	PVSA
Adsorbent	5A Zeolite	LiX Zeolite	Li-ZnX Zeolite
P_A : P_D (atm)	3 – 6.3 : 1	1.43 : 0.34	~ 1.3 : 0.45
Parallel adsorbers	3, 4	2	1
O_2 product purity	90 – 93%	90%	90%
O_2 recovery	33 – 46%	53%	40 – 75%
Total cycle time (seconds)	240	70	~ 45
BSF (lbs/TPD_c O_2)	~ 5600 – 3000	830	777

Tonnage scale production of ~ 90% O_2 from air by a PVSA process appears to have become the 'state of the art technology'. Some of the applications of such process include waste water treatment, ozone generation, bio-reactors, glass furnaces, etc. The minimum individual step and total cycle times of these PSA processes may be limited by hydrodynamics of large volume gas flow through the packed column adsorbers and gas headers, valve switching time, vacuum pump size, etc.

1.1.5 Bed Size Factor of Experimental Simulated Oxygen Concentrator

A drastic reduction of total cycle time (t_c ~ 10 seconds or less) in order to decrease BSF for PSA O_2 concentrators has been achieved by RPSA. Table 1.3 provides a few examples of patented RPSA processes (Sircar 1991; Kulish and Swank 1998; Ackley and Zhong 2003; Jagger et al. 2006; McCombs et al. 2006; Chai et al. 2010) which have been experimentally demonstrated. These RPSA processes employ very small particles of the adsorbent ($d_p \leq 0.5$ mm) to enhance gas-solid adsorption and heat transfer kinetics.

The first entry in Table 1.3 describes the performance of a RPSA process for production of 25 – 60% O_2-enriched air from ambient air (Sircar and Hanley 1995; Sircar 1991). The process uses a novel design of two shallow adsorber beds packed with NaX zeolite, each sequentially undergoing two steps: (a) simultaneous pressurization with compressed air and production of O_2 enriched product gas, and (b) simultaneous depressurization and back purging with a part of the O_2 enriched gas. The reported data is from a pilot test rig using dry air feed. The process was designed for tonnage scale production of medium purity O_2 for enhanced combustion application (Sircar 1996).

The other RPSA O_2 concentrators described in Table 1.3 were developed for providing medical O_2 to patients suffering from Chronic Obstructive Pulmonary Disease (COPD) caused by emphysema, chronic bronchitis, pulmonary fibrosis, etc. Note that BSF of these experimental demonstrated processes were reduced drastically compared to those of tonnage commercial plants in Table 1.2.

Table 1.3: Experimental performance of various RPSA O2 concentrators.

Process	**Cycle Mode & time (sec)**	**Zeolite d_p (mm)**	**$P_A : P_D$ (atm)**	**Product O_2**		**BSF (lbs/TPD_c)**
				Purity (mole %)	**Recovery (%)**	
Sircar **U.S. Patent 5,071,449 (1991)**	PSA 8 – 12	NaX 0.5	3 : 1	40	46	263
Kulish & Swank **U.S. Patent 5,827,358 (1998)**	PSA 6	Zeolite (not disclosed)	3 : 1	90 – 96	not disclosed	~ 110 – 125
Ackley & Zhong **U.S. Patent 6,551,384 (2003)**	PSA 9 PVSA 4	LiX C.D. 0.5	3 : 1 (PSA) 1.5 : 0.5 (PVSA)	85 – 95	50 (PSA) 60 (PVSA)	90 (PSA) 50 (PVSA)
Jagger, Van Brunt, Kivisto, Lonnes **U.S. Patent 7,121,276 (2006)**	VSA 0.54 – 5.6	Low Silica LiX	1 : 0.3	85 – 95	60	~ 20 – 230
McCombs, Bosinski, Casey, Valvo **U.S. Patent Application 0117957 (2006)**	PSA 11	Zeolite (not disclosed)	1.9 – 2.7 : 1	~ 90	~ 28	~ 274
Chai, Sircar, Kothare **U.S. Patent Application 0300285 (2010)**	**PSA** **~ 3 – 5**	**Low Silica** **LiX** **~ 0.35**	**3 – 4 : 1**	**~ 90**	**~ 25 – 35**	**~ 25 – 50**

1.1.6 Bed Size Factor of Model Simulated Oxygen Concentrator

A large volume of work on model simulated performance of different RPSA processes using different zeolites for MOC application has also been published in recent years, Table 1.4 is a list of selected works on this topic (Kopaygorodsky et al. 2004; Santos et al. 2004; Santos et al. 2006; Zhong et al. 2010; Rama Rao et al. 2010). It describes adsorbent particle size, cycle time, and process performance. Different models for adsorption isotherms of N_2 and O_2 and different models for their sorption kinetics have been used in these simulations. All models include axial dispersion in the gas phase, but some of them ignore column pressure drop and non-isothermal operation of the PSA process.

It may be seen from Tables 1.3 and 1.4 that many of the RPSA cycles are indeed successful (experimentally demonstrated or model simulated) in satisfying the required performance of a MOC and the BSF is reduced substantially from those of the tonnage scale units (Table 1.2) when the cycle time is lowered by an order of magnitude or more. The total cycle times of the PSA processes listed in Table 1.4 lie between 3 – 18 seconds (except that of Rama Rao et al.) and the BSF of these processes decrease as the cycle time is reduced irrespective of the type of the process.

Table 1.4: Model simulation studies of various PSA O2 concentrators.

Authors	**Performance**			**Mass Transfer Kinetics**	**Heat Balance**	**Pressure Drop**	**Axial Dispersion**	**Adsorption Isotherms**	**Adsorbent**	**Total Cycle Time (sec)**
	O_2 product purity (%)	**Rec. (%)**	**BSF (lb/TPD_c)**							
Kopaygorodsky et al. 2004 **Process: RPSA** **P_A = 1.6 atm** **P_D = 1.0 atm**	85	56	14.2	No resistance	No	Yes	Yes	Linear	5A zeolite monolith d_p=2 µm	3
Santos et al. 2004 **Process: PSA** **P_A = 3.0 atm** **P_D = 1.0 atm**	94.5	22	528	LDF model	No	No	Yes	Langmuir-Freundlich	Oxysiv 5 (NaX zeolite) d_p=~600 µm	18
Santos et al. 2006 **Process: PSA** **P_A = 3.0 atm** **P_D = 1.0 atm**	94.5	32	455	LDF model	No	No	Yes	Langmuir-Freundlich	Oxysiv 5 (NaX zeolite) d_p=~600 µm	16
Zhong et al. 2010 **Process: RPVSA** **P_A = 1.5 atm** **P_D = 0.5 atm**	90	25 – 50	< 50	Solid phase diffusion model	Yes	Yes	Yes	Langmuir	LiX 2.0 C.D. zeolite $150 \le d_p \le 350$ µm	< 4
Rama Rao et al. 2010 **Process:** **Pulsed PSA** **P_A = 3.5 atm** **P_D = 1.0 atm**	~90	24	5.2	LDF model	No	Yes	Yes	Langmuir	Ag-Li-X zeolite d_p=20 µm	0.235

1.1.7 Non-Negligible Mass, Heat, Momentum Transfer Resistances

A simplified isothermal model of an idealized differential PSA concept using a single pellet of an adsorbent for removal of a single adsorbate from an inert gas, where the gas phase partial pressure of the adsorbate was periodically changed differentially, showed that the net rate of the adsorbate removal (moles/gm of adsorbent/time) by the adsorbent increased as the cycle time was decreased (Sircar and Hanley 1995). This was equivalent to decreasing BSF with decreasing cycle time. In absence of an adsorbate mass transfer resistance, the rate approached infinity when the cycle time approached zero (BSF approached zero). However the rate of adsorbate removal approached a finite value at the limit of zero cycle time when the adsorbate had a finite mass transfer coefficient (k) for sorption. The limiting rate was a strong function of k (Sircar and Hanley 1995).

A non-isothermal extension of the model, where the effect of a finite gas-solid heat transfer coefficient (h) on the performance of a rapid cycle was investigated, showed that the relative efficiency of adsorbate removal by the non-isothermal process relative to the isothermal performance decreased substantially at smaller cycle times (Sircar 2005).

Adsorbent particles of small diameter (d_p) are deliberately used in a RPSA cycle in order to decrease the above-mentioned kinetic resistances. The transfer coefficients k and h are typically proportional to d_p^{-2} and d_p^{-1}, respectively, in absence of axial dispersion. However, as discussed in the model simulation work by Zhong, Rankin and Ackley (2010), further reducing d_p to below ~ 1.0 mm causes the mass transfer resistance to shift from pore diffusion to axial dispersion, and the pressure drop to increase faster than the gain in mass transfer rate. The BSF can also increase and R can decrease due to increase in the quantity of product O_2 back purge needed for desorption of N_2 from a zeolite column with a large pressure drop (Sircar and Golden 1995).

The column pressure drop in a RPSA adsorber can be minimized by employing a shallow adsorber having a relatively low length (L) to diameter (D) ratio, but $L/D <$ 0.5 – 1.0 can cause poor gas distribution and channeling, and inadequate contact between gas and solid particles (Porter et al. 1993). The problem may be alleviated by adding elaborate gas distributors at two ends of the adsorber but that option may not be appropriate for small scale RPSA systems since it will increase the specific void gas volume of the system. A very small column length can also lower the separation efficiency by creating a relatively large ratio of adsorbate mass transfer zone length to column length (Zhong et al. 2010). Use of very small adsorbent particles ($d_p <$ 500 μm) can also cause agglomeration in packed beds which can impede gas-solid mass and heat transfer rates (Moulijn and Van Swaaij 1976).

1.1.8 Limitations in Adsorbent Bed Size Reduction

Due to the resistances mentioned above, Figure 1.4, therefore, is proposed as the realistic BSF vs t_c profile for a PSA O_2 concentrator. For a given particle size, the BSF decreases approximately linearly with decreasing cycle time, according to equation (1.3), when the process cycle time is large. However, this pattern is not continued indefinitely. The BSF levels off or even increases with decreasing t_c for very rapid cycles due to the impediments introduced by mass and heat transfer resistances and column pressure drop. The progressive increase of BSF at low cycle time is expected because the separation efficiency suffers when the PSA cycle time is faster than the characteristic time constants for the heat and mass transfer process.

Therefore, one of the main objectives of this dissertation research is to identify the practical limitations (cycle time, particle size, adsorption pressure, bed size) of a MOC producing ~ 90% O_2 and its optimum process performance using an experimental simulated RPSA process.

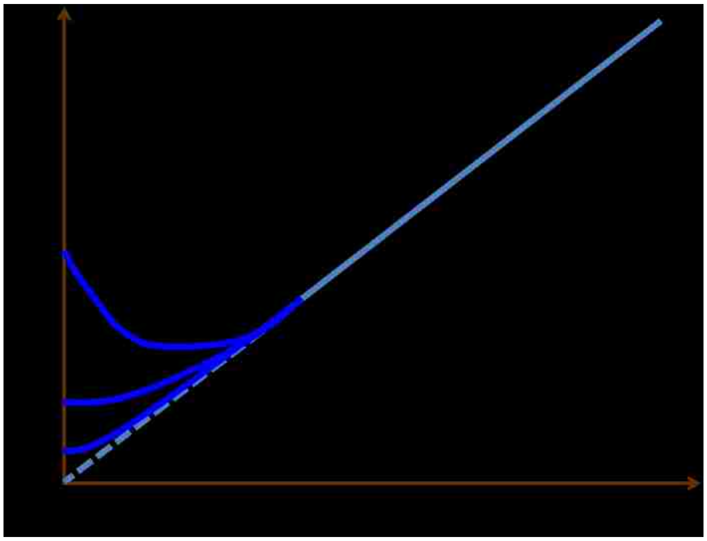


Figure 1.4: Schematic drawing of BSF vs cycle time for a RPSA process.

1.2 Desorption-by-Purge: The Critical System Performance Governing Step

Generally, adsorption step of PSA air separation is carried out at super-ambient pressure (3 – 4 atm). Hence a certain amount of pressure drop (say < 5 psi associated with small particle size and high mass flow rate) in the column during adsorption step may not be critical. However such pressure drop may be critical (inferior & inefficient purge clean up) during N_2 desorption by purge. Being the primary energy-consuming step and an important overall-separation-efficiency-determining step (Sircar and Golden 1995), desorption-by-purge step, however, received less attention compared to the thermodynamically spontaneous adsorption step.

1.2.1 The Roles of Desorption-by-Purge Step in RPSA Cycle of MOC

The RPSA process cycles used in the design of MOC are often similar to or variations of the classic Skarstrom (1960, 1966) PSA cycle, in which adsorbent bed is regenerated by pressure reduction followed by back purge with a part of the O_2-enriched product gas. The back purge step is critical for a PSA process design because (a) it provides an efficient mean of N_2 desorption when its partial pressure is brought lower by the presence of O_2-enriched gas, and (b) it creates a section of the column at the O_2 product end which is equilibrated with the O_2-enriched product gas at the start of an adsorption step. Cause (b) is critical for maintaining the product O_2 purity during the adsorption step.

1.2.2 The Actual Purge Volume

To increase O_2 recovery, the molar amount of O_2 product gas used for purge-regenerating adsorbent bed should be minimized. The working principle of PSA prevails because its regeneration by purge is determined by the actual purge volume instead of the molar amount (Sircar and Golden 1995). A rule of thumb to ensure complete purge is to have purge to feed volume ratio, at their respective pressures, be at least equal to one (Yang 2003). Purge to feed ratio is defined as the amount of light component (O_2) in purge to that in feed (Yang 1987). Therefore, employing purge gas at very low total pressure, which gives large volume at low molar amount, can displace the volume of the feed gas at high adsorption pressure (Skarstrom 1972).

1.2.3 Thermal, Pressure Drop, Selectivity Effects on Purge Efficiency

The effect of resistances, particularly pressure drop and non-isothermality, onto desorption processes, were investigated through some earlier work using models and/or experiment as listed below.

Sircar and Kumar (1985) compared local equilibrium models of adiabatic-isobaric and isothermal-isobaric for desorption of bulk binary gas mixtures by purge and showed that the assumption of column isothermality during the purge process can severely underestimate the quantity of the purge gas requirement of an adiabatic column.

Sircar and Golden (1995) developed an extended local equilibrium model for isothermal-isobaric desorption of bulk binary gas mixtures by purge and compared it with an experimentally evaluated isothermal-isobaric desorption by purge. The purge gas requirement of the model was slightly underestimated which might due to imperfect isothermality in the experimental system. Nonetheless, both model and experiment yielded consistent conclusions in which the efficiency of desorption by purge can be increased by lowering the desorption pressure, increasing adsorbent temperature and adsorption selectivity of purge gas over the desorbing component.

Kim et al. (1995) showed that low desorption pressure can reduce the amount of purge gas required for desorption.

Chahbani and Tondeur (2001) numerically studied the effect of pressure drop on the dynamic behavior of a fixed-bed adsorber during adsorption and desorption steps for constant volume and molar flow rate at the bed inlet. It appeared that pressure drop leads to an overconsumption of purge gas, thus pressure drop is unfavorable to regeneration.

Kupiec et al. (2009) developed an exact model and a lumped model, the former included momentum balance and the latter assumed isobaric process. The predictions of both models were compared with experimental results and a satisfactory agreement was found only in case of the exact model. The lumped model tended to overestimate the efficiency of regeneration of bed during purge step.

1.2.4 Pressure Drop, Thermal, Axial Dispersion Effects on Process Performance

In addition to the specific desorption step, some reported effects of resistances onto the process performance using models and/or experiment are listed below.

Sundaram and Wankat (1988) investigated the pressure drop effects during the pressurization and blowdown steps using linearized Darcy's law assuming linear equilibrium, negligible mass transfer resistance, negligible dispersion and isothermal operation. Pressurization demonstrated shock wave behavior while blowdown exhibited diffuse wave behavior, the latter being more time consuming as demonstrated by an equilibrium model for isothermal-nonisobaric operation by Lu et al. (1993). Blowdown solutions also indicated a non-uniform composition in the column even at large cycle time.

Buzanowski et al. (1989) performed some experimental and numerical evaluations of the effect of pressure drop for isothermal adsorption and desorption steps and concluded that pressure drop delayed breakthrough, sharpened the desorption curves and improved desorption along the column.

Kikkinides and Yang (1993) critically evaluated and corrected the work done by Buzanowski et al. (1989) using analytical solutions, detailed numerical solutions, and experimental breakthrough curves. It was concluded that pressure drop along the column always results in earlier breakthrough and distorted constant pattern behaviors during adsorption step.

Yang et al. (1998) also came to the same conclusion in their theoretical studies that pressure drop led to earlier breakthrough. Moreover, the effect of pressure drop during the adsorption step under the non-isothermal condition was slightly larger than that under isothermal and adiabatic conditions.

High pressurization rate, though giving a little higher productivity rate, increases dispersion in the PSA column thus decreases product purity and recovery (Mendes et al. 2000).

It is seen that most models in the literatures do not incorporate all the possible resistances thus resulting in overestimation of system performance.

1.2.5 Sensitivity of Back Purge Gas Quantity to PSA Process Performance

Given in this section is an overall and component mass balances performed on an idealized PSA process producing pure O_2 from air consisting of 79% N_2 + 21% O_2 in order to demonstrate the sensitivity of the specific amount of back purge gas (moles/kg of sorbent) used in a generic Skarstrom-like PSA cycle in determining BSF and R. Two other schemes that reach to the same conclusion are presented in Section 7.3 of Chapter 7.

The cyclic process steps are: (A) counter-current column back purge at ambient pressure using pure O_2, (B) counter-current column pressurization from ambient to adsorption pressure level of P_A with pure O_2, (C) co-current adsorption from compressed air at pressure P_A to produce a pure O_2 effluent gas at pressure P_A, and (D) counter-current column depressurization from pressure P_A to ambient pressure. A part of the effluent gas from step C is withdrawn as the net pure O_2

product by the process and the balance is used in steps A and B. The effluent gases from steps A and D are wasted.

Using N_2 = component 1, O_2 = component 2, the ideal model assumptions include (a) isothermal operation, (b) local thermodynamic equilibrium (mass and heat) between the gas and solid phases in the column at all times during the cycle, and (c) plug flow of gas with no column pressure drop or axial dispersion.

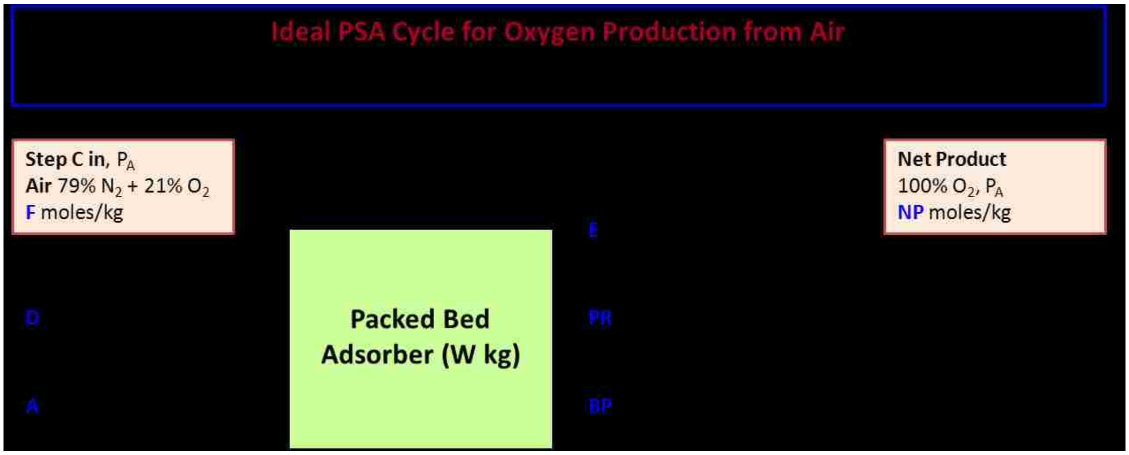

Ideal PSA Cycle for Oxygen Production from Air
Step C in, P_A
Air 79% N_2 + 21% O_2
F moles/kg
Net Product
100% O_2, P_A
NP moles/kg
E
D
Packed Bed Adsorber (W kg)
PR
A
BP

Figure 1.5: Block diagram of an ideal O_2 PSA system.

Figure 1.5 is a block diagram of the model showing various influent and effluent flows and quantities (moles/kg of total adsorbent in the PSA system/cycle) for the steps. The key performance variables for the process are:

$$BSF = 1/(NP.\omega) \tag{1.6}$$

$$R = NP/0.21F \tag{1.7}$$

where ω (= $1/t_c$) is the cycle frequency (number of cycles per unit time) and t_c is the total cycle time.

It can be shown by component mass balances (N_2 = 1; O_2 =2) for various steps that:

$$F = N_1^C/0.79 \tag{1.8}$$

$$PR = \ N_2^B - N_2^A \tag{1.9}$$

$$E = PR + BP + NP = 0.21F + \ N_2^B - N_2^C \tag{1.10}$$

$$NP = \ \alpha - BP \ ; \alpha = \ E - PR \tag{1.11}$$

where $N_i^j \ = n_i^j + \varepsilon\rho_g^j y_i^j/\rho_b$ is the total (adsorbed + void gas) specific amount (moles/kg) of component i in the column at the end of step j, n_i^j is the specific equilibrium amount (moles/kg) of component i adsorbed at the end of step j when the gas phase density (moles/cm^3) is $\rho_g^j \ = P^j/RT$ and the gas phase mole fraction of component i is y_i^j. P^j (atm) is the column pressure at the end of step j, and ε is the helium void fraction of the column. T is the system temperature. R is the gas constant. Variables$F$, E, PR and α can be estimated using equations (1.8) – (1.11) and the adsorption isotherms of pure N_2 and O_2 and their mixtures at T.

It follows from equations (1.6), (1.7), and (1.11) that:

$$\frac{\Delta BSF}{BSF} \ = \ \frac{\Delta R}{R} \ = \beta \ \frac{\Delta BP}{BP} \tag{1.12}$$

$$\beta = \ \frac{\alpha}{0.21F(R)} - 1 \tag{1.13}$$

Equations (1.12) and (1.13) indicate that the changes in the absolute values of the parameters (BSF) and (R) due to a change in the value of (BP) will be magnified by a factor of β (> 1). Thus, the amount of back purge used in a PSA process is a sensitive variable for determining the overall performance of a PSA process.

Using the published adsorption isotherms of pure N_2 and O_2 and their mixtures on a sample of ~ 100% Li exchanged X zeolite, which could be described by an empirical Langmuir-like model (Rege and Yang 1997), in order to derive the following relationship between the parameters β and R for the above-described PSA cycle by assuming that the adsorption pressure (P_A) is 5 atm and the system temperature (T) is 298 K:

$$\beta = \frac{1.31}{R} - 1 \tag{1.14}$$

According to equation (1.14), the value of the parameter β is 4.24 if the base case O_2 recovery is 25% (R = 0.25), which is typical for an O_2 RPSA system (Chai et al. 2011). Consequently, it follows from equations (1.12) and (1.14) that a ~ 10% change (or estimation error) in the quantity of back purge (BP) from a base case value can produce a ~ 42.4% change (or estimation error) in BSF or R from those of the base case values. This clearly demonstrates that the quantity of back purge in a PSA process is a very sensitive design variable in establishing the overall process performance, albeit for an idealized O_2 PSA process.

Hence, the second main objective of this dissertation research is to carry out a systematic numerical analysis of the transfer resistances (non-isobaric, non-isothermal, and non-equilibrium) in the critical desorption-by-purge step.

1.3 Dissertation Research Goals

The goal is to investigate the limitations of adsorbent bed size reduction of a MOC using experimental simulated RPSA cycle and analyze the purge efficiency in desorption-by-purge step using rigorous model simulation. The integral parts of the dissertation are:

(a) Experimentally evaluate the cyclic process performance:

- intelligent design of the PSA system containing a minimum number of very small switching valves, equipped with process control and data acquisition system
- develop experimental protocols
- find out the patterns and limitations of bed size reduction and the corresponding O_2 recovery and BSF

(b) Mathematical modeling and simulation of purge desorption step:

- systematically evaluate the order of resistances that affect purge efficiency
- determine the key factors in defining efficient cleaning
- investigate the purge efficiency at varying adsorbent particle sizes, purge durations, in different bed geometries

This study will define some practical boundaries on adsorber size, PSA cycle time, desorption-by-purge time, adsorption pressure, adsorbent particle size, bed geometry for a small-scale RPSA MOC.

1.4 References

Ackley, M.W., Zhong, G.M.: Medical Oxygen Concentrator. U.S. Patent 6,551,384 (2003).

Adsorbents: Fundamentals and Applications. Edited by Yang, R.T. John Wiley & Sons, Inc. (2003).

American Lung Association: Chronic Obstructive Pulmonary Disease (COPD) Fact Sheet. (August 2006). http://www.lungusa.org/site/pp.asp?c=dvLUK9O0E&b=35020

Batta, L.B.: Selective Adsorption Gas Separation Process. U.S. Patent 3,636,679 (1972).

Buzanowski, M.A., Yang, R.T., Haas, O.W.: Direct Observation of the Effects of Bed Pressure Drop on Adsorption and Desorption. *Chem. Eng. Sci.*, **44**, 10, 2392 – 2394 (1989).

Chahbani, M.H., Tondeur D.: Pressure Drop in Fixed Bed Adsorbers. *Chem. Eng. J.*, **81**, 23 – 24 (2001).

Chai, S.W., Sircar, S., Kothare, M.V.: Miniature Oxygen Concentrators and Methods. U.S. Patent Application 2010/0300285 (2010).

Collins, J.J.: Air Separation by Adsorption. U.S. Patent 3,973,931 (1976).

Dobson, M.B.: Oxygen Concentrators for District Hospitals. John Radcliffe Hospital, Oxford, U.K. **10**, 11 (1999).

Gas Separation by Adsorption Processes. Edited by Yang, R.T. Butterworths Series in Chemical Engineering (1987).

Jagger, T.W., Van Brunt, N.P., Kivisto, J.A., Lonnes, P.B.: Personal Oxygen Concentrator. U.S. Patent 7,121,276 (2006).

Jiang, Z.P., Abraham, K.: Oxygen Batteries based on a Solid Polymer Electrolyte. *NASA Tech Briefs* (August 2000).
http://findarticles.com/p/articles/mi_qa3957/is_/ai_n8910280

Kikkinides, E.S., Yang, R.T.: Effects of Bed Pressure Drop on Isothermal and Adiabatic Adsorber Dynamics. *Chem. Eng. Sci.*, **48**, 9, 1545 – 1555 (1993).

Kim, J.N., Chue, K.T., Cho, S.H.: Production of High Purity Nitrogen from Air by Pressure Swing Adsorption on Zeolite X. *Sep. Sci. Tech.*, **30**, 3, 347 – 368 (1995).

Kopaygorodsky, E.M., Guliants, V.V., Krantz W.B.: Predictive Dynamic Model of Single-Stage Ultra-Rapid Pressure Swing Adsorption. *AIChE J. Separations*, **50**, 953 (2004).

Kulish, S., Swank, R.P.: Rapid Cycle Pressure Swing Adsorption Oxygen Concentration Method and Apparatus. U.S. Patent 5,827,358 (1998).

Kupiec, K., Rakoczy, J., Lalik, E.: Modeling of PSA Separation Process including Friction Pressure Drop in Adsorbent Bed. *Chem. Eng. Pro.*, **48**, 1199 – 1211 (2009).

Leavitt, F.W.: Air Separation Pressure Swing Adsorption Process. U.S. Patent 5,074,892 (1991).

Leavitt, F.W.: Vacuum Pressure Swing Adsorption Process. U.S. Patent 5,415,683 (1995).

Lu, Z.P., Loureiro, J.M., Rodrigues, A.E.: Pressurization and Blowdown of Adsorption Beds – II. Effect of the Momentum and Equilibrium Relations on Isothermal Operation. *Chem. Eng. Sci.*, **48**, 9, 1699 – 1707 (1993).

McCombs, N.R., Bosinski, R., Casey, R.E., Valvo, M.R.: Mini-Portable Oxygen Concentrator. U.S. Patent Application 2006/0117957 (2006).

Mendes, A.M.M., Costa, C.A.V., Rodrigues, A. E.: Analysis of Nonisobaric Steps in Nonlinear Bicomponent Pressure Swing Adsorption Systems. Application to Air Separation. *Ind. Eng. Chem. Res*., **39**, 138-145 (2000).

Moulijn, J.A., Van Swaaij, W.P.M.: The Correlation of Axial Dispersion Data for Beds of Small Particles. *Chem. Eng. Sci.*, **31**, 845 (1976).

Neill, A., Leavitt, F.W., Figueiredo, A.F.Z.: Oxygen Recovery Pressure Swing Adsorption Process. U.S. Patent 5,711,787 (1998).

No Oxygen at Prince Mshiyeni. *East Coast Radio Newswatch* (16 July 2008). http://blog.ecr.co.za/newswatch/?p=2182

OGSI Leads the Way in Onsite Oxygen Technology for Aquaculture (2008). http://www.ogsi.com/aquaculture_fish_farming_.php

Patterson, J.M.: Liquid Oxygen Supplements Provide Life-Giving Health Benefits. *Health-and-Fitness/Supplements* (2008). http://ezinearticles.com/?Liquid-Oxygen-Supplements-Provide-Life-Giving-Health-Benefits&id=1636264

Porter, K.E., Ali, Q.H., Hassan, A.O., Aryan, A.F.: Gas Distribution in Shallow Packed Beds. *Ind. Eng. Chem. Res.*, **32**, 2408 (1993).

Rama Rao, V., Farooq, S., Krantz, W.B.: Design of a Two-Step Pulsed Pressure Swing Adsorption Based Oxygen Concentrator. *AIChE J. Separations*, **56**, 354 (2010).

Rege, S.U., Yang, R.T.: Limits for Air Separation by Adsorption with LiX Zeolite. *Ind. Eng. Chem. Res.*, **36**, 5358 (1997).

Santos, J.C., Portugal, A.F., Magalhaes, F.D., Mendes, A.: Simulation and Optimization of Small Oxygen Pressure Swing Adsorption Units. *Ind. Eng. Chem. Res.*, **43**, 8328 (2004).

Santos, J.C., Portugal, A.F., Magalhaes, F.D., Mendes, A.: Optimization of Medical PSA Units for Oxygen Production. *Ind. Eng. Chem. Res.*, **45**, 1085 (2006).

Sircar, S.: Air Fractionation by Adsorption. *Sep. Sci. Tech.*, **23**, 2379 (1988).

Sircar, S.: Gas Separation by Rapid Pressure Swing Adsorption. U.S. Patent 5,071,449 (1991).

Sircar, S.: Influence of Gas-Solid Heat Transfer on Rapid PSA. *Adsorption*, **11**, 509 (2005).

Sircar, S.: Rapid Pressure Swing Adsorption Process for Production of Oxygen Enriched Air. *Research Disclosure*, **59** (1996).

Sircar, S., Golden, T.C.: Isothermal and Isobaric Desorption of Carbon Dioxide by Purge. *Ind. Eng. Chem. Res.*, **34**, 2881(1995).

Sircar, S., Hanley, B.F.: Production of Oxygen Enriched Air by Rapid Pressure Swing Adsorption. *Adsorption*, **1**, 313 (1995).

Sircar, S., Kumar, R.: Equilibrium Theory for Adiabatic Desorption of Bulk Binary Gas Mixtures by Purge. *Ind. Eng. Chem. Process Des. Dev.*, **24**, 2, 358 – 364 (1985).

Sircar, S., Myers, A.L.: Gas Separation by Zeolites. In *Handbook of Zeolite Science and Technology*; Auerbach, S.M., Carrado, K.A.; Dutta, P.K., Eds.; Marcel Dekker: New York, Chap. 22; pp 1063-1105 (2003).

Sircar, S., Naheiri, T., Fischer, J.R. Oxygen Generation Process and System using Single Adsorber and Single Blower. U.S. Patent 6,146,447 (2000).

Sircar, S., Rao, M.B., Golden, T.C.: Fractionation of Air by Zeolites. In *Adsorption and its Applications in Industry and Environmental Protection*; Dabrowski, A., Ed; Elsevier: New York, **120**, Part 1; pp 395-423 (1999).

Sircar, S., Zondlo, J.W.: Fractionation of Air by Adsorption. U.S. Patent 4,013,429 (2003).

Skarstrom, C.W.: Method and Apparatus for Fractionating Gaseous Mixtures by Adsorption. U.S. Patent 2,944,627 (1960).

Skarstrom, C.W.: Oxygen Concentration Process. U.S. Patent 3,237,377 (1966).

Skarstrom, C.W.: Recent Development in Separation Science. CRC Press Cleveland, **2**, 95 (1972).

Solid Bed Dehydrator - Bed Height to Diameter Ratio. http://www.process-facility.com (accessed March 1, 2011).

Sundaram, N., Wankat, P.C.: Pressure Drop Effects in the Pressurization and Blowdown Steps of Pressure Swing Adsorption. *Chem. Eng. Sci.*, **43**, 1, 123 – 129 (1988).

Teague, K.G. Jr., Edgar, T.F.: Predictive Dynamic Model of a Small Pressure Swing Adsorption Air Separation Unit. *Ind. Eng. Chem. Res.*, **38**, 3761(1999).

Yang, J., Park, M.W., Chang, J.W., Ko, S.M., Lee, C.H.: Effects of Bed Pressure Drop in a PSA Process. *Korean J. Chem. Eng.*, **15**, 2, 211 – 216 (1998).

Zhong, G.M., Rankin. P.J., Ackley, M.W.: High Frequency PSA Process for Gas Separation. U.S. Patent 7,828,878 (2010).

Chapter 2

Preliminary Work I: Literature Reviews, Theoretical Studies, and Conceptual Design of Adsorbent Packed Beds

Section 2.1 focuses on literature reviews and theoretical studies supplemented with simple model simulations in order to investigate the basic and key factors for system design. Zeolite based air separation is introduced by zeolite type and its selection, activation and pretreatment, followed by adsorption equilibriums and kinetics studies that correlate to adsorbent loading capacity, working pressure, step time, particle size etc. Brief reviews about pressure drop, heat effect, Skarstrom cycle and performance variables are also included. These preliminary studies assist the conceptual design of adsorbent packed beds in Section 2.2. A side study of the potential use of cobalt complexes for air separation is presented in Section 2.3.

2.1 Literature Reviews, Theoretical Studies

The key design components for zeolite based air separation are generally reviewed in this section.

2.1.1 Zeolite Adsorbent and Selection

PSA bed contains adsorbent (sieve material) with certain selectivity to retain specific component(s) from a stream of mixture while allowing the remaining component(s) to pass through it. Many zeolites including type A (Ca-Na exchanged

5A, high Ca exchanged A), type X (Na, Li, Ca, Sr, Ba exchanged), low silica X (LSX) (Li, Li-Zn exchanged), and mordenite (Na, Ca exchanged) have been commercially used for separation of air, composed of N_2, O_2, argon (Ar), trace amount of carbon-dioxide (CO_2) and water content (H_2O). These zeolites are all highly polar materials, and H_2O and CO_2 are generally removed from the feed air prior to air separation by using a layer of a desiccant (NaX zeolite or alumina) at the feed gas end of the adsorber. The selectivities of adsorption exhibited by these zeolites for the components of air are in the order $N_2 > O_2 \geq Ar$, but they exhibit significantly different thermodynamic sorptive properties (equilibrium isotherms and heats) for sorption of these gases (Sircar 1988; Sircar et al. 1999; Sircar and Myers 2003).

Selection of the optimum zeolite for a PSA process can be fairly complex (Sircar et al. 1998), this can be explained by the adsorption isotherms (equilibrium loading at constant temperature) of N_2 and O_2 onto different zeolites in Figure 2.1 (Sircar and Myers 2003). Zeolite that has high N_2 selectivity, such as CaLSX, is not always the best choice for air separation because of increased difficulty to desorb N_2. A much deeper vacuum and more O_2 purge gas are necessary to remove N_2 from this adsorbent. On the contrary, though giving lower equilibrium loading, LiLSX portrays a relatively linear capacity with respect to pressure. By swinging the pressure between low and high pressures, LiLSX is able to give the highest working capacity without facing much difficulty in desorption. Thus, LiX, Li-LSX and Li-ZnX are currently identified as the best adsorbent in industry for O_2 concentration (Sircar and Myers 2003; Hutson et al. 1999; Leavitt 1991).

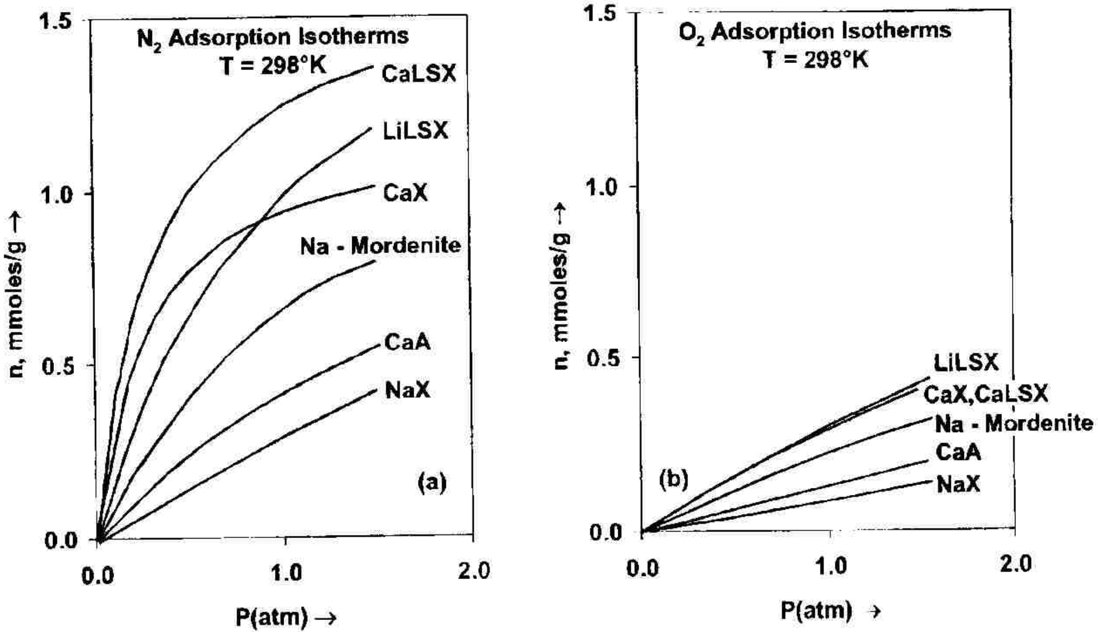


Figure 2.1: Adsorption isotherms for N_2 and O_2 onto different zeolites (Sircar and Myers 2003).

2.1.2 Zeolite Activation and Air Pretreatment

A small amount of polar adsorbate like water can significantly reduce thermodynamic and kinetic adsorptive properties of zeolite (Sircar et al. 1998), therefore, zeolite activation (regeneration) must be performed prior to operation. Some activation methods include heating at 260^{o}C while being purged by dry air (Skarstrom 1966), vacuum dehydration at 450^{o}C for at least 4 hours (Hutson et al. 1999); heating at 540^{o}C for 1 hour (Wang et al. 2006); and heating at 300 – 650^{o}C while being purged by dry N_2 (Hees et al. 1999). Nonetheless, for the near fully exchanged Lithium (Li) material, there is no significant increase in N_2 capacity with dehydration at 450^{o}C over that at 350^{o}C. This is because majority of the zeolie water is easily removed by 250^{o}C, and most tenaciously held water molecules are removed by 350^{o}C (Hutson et al. 1999; Breck 1984).

Similarly, air needs to be dehumidified too before entering into PSA system, examples are using water selective desiccants such as Na-X zeolite (Sircar et al. 1998), 5A zeolite and silica gel (Mendes et al. 2001) at the upstream of feed line. The ratio of the amount of desiccant to zeolite used is typically between 0.2 – 0.5 (Sircar et al. 1998).

2.1.3 Adsorption Equilibriums

In practical operations, maximum capacity of adsorbent cannot be fully utilized because of the mass transfer effects in actual gas-solid contacting process. In order to estimate practical or dynamic adsorption capacity, however, it is essential to have information on adsorption equilibriums. After that, kinetic analyses are conducted because diffusion rate is generally secondary in importance (Motoyuki 1989; Yang 2003).

Based on isotherms, the important factors for design of the separation process, such as (1) capacity of the adsorbent in the operating pressure and temperature range, (2) the magnitude of pressure swing for regeneration, (3) the length of the unusable bed (LUB), and (4) the product purity, can be determined. LUB is primarily determined by equilibrium isotherm, it is approximately one half the span of the concentration wavefront, or mass transfer zone. A sharp wavefront, or a short LUB, is desired because it results in high adsorbent productivity and high product purity (Yang 2003). In this study, only the capacity and the pressure range are examined.

Since LiLSX (or LiX Si/Al=1.0) is identified as the best adsorbent in industry for separation of air (Sircar et al. 1998; Hutson et al. 1999), its working capacity for N_2 and O_2 is studied via the adsorption isotherms. Extended Langmuir model represents this system decently. This model assumes that each species maintains its own molecular area (the area covered by one molecule that is not influenced by the presence of other species on the surface) as that in pure component adsorption (Hutson et al. 1999, Yang 2003). The adsorption isotherms in Figures 2.2 and 2.3 are simulated using the equilibria model from Hutson et al. (1999).

Figure 2.2 shows the amount of N_2 and O_2 adsorbed with respect to their partial pressures, while Figure 2.3 demonstrates the N_2 and O_2 adsorbed with respect to total adsorption pressure (P_T), comprises partial pressures of $0.78P_T$ for N_2 and $0.22P_T$ for O_2. From Figure 2.2, N_2 and O_2 adsorbed onto LiLSX at elevated pressure, but LiLSX is highly selective for N_2. On the other hand, Figure 2.3 shows that total working pressure of air within the range of 0 – 1 atm on LiLSX is favorable because equilibrium loading of 1.1117 mmol N_2/g adsorbent can be achieved within a nearly linear regime. If operating pressure is extended to 2 atm, instead of giving doubled capacity, the maximum loading is only about 1.5550 mmol N_2/g adsorbent. Therefore, judicious decision of pressure envelop for miniature O_2 concentrator via PSA based on these equilibrium isotherms is necessary to balance between the compression cost and O_2 recovery.

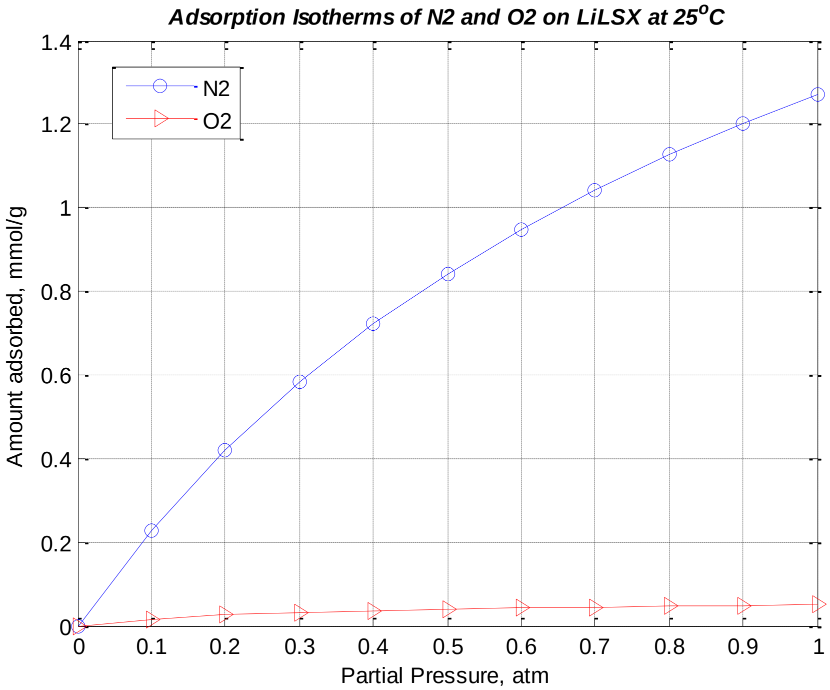


Figure 2.2: N_2 and O_2 adsorbed versus partial pressures. Simulated using the equilibria model from Hutson et al. (1999).

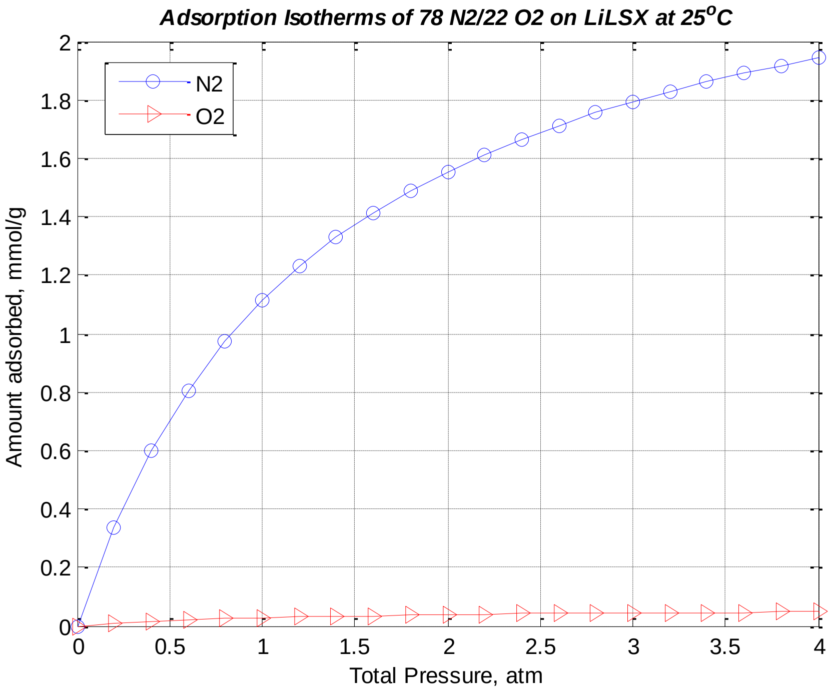


Figure 2.3: N_2 and O_2 adsorbed versus total adsorption pressures. Simulated using the equilibria model from Hutson et al. (1999).

Reversible equilibrium capacity is another important consideration of choosing adsorbent type. Reversible equilibrium capacity simply means the capacity remains the same regardless of the forward (adsorption) or reverse (desorption) direction of PSA, as demonstrated by the N_2 isotherms of zeolite molecular sieves in Figure 2.4, obtained from Du and Wu (2007).

PSA separation of air using LiX or LiLSX zeolite is an equilibrium separation. However, it does not imply that the adsorbent can achieve equilibrium capacity. Equilibrium separation is simply a separation mechanism based on thermodynamic properties (e.g. quadrupole moment) rather than based on diffusion rates, which is called kinetic separation. On the contrary, adsorption isotherm represents equilibrium capacity of the adsorbent at certain pressure and temperature. Only under the conditions free of diffusion limitation, each contact between gas molecules and adsorbent is equivalent to an equilibrium stage or theoretical plate (Yang 2003).

To sum up, equilibrium isotherms of N_2 and O_2 help us to identify the working capacity with respect to operating pressure. In order to check whether the system of separating N_2 and O_2 using LiX with specific particle size and at designated step time can work near to local equilibrium conditions, thus giving maximum loading capacity, kinetics analyses in the adsorbent pellet is carried out in the next section.

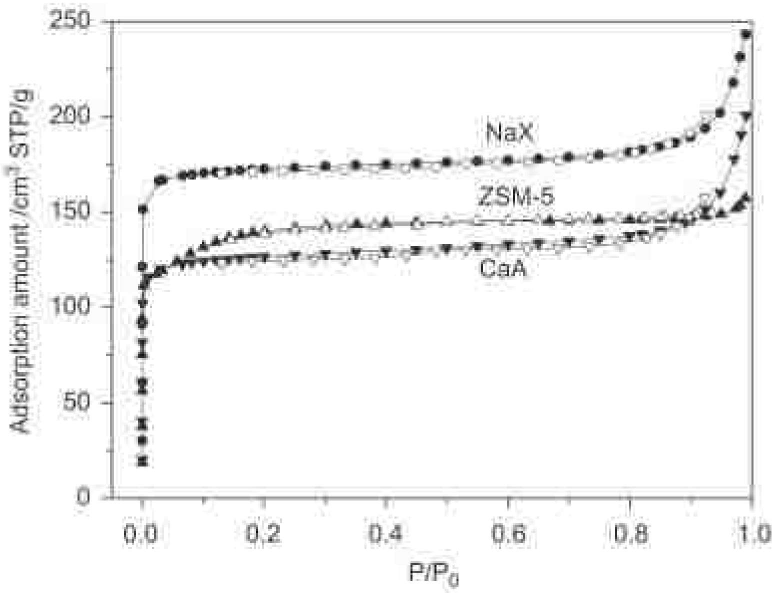
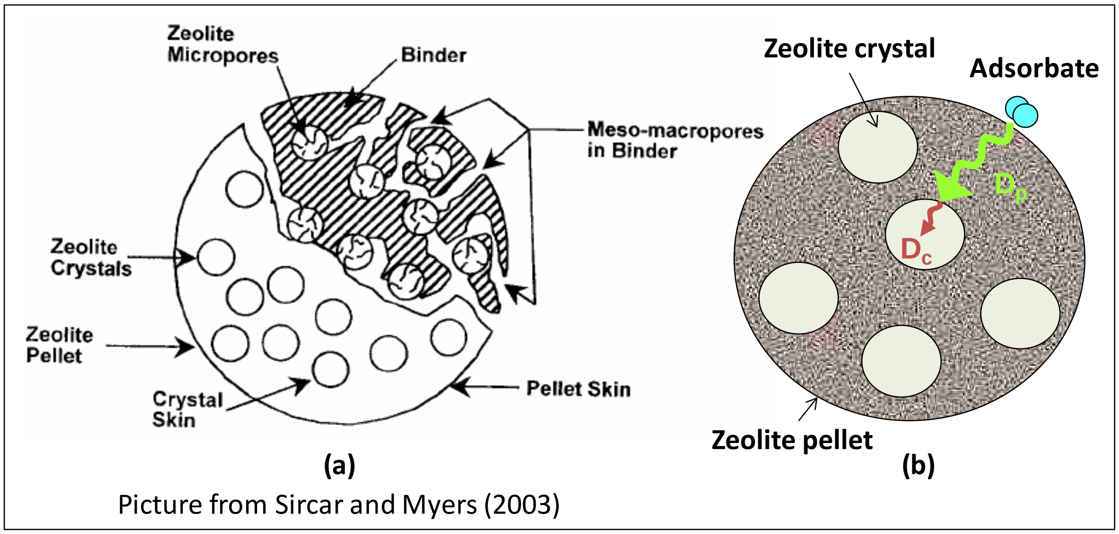


Figure 2.4: N_2 adsorption isotherms of zeolite molecular sieves NaX, CaA, and ZSM-5 at 77K. Closed symbol: adsorption. Open symbol: desorption. (Du and Wu 2007)



Picture from Sircar and Myers (2003)

Figure 2.5: (a) Structure of a bound zeolite pellet. (b) Adsorbate diffusion through pellet and crystal.

2.1.4 Adsorption Kinetics

From the adsorption isotherm analysis above, the extent of reaching maximum loading capacity of an adsorbent within a finite cycle time remains unknown unless an understanding in the adsorption kinetics is gained. Generally, the rates of adsorption and desorption of an adsorbate onto and from an adsorption site in the zeolite crystal are extremely rapid, of the order of microseconds. However, the transport of adsorbate from the gas phase outside the particle to the adsorption sites inside a crystal is slowed by series and parallel of mass transfer resistances. They are (1) external gas film resistance (for mixtures), (2) diffusional resistance through the meso- and macropores to the surface of the zeolite crystals, (3) diffusional resistance through the micropores of the zeolite crystals to the adsorption sites, (4) possibly skin resistance at the surfaces of the adsorbent pellet and/or zeolite crystal (Sircar and Myers 2003). In this section, diffusion resistances in an adsorbent pellet [which includes (2) and (3)] will be examined because transportation in narrowed routes imposes the biggest resistance thus dominates the mass transfer rate. Before looking into the kinetic issues, a single adsorbent particle is scrutinized.

Commercial zeolites are generally available in bound forms where the zeolite crystals (1 – 5 µm in diameter) are formed in regular particle shape using a binder material such as clay, alumina or polymer. The purpose of the bound forms (diameter of 0.5 – 6 mm are common) is to reduce the pressure drop in adsorbent bed. The binder phase contains a network of meso- and macropores (0.5 – 5 µm in diameter) to facilitate transport of the adsorbate molecules from the external gas phase to the mouths of the zeolite crystal pores (Sircar and Myers 2003). Then, micropores in zeolite crystal facilitate adsorbate to the binding sites. Figure 2.5 (a) is a schematic diagram of a bound zeolite particle showing the transport path of a gas molecule during the adsorption or desorption process. D_p and D_c represent the diffussivities of adsorbate in pellet and in crystal, repsectively.

As depicted in Figure 2.5 (b), adsorption kinetic within an adsorbent particle consists of intra-pellet and intra-crystalline diffusions, but in most cases it is controlled by the intra-pellet diffusion (Sircar and Myers 2003; Motoyuki 1989) because of the extremely small size of zeolite crystal (crystal radius, r_c ~ 1μm) which renders to large diffusional time constant in crystal, $\tau_c = D_c/r_c^2$, despite its smaller diffusivity (D_c ~ 10^{-4} – 10^{-10} cm^2/s) compared to that in pellet (D_p ~ 10^{-2} – 10^{-4} cm^2/s) (Sircar and Myers 2003).

In order to investigate the extent of reaching maximum loading capacity of an adsorbent at a finite adsorption step time, fractional adsorbate uptake is defined. While n_A^* is the equilibrium specific amounts (moles/lb) adsorbed, $\bar{n}_A$ and $\bar{n}_D$ are the average specific amounts adsorbed at the end of adsorption and desorption steps under the cyclic steady-state condition (Sircar and Hanley 1995). Fractional uptake is defined as below, with t_A the step time for the adsorption.

Fractional Adsorbate Uptake: $f(t_A)$ [redacted] (2.1)

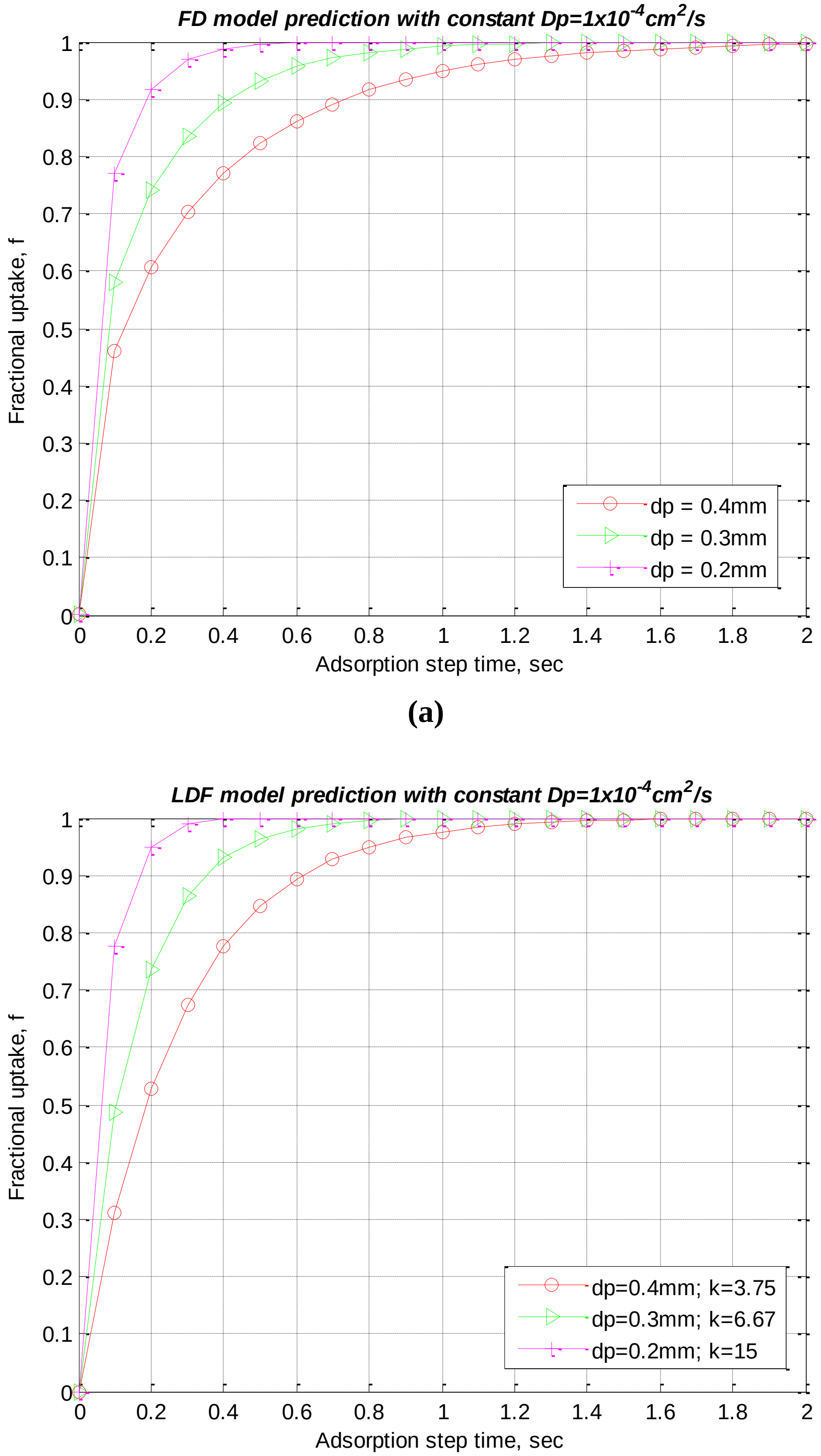
Fickian Diffusion (FD) and Linear Driving Force (LDF) models can be used to simulate the fractional uptake of adsorbate by the zeolite pellet or crystal. In this case, the controlling intra-pellet diffusion will be investigated. Given below are the FD and LDF solutions for isothermal adsorption on a spherical zeolite particle from a gas phase maintained at constant pressure (Sircar and Myers 2003).

FD solution: $$f(t_A)\,\text{[redacted]}\,\frac{6}{\text{[redacted]}}\,e^{\text{[redacted]}\,t_A} \qquad \text{[redacted]}\,\frac{D_p}{r_p^2} \tag{2.2}$$

LDF solution: $$f(t_A)\,\text{[redacted]}\,t_A \qquad k\,\text{[redacted]}\,\frac{15D_p}{r_p^2} \tag{2.3}$$

where r_p is the pellet radius (cm), D_p is the intra-pellet diffusivity (cm^2/sec), [redacted] is the diffusional time constant (sec^{-1}), k is the mass transfer coefficient (sec^{-1}), t_A is the adsorption step time and f is the fractional uptake (dimensionless). These models indicate that loading capacity of an adsorbent is adsorption step time, mass transfer and pellet size dependent.

The profiles of fractional uptake versus adsorption step time in Figure 2.6 were generated for different particle sizes (d_p) using a constant diffusivity value at the order of magnitude provided by Sircar and Myers (2003). From these simulation results, we can see that FD and LDF models agree to each other closely in predicting the uptake of adsorbate by an adsorbent particle. From the given diffusivity in meso and macropores, a single zeolite pellet is saturated by adsorbate within seconds, with shorter time possible as the pellet size goes smaller. Note that LDF mass transfer coefficient, *k*, grows larger as particle size decreases due to reduced transport resistance.



(b)

Figure 2.6: Fractional adsorbate uptake by a single particle using the solutions from (a) Fickian Diffusion model and (b) Linear Driving Force model.

Since adsorption and desorption mechanisms can happen instantaneously in the absence of mass transfer resistance, it is expected that production rate will increase exponentially if the process is sped up indefinitely. With the inclusion of finite mass transfer coefficients, the pattern of increased productivity in a single particle as cycle time is reduced is investigated here to define net working capacity and net rate of adsorbate removal. Assuming that the process cycle consists of adsorption and desorption with the same step time, t_A [redacted] $_D$ [redacted], then the net working capacity N (moles/lb) and net rate of adsorbate removal R (moles/lb/time) under cycle time $2\bar{t}$ are given by (Sircar and Hanley 1995)

Net Working Capacity:
$$\frac{N(\bar{t})}{N*}\ \text{[redacted]}\ \frac{1\ \text{[redacted]}\ \text{[redacted]}\bar{t}}{1\ \text{[redacted]}\ \text{[redacted]}\bar{t}} \tag{2.4}$$

Net Rate of Adsorbate Removal:
$$\frac{R}{N*}\ \text{[redacted]}\ \frac{1}{2\bar{t}}\frac{1\ \text{[redacted]}\ \text{[redacted]}\bar{t}}{1\ \text{[redacted]}\ \text{[redacted]}\bar{t}} \tag{2.5}$$

where N [redacted] $_A$ [redacted] $_D$ and $N*$ [redacted] $^*_A$ [redacted] $^*_D$ are the specific steady state working capacity under the presence and absence of mass transfer resistance, respectively. $N*$ is also regarded as equilibrium capacity with f [redacted].

Figure 2.7 shows the net rate of adsorbate removal with respect to step time, $\bar{t}$, under the given mass transfer coefficients, ranging from low to high values, corresponding to high to low resistances. It is observed that in the absence of adsorbate mass transfer resistance where k [redacted]0000 sec^{-1} (local equilibrium case), the net rate of adsorbate removal increases indefinitely as $\bar{t}$ is reduced. On the other hand, if mass transfer resistance is prevalent where k [redacted].1 sec^{-1}, no substantial gain in the net rate of adsorbate removal by the adsorbent particle can be achieved by reducing $\bar{t}$. For k ranging from 3.75 to 15 sec^{-1}, the net removal rates are the same at $\bar{t}$ [redacted] 1.5 sec and above because 1.5 sec is sufficient to achieve equilibrium condition within that range of mass transfer resistances. However, if $\bar{t}$ is reduced below 1.5 sec, the adsorbent particle with higher mass transfer resistance (lower k) is only able to introduce marginal increase in the net removal rate.

In order to investigate the extent of utilization of an adsorbent particle with finite mass transfer resistance when it is subjected to short cycle time, the local specific working capacity as a function of step time $\bar{t}$ and dimensionless radius x (= r/Z) is studied, where Z is the full radius of the pellet (Sircar and Hanley 1995).

Local Working Capacity:

$$\frac{N(x,\bar{t})}{N^*}\ \text{[redacted]}\ \frac{5e^{\text{[redacted]}}}{1\ \text{[redacted]}}(1\ \text{[redacted]}^{2}) \tag{2.6}$$

Figure 2.8 is a plot of the radial working capacity of a single particle with k = 6.67 sec^{-1} (corresponding to d_p = 0.3 mm and D_p = 1x10^{-4} cm^2/sec) for different values of adsorption step time, $\bar{t}$. x = 0 and x = 1 represent particle core and surface, respectively.

The simulation results in Figure 2.8 show that when the adsorption step time is large, steady state local working capacity in the entire particle approaches the equilibrium working capacity, N^*. On the other hand, when the step time is small, a large variation in the steady state local working capacity can exist within the particle, indicating poor utilization of the adsorbent.

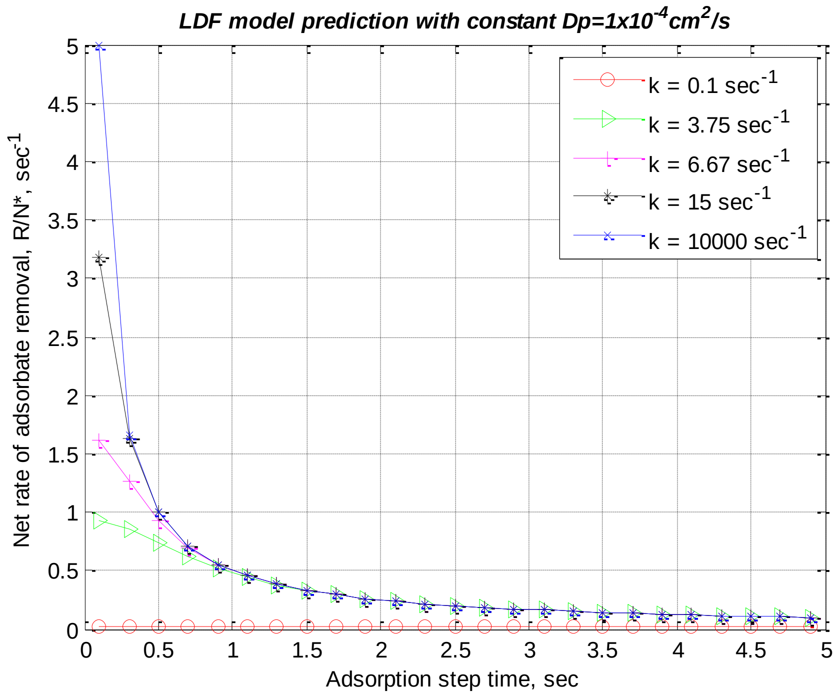


Figure 2.7: Net rate of adsorbate removal by a single particle with different mass transfer coefficients.

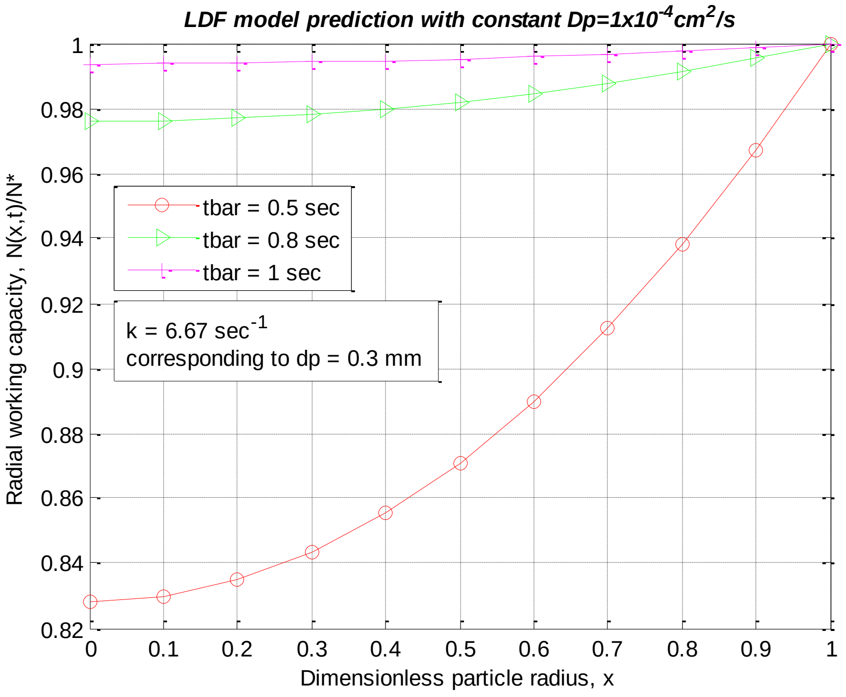


Figure 2.8: Radial working capacity by a single particle with different adsorption step times.

To sum up, this kinetics study details the role of mass transfer resistance in restraining the instantaneous reach of equilibrium capacity of a single adsorbent particle. For maximum utilization of adsorbent, smaller adsorbent particle (has smaller mass transfer resistance) with shorter adsorption step time, or, larger particle (has larger mass transfer resistance) with longer step time will do. However, high pressure drop and lower productivity will be inevitable for the former and latter case, respectively.

2.1.5 Pressure Drop

Increasing the productivity of small PSA bed means increasing feed flow rate and speeding up the nonproductive pressurization and depressurization steps, thus inflicting high pressure drop problem. All these while, pressure drop was neglected in most modeling and simulation studies for conventional PSA. Recently, the effect of pressure drop on the dynamics of gas flow in fixed-bed adsorbers has received a great deal of attention, for conventional and rapid PSA (Sundaram and Wankat 1988; Yang et al. 1998; Doong and Yang 1988). Literature reviews of pressure drop across column for PSA operation has been discussed in Sections 1.2.3 and 1.2.4.

2.1.6 Heat Effect

Temperature fluctuation in the bed due to exothermic heat of adsorption and endothermic heat of desorption is detrimental to separation process. The best separation is accomplished when the bed is operated near isothermal condition. An isothermal separation can be realized by having small bed, short cycle and low throughput per cycle (Yang 2003). Issues of non-isothermality for PSA operation has been reviewed in Sections 1.2.3 and 1.2.4.

2.1.7 Skarstrom Cycle and Its Variations

The first PSA operation cycle for concentrating O_2 was patented by Charles Skarstrom (U.S. Patent 2,944,627, 1960). Skarstrom cycle consists of (1) co-current bed pressurization using air feed, (2) co-current production of O_2-enriched air by adsorption, (3) counter-current bed depressurization or blow down, and (4) counter-current bed purging using part of the O_2 product (Skarstrom 1966). An alternative would be replacing the feed pressurization by product pressurization because under complete cleanup conditions, the latter leads to higher O_2 recovery than the former (Knaebel and Hill 1983).

Pressure equalization steps introduced for a minimum two-bed system, besides saving the mechanical energy, can salvage part of the product-enriched void gas thus giving higher O_2 recovery (Mendes et al. 2001). On the other hand, if higher production rate is desired, one may truncate the cycle time by combining process steps such as simultaneous pressurization + adsorption and simultaneous depressurization +

purging as demonstrated in the novel design by Sircar and Hanley (1995). Therefore, optimum process cycle can be tailored, by considering adsorbent properties and the effects of operating conditions (Sircar et al. 1998).

2.1.8 Performance Variables and Operating Conditions

PSA performance can be measured in terms of O_2 purity, O_2 productivity, O_2 recovery and bed size factor (BSF), as defined below. Operating conditions such as adsorption pressure, feed flow rate, purge to feed ratio and step times are the determining factors for performance. Generally, O_2 purity and recovery increase with purge quantity and adsorption pressure. However, beyond certain threshold values, purity and recovery deteriorate (Mendes et al. 2001; Lee et al. 2007), because more O_2 is vented out by extended purging, and more O_2 adsorbs onto zeolite at elevated pressure (Sircar et al. 1998). Therefore, it is preferably to keep the pressure between 1.7 – 2.3 atm (Skarstrom 1966; Yuwen et al. 2005). Higher O_2 purity is also attainable at low feed flow rate but at the price of reduced production rate (Mendes et al. 2001; Lee et al. 2007; Yuwen et al. 2005). Even so, maximum O_2 purity achievable using zeolite is 95 mole % because argon, having physical properties similar to those of O_2, remains in O_2 enriched gas as impurity (Rege and Yang 2000).

O_2 purity (mole %) =

$$\frac{moles\ \ of\ \ O_2\ \ in\ \ product}{total\ \ moles\ \ of\ \ O_2,\ N_2\ \ and\ \ Ar\ \ in\ \ product}\text{[redacted]}00\% \tag{2.7}$$

O_2 productivity (mole O_2/g adsorbent/hr) =

$$\frac{net\ \ moles\ \ of\ \ O_2\ \ produced\ in\ \ one\ \ cycle}{mass\ \ amount\ \ of\ \ adsorbent\text{[redacted]}ycle\ \ time} \tag{2.8}$$

O_2 recovery (%) =

$$\frac{moles\ \ of\ \ O_2\ \ in\ \ adsorption\ \ step\text{[redacted]}noles\ \ of\ \ O_2\ \ in\ \ purge\ \ step}{moles\ \ of\ \ O_2\ \ in\ \ feed\ \ step}\text{[redacted]}00\% \tag{2.9}$$

Bed size factor (lbs adsorbent/TPD_c) =

$$\frac{lbs\, of\ adsorbent}{Tons\, of\ O_2\ \ in\, net\ product/cycle}\text{[redacted]}\frac{day}{cycle} \tag{2.10}$$

2.2 Conceptual Designs of Adsorbent Packed Beds

The objective of this section is to evaluate the design alternatives of adsorbent packed bed in an oxygen concentrator using PSA concept.

2.2.1 Design Methodology

A tentative flow diagram incorporating the preliminary understanding of zeolite-based air separation by adsorption from Section 2.1 into packed bed and system design is depicted in Figure 2.9 and explained below:

1. Preliminary assessment on the practicality of miniature scale PSA for producing medical grade oxygen from experimental design and testing. If it is not feasible, change the research topic; otherwise, proceed with the project.
2. Theoretical study of the adsorption equilibrium and kinetics to determine working capacity, pressure envelop and rate of diffusion.
3. Theoretical study of the effects of physical design parameters (e.g. particle size, bed length and bed radius).
4. Theoretical study of the effects of operational design parameters (e.g. step times, flow rates, adsorption pressure).
5. Relate the physical and operational parameters to pressure drop across the bed.
6. Decision to be made:
    i. If it is not for a new PSA bed, remove whatever instrumental, physical and operational constraints (e.g. glass wool as gas flow distributor should not be an inhibitor for bed exhaustion; dead volume

minimization in process lines and connectors should not impose gas flow resistance and limit gas flow rate; orifices of solenoid valves should be free of obstacles).

   ii. If it is for a new PSA bed, mechanically fabricate the design piece considering all operational effects and based on the suitable physical parameters from the study.

7. Assimilate the decision factors from studies in steps 2, 3, 4 and 5 into process cycle, wherever applicable.
8. Test the design piece experimentally. Characterize its operational conditions and system performance.
9. Decision to be made:
   i. If the existing bed can be improved further, go back to step 2.
   ii. If the existing bed cannot be further improved, conceptually design a new miniature PSA bed that can overcome the existing problems. Then go back to step 2.
   iii. If the results are satisfying, conclude the research work.

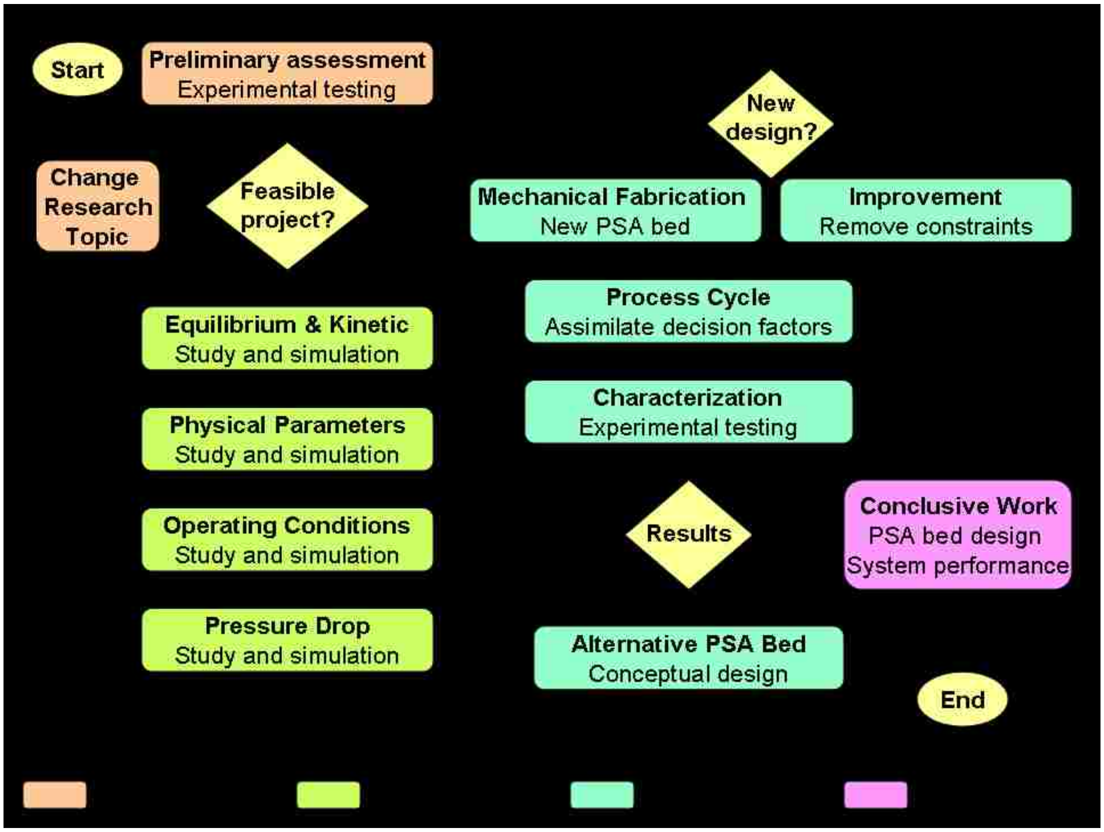


Figure 2.9: Tentative work flow for packed bed and system design.

2.2.2 Physical and Operational Criteria for Improved Design

From the literature reviews and theoretical studies in Section 2.1, it is clear that many operational and physical factors contribute to the separation performance of PSA. Since the targeted outcome of this research is a miniaturized oxygen concentrator, in which PSA bed is the core component, an adequate physical design of adsorber in small volume and light weight is our priority. Operating parameters are considered during the design procedure but will be optimized later during experimental testing.

Listed below are the overall physical and operational design criteria for an improved miniature PSA system, which will be considered for the design alternatives and for improving system in the future:

Physical parameters:

1. Using smaller particle size
   - to improve diffusion kinetic
   - to enhance full utilization of adsorbent
   - to avoid much product lost from the void
   - to avoid early breakthrough due to micro-channeling
2. Using small bed
   - to minimize heat effect
   - to minimize dead volume due to better gas flow distribution
3. Improving bed geometry: large flow area and short axial length
   - to reduce pressure drop problem
4. Removing instrumental constraint such as enlarging exhaust orifice
   - to allow fast and complete depressurization
5. Simplifying piping and instrumentation
6. Robustness – no moving parts
7. Reducing footprint
8. Easy to fabricate
9. Scalable and flexible

Operational parameters:

1. Reducing cycle time

   - to increase productivity
   - to reduce temperature excursion by quick reuse of conserved heat

2. Increasing flow rate marginally

   - to increase product throughput but do not inflict high pressure drop

3. Lowering adsorption pressure

   - to reduce compression power
   - to reduce pressurization time
   - to achieve most beneficial working range

4. Complete depressurization

   - to improve desorption efficiency
   - to reduce purge/feed ratio

5. Using product pressurization step

   - to increase oxygen recovery

6. Adopting five-step PSA cycle used for air separation (Leavitt 1991):

   - to increase oxygen recovery by collecting part of the depressurized gas

     i. Co-current feed pressurization

     ii. Co-current adsorption at high pressure

     iii. Co-current depressurization

     iv. Counter-current blow down

     v. Counter-current purge at low pressure

2.2.3 Packed Bed Design Alternatives

In designing a miniature PSA bed, it is desired to process large amount of air at high speed in order to produce large O_2 product throughput. To achieve this objective, one may consider using very small particle size, introducing high input flow rate and speeding up process steps. However, these operating conditions bring forth severe pressure drop problem, which greatly reduces separation efficiency of PSA.

Presented below are parallel short beds design and radial bed design that address pressure drop problem, in addition to meeting part or most of the design criteria mentioned in Section 2.2.2. The operational issues, advantages and disadvantages of each design will be discussed briefly. Following this, a small radial PSA bed with temperature control system is proposed. The temperature control is meant for increasing equilibrium capacity by exploiting the improved heat transfer rate in small bed through larger planar surface.

2.2.3.1 Parallel Short Beds

Parallel short beds design is considered because it significantly lowers the pressure drop and allows higher introduction of process fluid. Series and parallel electric circuits can be used to explain the advantages of using multiple parallel beds, which is represented by the parallel circuit, rather than a single long bed, represented by the series circuit. In a series circuit, the voltage of the source (represents total pressure drop) is the sum of the voltages across each of the components (represents individual pressure drop), while the current (represents air or purge flow rate) through all of the components is equal. In a parallel circuit, the voltage across each of the components is equal while the total current is the sum of the currents through the components.

Figure 2.10 (a) and (b) show series and parallel circuits with 3 resistors of 30 Ω each. If a total current, I_T, of 1/6 A is to be flowed through the series and parallel circuits individually, the required voltage source is 15 V for series circuit but only 1.67 V for parallel circuit. Analogically, to process the same amount of process fluid, series beds (or a single long bed) incur much higher pressure drop than the parallel beds because in parallel configuration heavy flow rate is split and processed by separate beds thus flow resistance is greatly reduced.

Figure 2.10 (c) and (d) show series and parallel circuits with 3 resistors of 30 Ω each and are subjected to the same amount of voltage source. With voltage source of 15 V, series circuit can only give rise to 1/6A total current, but parallel circuit can yield 1.5A total current. Analogically, when subjected to the same amount of total pressure drop, series beds (or a single long bed) can only process low flow rate but parallel beds can process much higher flow rate thus giving higher productivity.

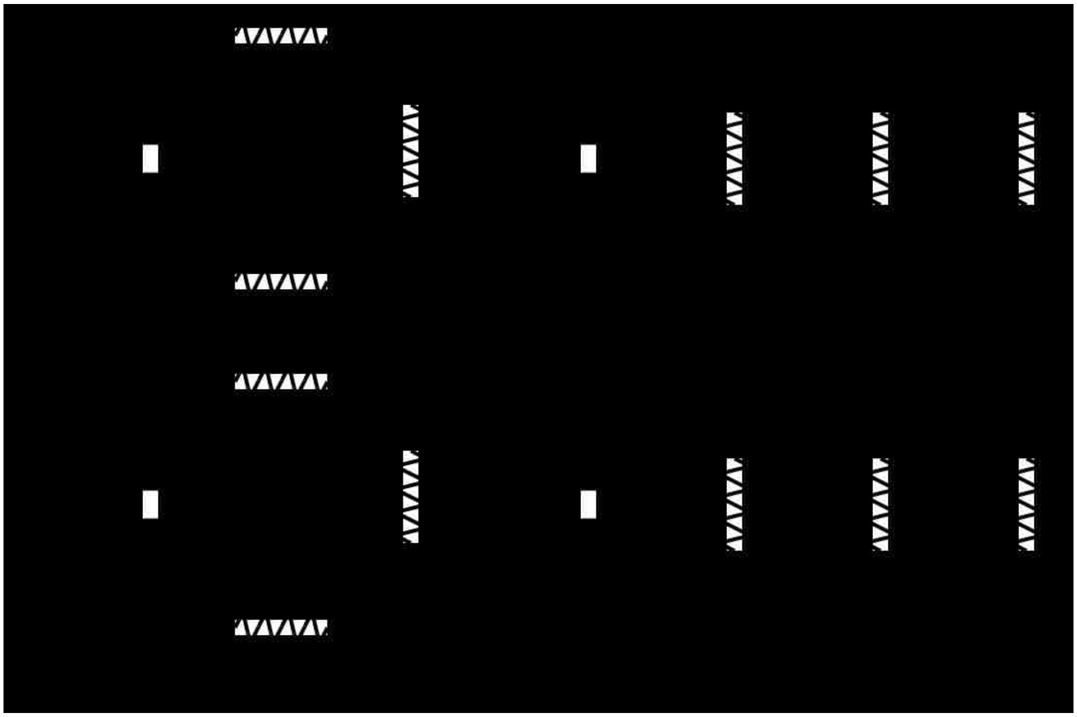


Figure 2.10: Series and parallel electrical circuits.

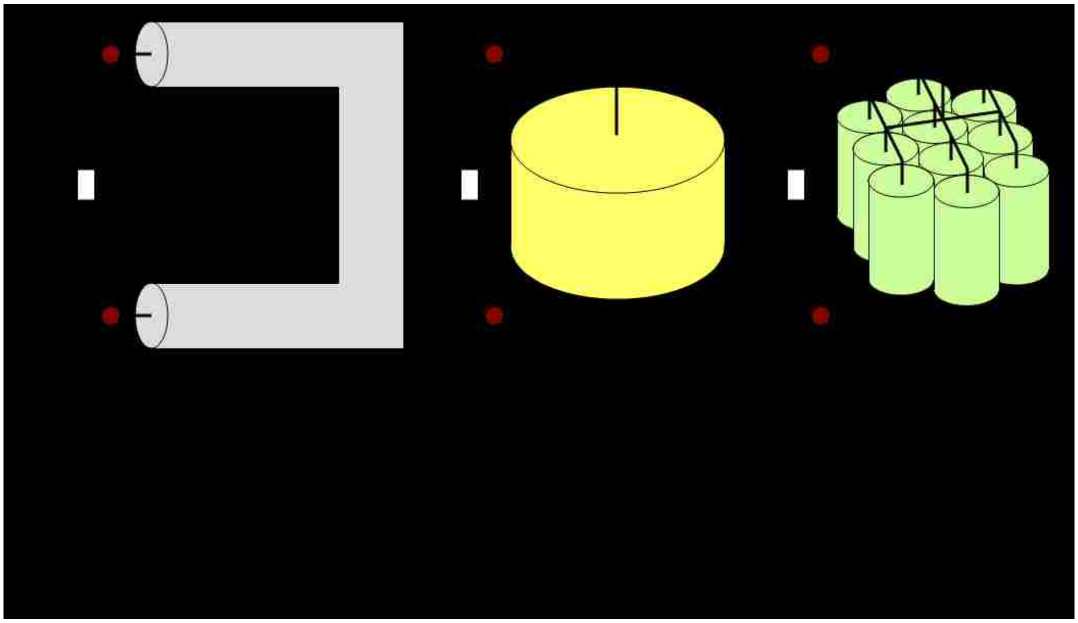


Figure 2.11: (a) Single long bed, (b) single short and big bed and (c) parallel short beds, all have same total volume.

Figure 2.11 shows (a) a single long bed, (b) a single short and big bed, and (c) 10 parallel shorts beds, each has the same total volume and contains the same amount of adsorbent with the same particle size and bed porosity. They are also subjected to same total air flow rate. Using Ergun equation, total pressure drop across the bed(s) for each orientation is calculated and given in Figure 2.11. We see that the single long bed in (a) produces highest pressure drop, while short beds in (b) and (c) produces same amount and low pressure drop. The reason that single bed in (b) yielded the same pressure drop as parallel beds in (c) is because both configurations are characterized by the same cross sectional flow area and same length. One may inquire why not using a single short bed with large cross sectional area design, the justification is small bed may give better separation efficiency due to improved gas flow distribution and lesser dead volume thus causing more complete utilization of the adsorbent.

A design of parallel short beds and the fabrication steps for feed/exhaust and product/purge ports with big orifices are presented in Figure 2.12. This design can be made scalable by topping layers of short packed beds instead of enlarging the perimeter of cylindrical monolith so that interconnectors at both ends can be used for all scale-ups. Glass wool cover acts as hydraulic impedance that ballasts the flow into packed array. The overall merits of this design are given as follows:

1. Low pressure drop allows the use of smaller particle size. This will promote intra-pellet diffusion and give rise to dense packing that eliminates micro-channeling problem.

2. Low pressure drop allows introduction of higher flow rate thus raising productivity.

3. Narrowed straight tube facilitates uniform gas flow distribution and improves adsorbent utilization.

4. Big orifice eliminates flow obstacle during depressurization.

5. Inlet and outlet ends in axial direction simplify the piping requirement.

However, this design with multiple beds in a monolith may require more efforts in fabrication.

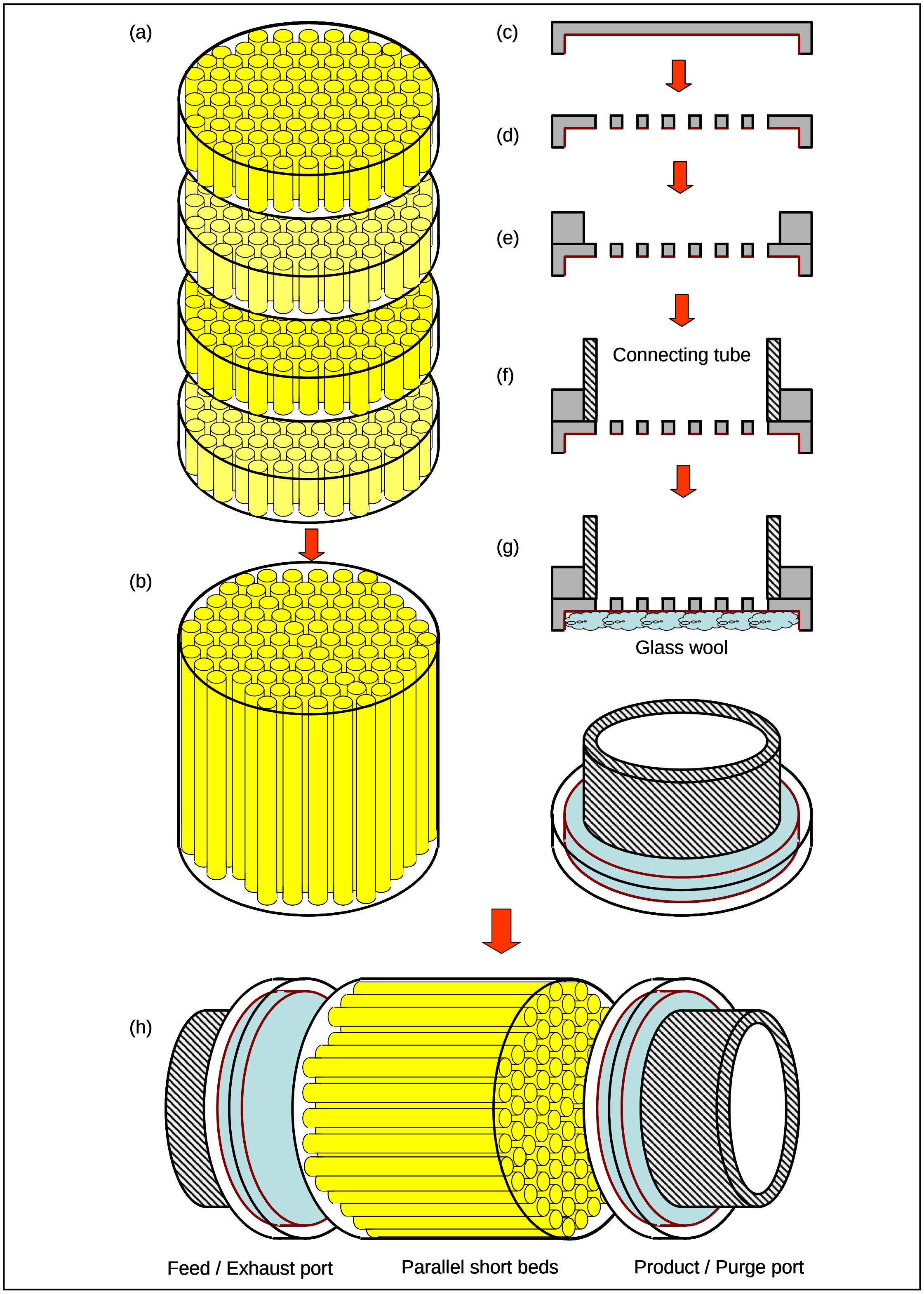


Figure 2.12: Parallel short beds and fabrication of inlet and outlet ends.

2.2.3.2 Radial Bed

Conventional PSA generally has an axial flow configuration characterized by ratio of bed length to bed diameter, $L/D > 1$ and $L/D < 1$, for vertical and horizontal packed beds, respectively (Sircar 2003). Changing the packed column configuration to a radial flow geometry had been strongly advocated by Rota and Wankat (1991) because besides giving comparable performance to that of axial bed (if equilibrium theory prevails), it also offers extra benefits of large cross sectional area, small pressure drop and ease of scaling up. The features and advantages of radial flow PSA bed are detailed below:

1. Adsorbent is contained in annular layer, feed air flows radially from outside toward inner cylinder. The reason for this flow direction is to reduce velocity variation (Chiang and Hong 1995; Huang and Chou 2003). As nitrogen is adsorbed along flow direction, gas velocity reduces, but the continual decrease of cross sectional area minimizes velocity variation. The same system showed virtually no separation effect when the feed direction was reversed (Chiang and Hong 1995).
2. Increased interstitial flow velocity toward the center sharpens the concentration wavefront, thus promoting deeper feed penetration and resulting in higher adsorbent utilization (Chiang and Hong 1995).
3. Purge gas flows radially from inner toward outside cylinder. The large exposure of the outside cylinder to low pressure promotes depressurization and

desorption (Chiang and Hong 1995). A depiction of the radial bed with the feed and purge flow directions is given in Figure 2.13.

4. For the same feed pressure and same amount of adsorbent, the separation performance of radial flow PSA is better than that of the axial flow PSA by utilizing smaller particle size. Particles as small as a few µm could be used directly due to the large cross sectional area that lowers the pressure drop. Smaller particle size facilitates faster adsorption kinetics and enables rapid PSA (Chiang and Hong 1995; Huang and Chou 2003).

5. For the same volume of packing, a radial bed would give a smaller total pressure drop due to larger cross sectional area (Chiang and Hong 1995). This is demonstrated by the pressure drop calculations using Ergun equation for axial beds in Figure 2.11 and for radial beds in Figure 2.14. With the same packing volume of 12.566 cm^3 of the same particle size and bed porosity, and subjected to same total air flow rate, axial short and long beds give 0.1711 and 17.7 psi pressure drop respectively, while radial thick and planar beds give 0.0034 and 0.0068 psi pressure drop respectively. Note that the pressure drop in radial bed is at least two order of magnitude smaller than that in axial bed, thus justifying the use of very small particle size and/or introduction of very high flow rate.

6. While thicker radial bed is better in term of lower pressure drop because it offers larger cross sectional flow area, planar radial bed is better in term of higher heat transfer rate because it offers larger planar surface area. Both

features are important for radial bed because it may be used to process large flow rate that brings up high pressure drop and heat excursion problems.

Radial bed design is attractive but fabrication is very complex. The inner and outer walls of the annular vessel that contains micro-size adsorbent particle must be very finely perforated to prevent solid overspill but allows high flow rate of gas permeation.

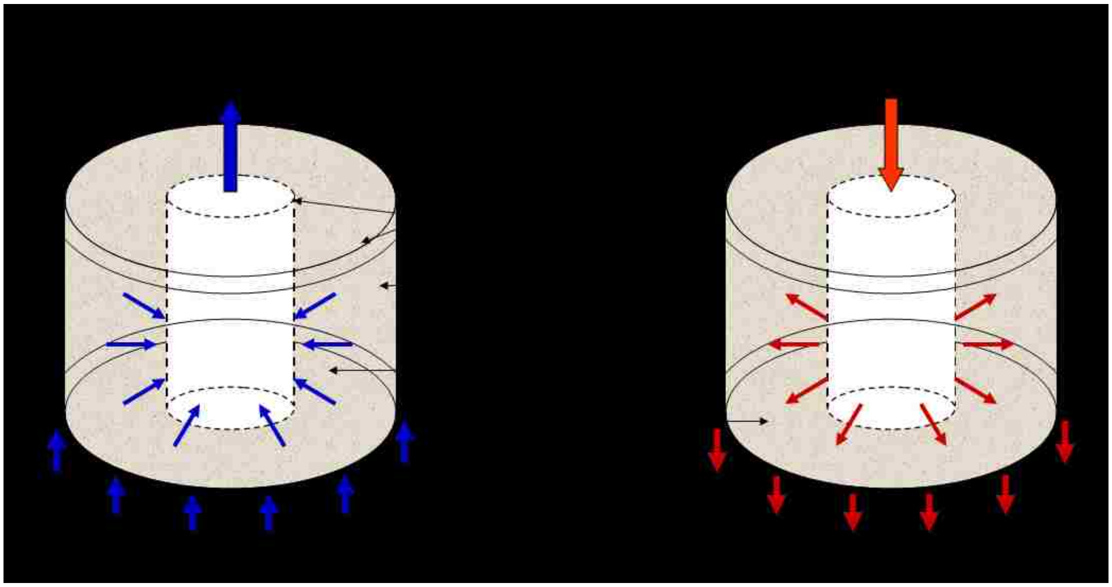


Figure 2.13: Radial bed and flow directions.

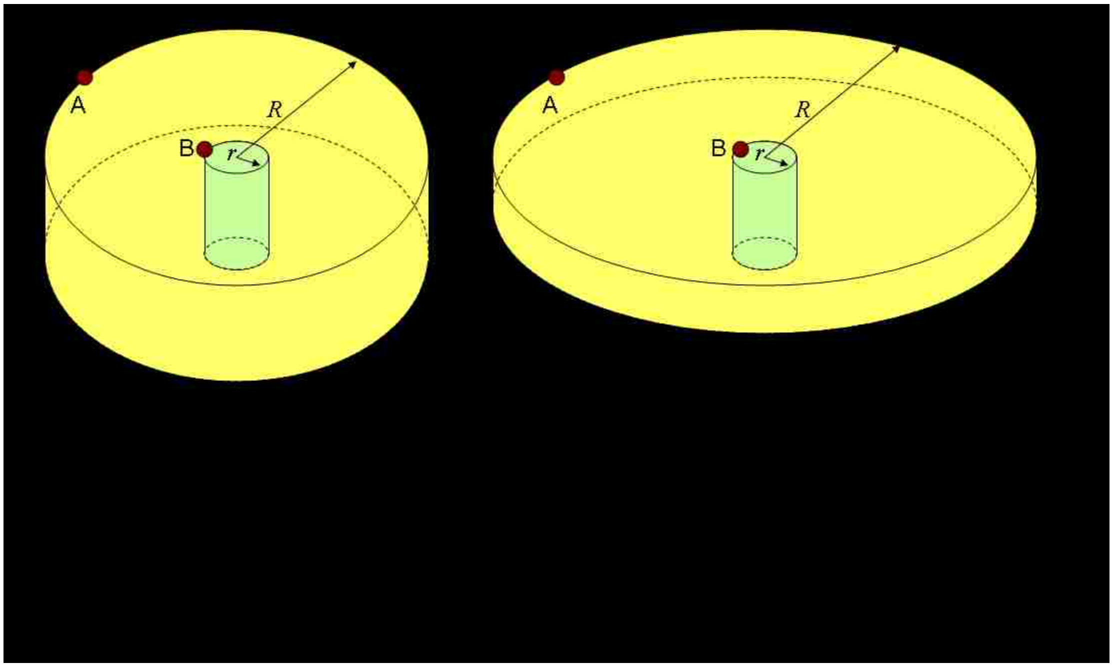


Figure 2.14: Pressure drop across the flow direction in (a) thick and (a) planar radial beds.

2.2.3.3 Planar Radial PSA Bed with Temperature Control System

Generally TSA (Temperature Swing Adsorption) process is used for trace removal purification, in which preheated purge gas is used to raise the bed temperature so that trace solutes on the adsorbent are desorbed easily. TSA is not used for bulk separation in industry such as O_2 concentration due to large amount of heat required to desorb bulk N_2 gas from the adsorbent. Moreover, heating is slow and often a rate determining step.

Due to the reasons above, O_2 concentration is generally carried out using PSA. However, when big PSA bed is introduced with large feed flow rate, adiabatic and non-isothermal conditions occur and system performance is affected. On the contrary, small PSA bed that handles small feed flow rate is near isothermal due to large surface to volume ratio and relatively small amount of heats of adsorption and desorption. For example, O_2 concentration in PSA systems with packed bed volumes of 1088 cm^3 (Teague and Edgar 1999) and 982 cm^3 (Jee et al. 2001) that were each fed by 48.65 L_{STP}/min and 9.5 L_{STP}/min air, experienced 8°C and 1.3°C temperature excursions at the center of the unit, respectively, as shown in Figures 2.15 and 2.16. The purge gas requirements predicted by the isothermal model are much lower than those predicted by the adiabatic model because bed temperature in the isothermal model is higher therefore it is easier to clean the isothermal bed than the adiabatic (Sircar and Kumar 1985). Many researchers and practitioners are satisfied with the thermal conditions in small bed and no further thermal treatment is applied.

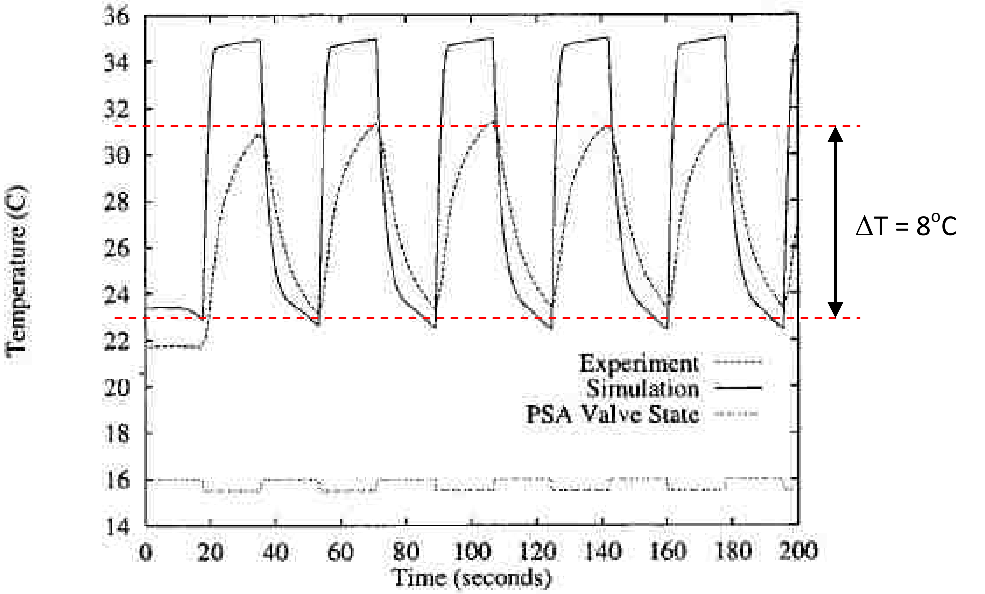


Figure 2.15: Dynamic bed temperature at the center of packed bed with 1088 cm^3 volume, 48.65 L_{STP}/min feed and 4.69 bars feed pressure (Teague and Edgar 1999).

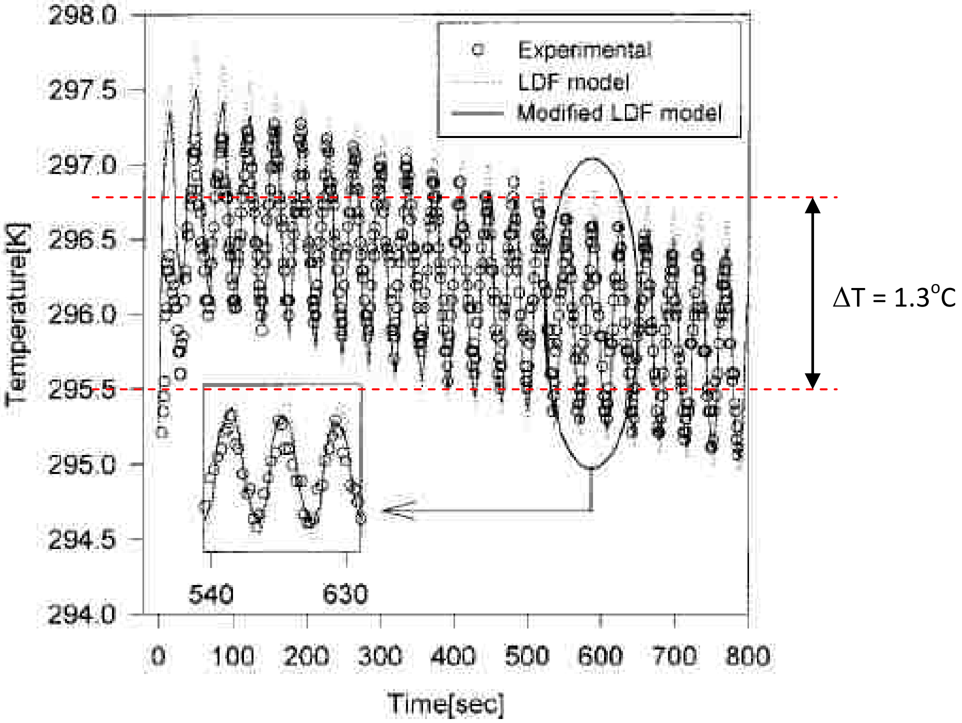


Figure 2.16: Dynamic bed temperature at the center of packed bed with 982 cm^3 volume, 9.5 L_{STP}/min feed and 4 atm adsorption pressure (Jee et al. 2001).

In this research, a new idea is proposed for combining small planar radial PSA bed with an integrated temperature control system. This idea is purposed to exploit the improved heat transfer efficiency in small bed and higher heat flux through the large surface area offered by planar geometry. The introduced temperature control system is not for maintaining a constant bed temperature throughout the process, rather it is for generating a sinusoidal-like temperature profile at appropriate magnitude in pace with the adsorption and desorption steps such as that in Figure 2.17. Since adsorption loading is a function of pressure (P) and temperature (T), changing the P-T profiles in such way may increase equilibrium capacity. Specifically, high pressure adsorption at lower temperature and low pressure desorption at higher temperature contribute to increased capacity. This is depicted by the adsorption isotherms at different temperatures in Figure 2.18 (Park et al. 2006) and the TSA and PSA isotherms in Figure 2.19. An example of temperature controlled profile in a small planar radial bed looks like that in Figure 2.20; it is compared against the temperature profiles of adiabatic large bed and near isothermal small bed. Note that temperature controlled profile is in the opposite trend to those without control. Besides changing the equilibrium property, a direct and positive implication of this temperature change are feed volume reduction during adsorption step and purge volume increment during desorption step, which is equivalent to high purge/feed volume ratio but at lower molar expense of purge gas.

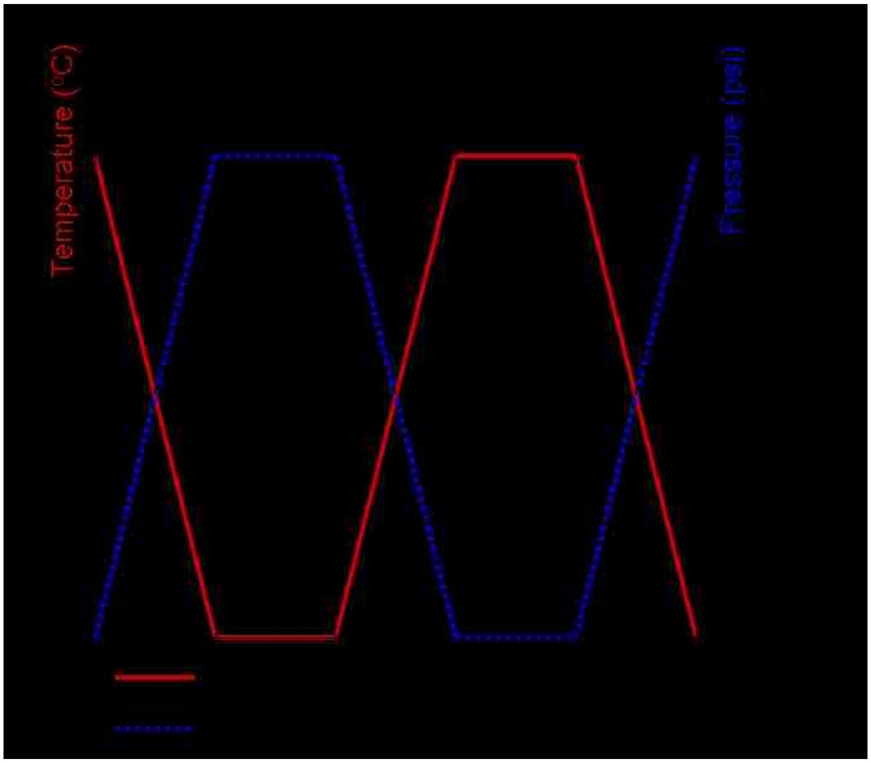


Figure 2.17: Design temperature and pressure profiles for adsorption and desorption to increase equilibrium capacity.

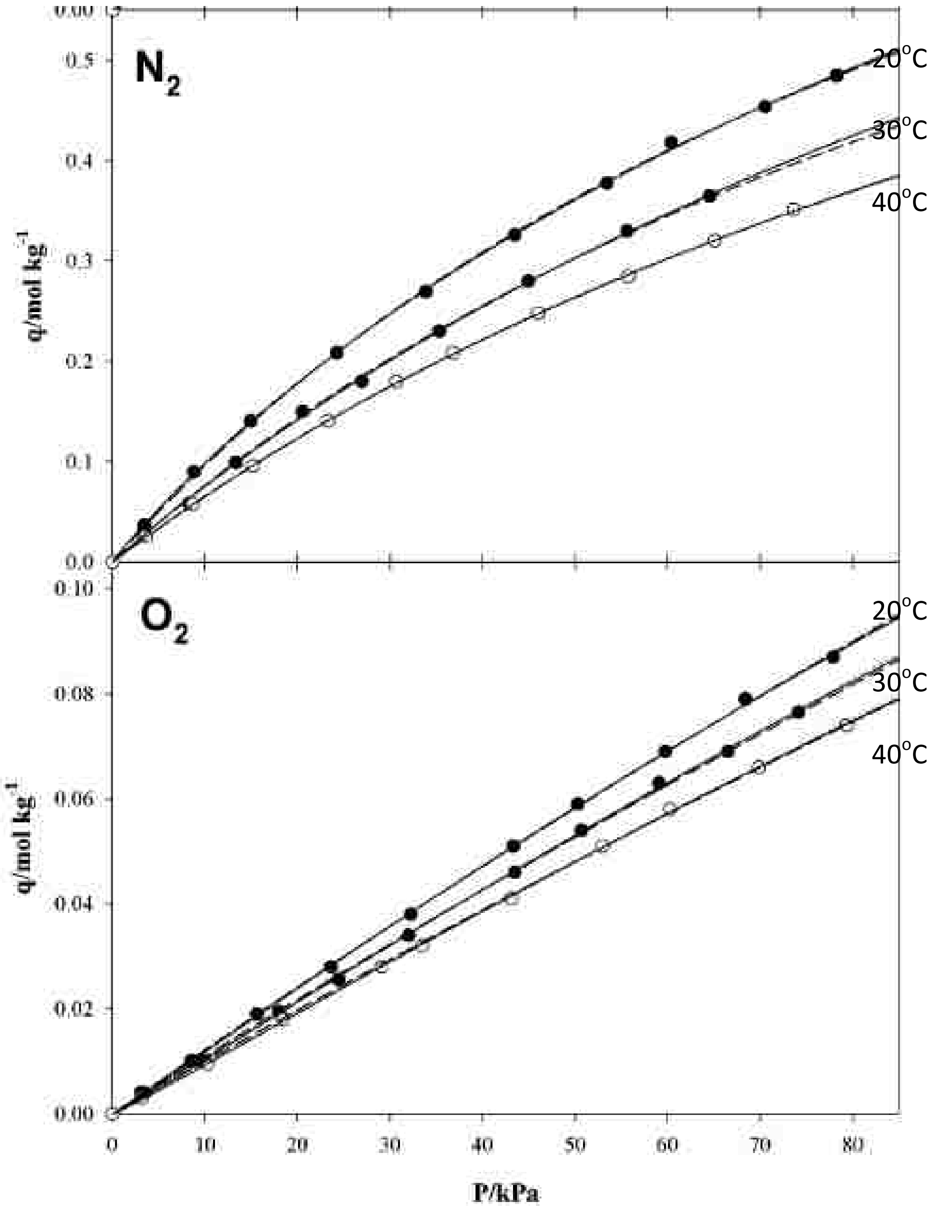


Figure 2.18: Adsorption isotherms of nitrogen and oxygen on LiX at different temperatures following ___ Sips; - - - Langmuir; . . . Toth equations (Park et al. 2006).

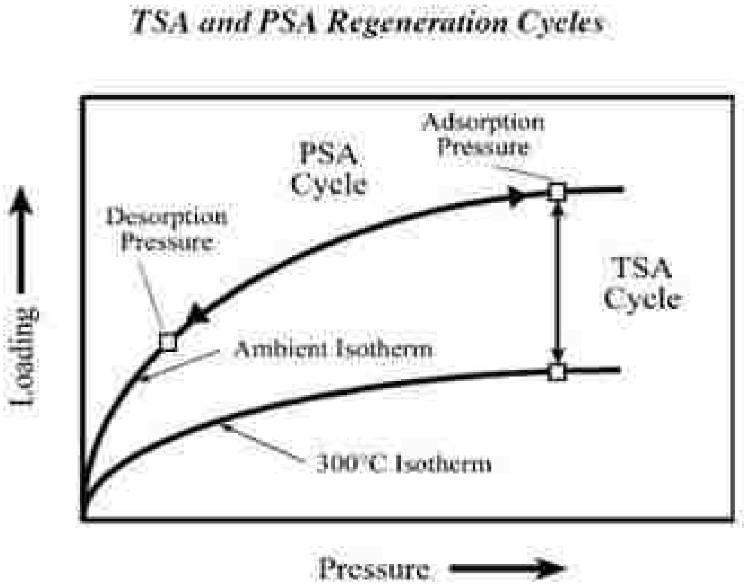


Figure 2.19: PSA and TSA regeneration.

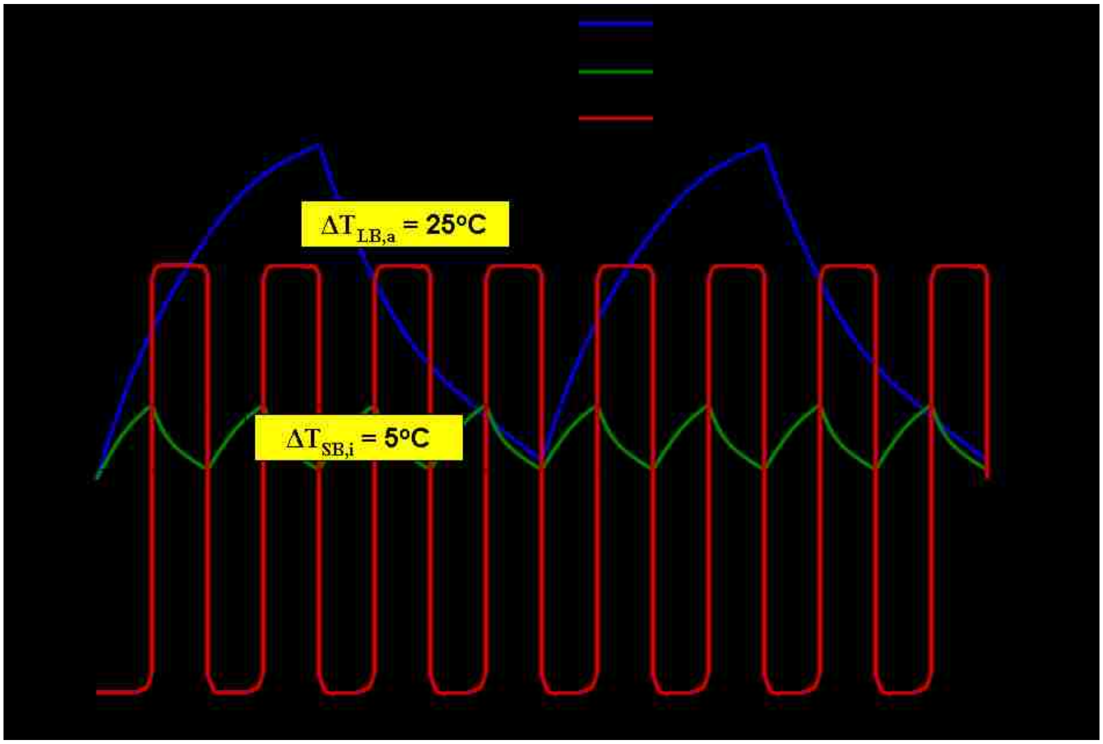


Figure 2.20: Example of temperature profiles for adiabatic large bed, near isothermal small bed, and temperature controlled small planar radial bed.

Resistive cooler/heater can be integrated onto the planar surface of radial bed to cool and heat the bed for adsorption and desorption, respectively. Heater is not used to preheat the purge gas directly but is applied outside the planar bed for safety reason, in addition to accomplishing balanced thermal cycle because cooler is also applied externally. Indeed, the heater and cooler is one device, see Figure 2.21, with its temperature ranges between the upper and lower process temperatures. Despite the possible heat effect complication such as temperature overshoot, relatively slow ramping and decaying, it is designed that additional cooling and heating will increase the adsorbent capacity thus adsorbent amount can be reduced.

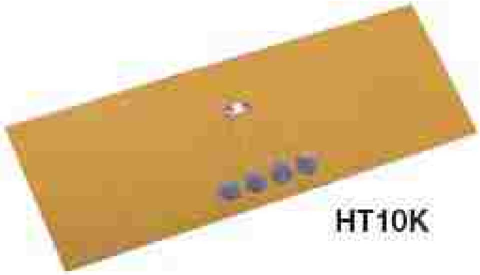


Figure 2.21: Foil Heating Element from Thorlabs.com. The coverage size is 1" x 3" (25.4mm x 76.2mm) and its temperature range is -32 to 100°C.

2.3 Evaluate the Potential Use of Cobalt Complexes for Air Separation

A U.S. Patent with Application Publication No. 2009/0293879 entitled "Device and Method for Producing Oxygen" by inventor Oskar Franberg presented a seemingly attractive alternative to use Cobalt complexes (Salcomine, Fluomine) for adsorbing O_2 instead of using zeolitic molecular sieve for adsorbing N_2 from the air. Its argument is based on the fact that O_2 only constitutes 1/5 of air, thus only 1/5 of space is needed for adsorption, which may result in lighter portable oxygen concentrator.

However, lighter oxygen concentrator must be a combination of good production capacity and high production rate of the adsorption on the sorbent (Salcomine or Fluomine) so that bed size factor (BSF) is reasonably small, in addition to simple operating conditions so that heavy equipment like vacuum pump and heat exchanger are not required for the oxygen concentrating process. The author of this patent did not give definite operating conditions for this process. He merely described that charging phase (cobalt complex adsorbs O_2) can be carried out at higher pressure and lower temperature, while discharging phase (cobalt complex desorbs O_2 as a product) at lower pressure and higher temperature. There was no mention about how high/low the temperature and pressure should be, which are the main contributors to high energy cost and increased complexity of the process. Performance results were not reported in this invention.

One of the earliest applications of Salcomine for producing cutting and welding O_2 on shipboards was reported by Fogler (1947). The performance of the pilot production unit was tested and given as follows. For adsorption step: air pressure at 90 psig (7.12 abs. atm.), cooling water at 50 ^{o}F (10 ^{o}C), and adsorption time of 8.7 min yielded ~30 ft^3 O_2/120 lb Salcomine/cycle (equivalent to ~0.696 mmol O_2/g Salcomine/cycle) of 96 – 98 % O_2. For desorption step: evacuation to 26 inches Hg (~ 0.869 abs. atm.), steam at 5 psig, and desorption time of 2 – 3 min was required. With the average production capacity 260 ft^3/hr of 98 – 99% O_2 using 360 lbs of Salcomine, bed size factor (BSF) for this pilot unit is calculated to be ~1314.7 lb/TPD$_c$. The average operation life time of Salcomine in this unit was ~1000 hours. Claimed to be uneconomical under most conditions, this O_2 generating unit was installed on ships stationed in the North Atlantic so that low temperature sea water would be available for cooling.

Indeed, using Salcomine and Fluomine to extract O_2 is not a new idea in air separation. For example, U.S. Patent 5,071,449 (1991) by Shivaji Sircar mentioned the use of these chemicals to get high concentration (95 – 99.5%) N_2 product directly via his invention of single separation vessel having multiple adsorption layers. Note that using these O_2-binding agents is more reasonable in this invention because the desired product is N_2, not O_2.

The paper by Hutson and Yang 2000 reported the characterized properties of O_2-binding Cobalt complexes and presented many complexities of using these chemicals for air separation. Some statements from this paper are directly quoted as follows:

> "None of these materials (cobalt complexes) has shown the necessary combination of reversibility, capacity, and stability needed for use in industrial

> gas separations." (Chen and Martell 1987; Chen and Martell 1989; Dzugan and Busch 1990; Ramprasad et al. 1995)

> "None of the efforts for protecting the oxygen-binding complex from oxidation and dimerization were proved effective for separating oxygen from nitrogen due to instabilities and/or inadequate O_2-binding capacity." (Lunsford 1975; Howe and Lunsford 1975; Imamura and Lunsford 1985; Herron 1986; Drago et al. 1988; Taylor et al. 1989; Taylor et al. 1992)

> "All these materials (the reported transition metal complexes with oxygen-binding ability) have suffered from one or more of the following drawbacks that have prevented commercialization: (1) chemical instability, (2) unacceptable adsorption characteristics, and/or (3) unacceptable cost." (Hutson and Yang 2000)

> "The most prohibitive problem facing the commercialization of Co(salen), Co(fluomine), and other O_2-binding complexes has been the chemical instability due to autoxidation. Three general mechanisms may be involved in the autoxidation of the O_2 carrying complexes: (i) dimerization of the complex, (ii) irreversible oxidation of the ligand, and/or (iii) irreversible oxidation of the central coordinating Cobalt atom." (Li and Govind 1994)

Several important technical barriers inhibiting the use of Cobalt complexes for production of O_2 enriched air are illustrated below by taking the sample characteristics reported in the paper of Hutson and Yang (2000). Co(salen) (also called Salcomine) has adequate equilibrium capacity of 1.06 mmol O_2/g at 25^{o}C and 1 atm O_2 partial pressure (equivalent to air pressure of ~4.762 abs. atm). However, its reversibility is only permissible at very deep vacuum level, it is also quickly deactivated by the presence of moisture. Co(fluomine) (also called Fluomine) came in as substitute, whose use is said to have lesser drawbacks than those of Co(salen). Despite its adequate equilibrium capacity of 1.13 mmol O_2/g at 25^{o}C and 1 atm O_2 partial pressure (equivalent to air pressure of ~4.762 abs. atm.), Co(fluomine) demonstrates irreversibility problem. Desorption of O_2 from this compound must be carried out at high temperature and at very deep vacuum, yet the desorption is incomplete. Moreover, its application was also hampered by long-term chemical instability. Therefore, varied Cobalt complexes were synthesized to improve O_2 sorption characteristics. Co(fluomine) was immbolized on three different nanoporous substrates, namely, LSX, MCM-41, and ion-exchange resin (IXR). Co(fluomine)-MCM-41, claimed by Hutson and Yang as the especially promising O_2-binding agent due to its reversibility and positive slope, yielding better O_2 working capacity among all others. Other chemicals do not even show reversible O_2 sorption, it may be possible to desorb O_2 from some of them at a very low but not practical vacuum level (say << 10 torr). Nonetheless, practical oxygen working capacity of Co(fluomine)-MCM-41 is still very low for producing medical grade O_2. For example a VSA working capacity for Co(fluomine)-MCM-41 operating between ambient pressure (O_2 partial pressure = 0.21 atm) and a vacuum of ~ 38 torr O_2 partial pressure is only ~ 0.016 mmoles/g, which is miniscule compared to the N_2 working capacity exhibited by a zeolite PSA process working between pressures of say 4 – 1 atm. The key point is not that the more dilute component of the feed air is removed but what is the working capacity for removal of that component? That will decide the BSF.

The low O_2 working capacity makes the un-separated air-like void gas predominant during desorption of O_2 enriched product gas. For example, the amount of void in the column (assumed bulk density ~ 0.64 g/cc and helium void fraction ~

0.7) at a pressure of 1 atm and a temperature of 25^{o}C is ~0.045 mmoles (21 % O_2 + 79 % N_2)/g. Thus the total desorbed gas will have an O_2 concentration of ~42 % due to dilution by void N_2. Hence, the material is not even suitable to make medical concentrator grade O_2.

Since a VSA will provide a much higher product O_2 concentration than a PSA (higher void to O_2 working capacity ratio) one will need a fairly large vacuum pump to use this material.

Adsorption and desorption kinetics is another important issue. For example, the rates of O_2 adsorption and desorption onto/from Co(salen) for thermal swing adsorption process for production of O_2-enriched combustion air in Park et al. (1991) were approximated to be 3 min (yielded O_2 capacity ~1.81 g of O_2/g of Co(salen)) and 4 min (yielded O_2 capacity ~1.83 g of O_2/g of Co(salen)), respectively, while zeolitic adsorbent only requires seconds to achieve complete adsorption and desorption (Sircar 1995). Fast adsorption and desorption kinetics is much desirable because it can lower the bed size factor tremendously due to more frequent usage of the bed inventory.

The water tolerance of these Cobalt materials is also unknown.

Cobalt Complexes is Not Appealing for Miniature O_2 Concentrators

From the above analysis, it is likely that much research and development efforts is still needed to come up with Cobalt complexes that have good combination of reversibility, stability, equilibrium working capacity, as well as adsorption and desorption kinetics. Further development work on Salcomine or Fluomine after this published work by Hutson and Yang (2000) are not available or not accessible. It is concluded that Cobalt compounds are far from being better adsorbent than zeolitic molecular sieve for concentrating O_2 from air. The following are the key points summarizing why Co complexes are not yet suitable for use in miniature O_2 concentrators:

1. Limited equilibrium capacity;
2. Dilution of oxygen product from the void gas due to limited equilibrium capacity;
3. Poor reversibility of oxygen isotherm;
4. Slow adsorption and desorption kinetics;
5. Short term stability and permanent degradation (oxidation) of Cobalt complex;
6. Need to use vacuum pumps, as compared to simpler compressors;
7. Need to use elevated temperature for desorption;
8. Costly operation;
9. Water tolerance of these materials is unknown.

2.4 References

Adsorption Engineering. Edited by Suzuki, Motoyuki. Chemical Engineering Monographs 25, (1989).

Adsorbents: Fundamentals and Applications. Edited by Yang, R.T. John Wiley & Sons, Inc. (2003).

Breck, D.W.: Zeolite Molecular Sieves. R. E. Krieger Publishing, Malabar, FL (1984).

Chen, D., Martell, A.E.: Dioxygen Affinities of Synthetic Cobalt Schiff Base Complexes. *Inorg. Chem.*, **26**, 1026 (1987).

Chen, D., Martell, A.E., Sun, Y.: New Synthetic Cobalt Schiff Base Complexes as Oxygen Carriers. *Inorg. Chem.*, **28**, 2647 (1989).

Chiang, A.S.T., Hong, M.C.: Radial Flow Rapid Swing Adsorption. *Adsorption*, **1**, 153-164 (1995).

Doong, S.J., Yang, R.T.: Role of Pressure Drop in Pressure Swing Adsorption. *AIChE Symposium Series*, **84**, 145-154 (1988).

Drago, R.S., Bresinska, I., George, J.E., Balkus, K., Taylor, R.J.: Entrapment of an Anionic Stable Moisture Resistant Oxygen Carrier in Zeolite Y. *J. Am. Chem. Soc.*, **110**, 304 (1988).

Du, X.M., Wu, E.D.: Porosity of Microporous Zeolites A, X and ZSM-5 Studied by Small Angle X-ray Scattering and Nitrogen Adsorption. *J. Phy. Chem. Solids*, **68**, 692-1699 (2007).

Dzugan, S.J., Busch, D.H.: Synthesis and Characterization of New Cobalt Dioxygen Carriers Based on a Familar Macrocyclic Ligand. *Inorg. Chem.*, **29**, 2528 (1990).

Fogler. B.B.: Regenerative Unit for Generating Oxygen. *Ind. Eng. Chem.*, **39** 10, 1353-1360 (1947).

Franberg, O.: Device and Method for Producing Oxygen. U.S. Patent Application Publication 0293879 (2009).

Hees, B., Puppe, L., Reiss, G.: Binder-Free Molecular Sieve Zeolite Granules Which Contain Zeolites of the Type Lithium Zeolite A and Lithium Zeolite X. U.S. Patent 5,962,358 (1999).

Herron, N.: A Cobalt Oxygen Carrier in Zeolite Y. A Molecular “Ship in a Bottle”. *Inorg. Chem.*, **25**, 4714 (1986).

Howe, R.F., Lunsford, J.H.: Electron Paramagnetic Resonance Studies of Some Cobalt Amine Oxygen Adducts in Zeolite Y. *J. Am. Chem. Soc.*, **97**, 18 (1975).

Huang, W.C., Chou, C.T.: Comparison of Radial and Axial Flow Rapid Pressure Swing Adsorption Processes. *Ind. Eng. Chem. Res.*, **42**, 1998-2006 (2003).

Hutson, N.D., Rege, S.U., Yang, R.T.: Mixed Cation Zeolites: LixAgy-X as a Superior Adsorbent for Air Separation. *AIChE J.*, **45**, 724-734 (1999).

Hutson, N.D., Yang, R.T.: Synthesis and Characterization of the Sorption Properties of Oxygen-Binding Cobalt Complexes Immobilized in Nanoporous Materials. *Ind. Eng. Chem. Res.*, **39**, 2252-2259 (2000).

Imamura, S., Lunsford, J.H.: Separation of Oxygen from Air by [CoII(byp)(terpy)]+2 Complex in Zeolite Y. *Langmuir*, **1**, 326 (1985).

Jee, J.G., Lee, J.S., Lee, C.H.: Air Separation by a Small-Scale Two-Bed Medical O_2 Pressure Swing Adsorption. *Ind. Eng. Chem. Res.*, **40**, 3647-3658 (2001).

Knaebel, K.S., Hill, F.B.: Analysis of Gas Purification by Pressure Swing Adsorption: Priming the Parametric Pump. *Sep. Sci. Technol.*, **18**, 1193 (1983).

Leavitt, F.W.: Air Separation Pressure Swing Adsorption Process. U.S. Patent 5,074,892 (1991).

Lee, S.J., Jung, J.H., Moon, J.H., Jee, J.G., Lee, C.H.: Parametric Study of the Three-Bed Pressure-Vacuum Swing Adsorption Process for High Purity O_2 Generation from Ambient Air. *Ind. Eng. Chem. Res.*, **46**, 3720-3728 (2007).

Li, G.Q., Govind, R.: Separation of Oxygen from Air Using Coordination Complexes: A Review. *Ind. Eng. Chem. Res*., **33**, 755 (1994).

Lunsford, J.H.: The Formation, Characterization, and Catalytic Activity of Transition Metal Complexes in Zeolites: The P. H. Emmett Award Address. *Catal. Rev. - Sci. Eng.*, **12**, 2, 137 (1975).

Mendes, A.M.M., Costa, C.A.V., Rodrigues, A.E.: Oxygen Separation from Air by PSA: Modeling and Experimental Results, Part I: Isothermal Operation. *Sep. Pur. Technol.*, **24**, 173-188 (2001).

Park, S., McLarnon, C., Mathur, V.K., Tomellini, S.A., Planalp, R.P.: A Thermal Swing Absorption Process for Production of Oxygen-Enriched Combustion Air. *ACS Division of Fuel Chemistry*, **36**, 3, 1361-1368 (1991).

Park, Y.J., Lee, S.J., Moon, J.H., Choi, D.K., Lee, C.H.: Adsorption Equilibria of O_2, N_2 and Ar on Carbon Molecular Sieve and Zeolites 10X, 13X and LiX. *J. Chem. Eng. Data*, **51**, 1001-1008 (2006).

Ramprasad, D., Pez, G.P., Toby, B.H., Markley, T.J., Pearlstein, R.M.: Solid State Lithium Cyanocobaltates with a High Capacity for Reversible Dioxygen Binding: Synthesis, Reactivity and Structures. *J. Am. Chem. Soc.*, **117**, 10694 (1995).

Rege, S.U., Yang, R.T.: Kinetic Separation of Oxygen and Argon using Molecular Sieve Carbon. *Adsorption*, **6**, 15 (2000).

Rota, R., Wankat, P.C.: Radial Flow Pressure Swing Adsorption. Edited by Meunier, F. and LeVan, M.D. *Proc. Adsop. Proc. for Gas Sep.*, 143-148 (1991).

Sircar, S.: Adsorption Technology for Gas Separation. Edited by Lee, C.H. *Proceedings of the 3rd Pacific Basin Conference on Adsorption Science and Technology, Korea*, 72-78 (2003).

Sircar, S.: Air Fractionation by Adsorption. *Sep. Sci. Tech.*, **23**, 2379 (1988).

Sircar, S.: Gas Separation by Rapid Pressure Swing Adsorption. U.S. Patent 5,071,449 (1991).

Sircar, S., Hanley, B.F.: Production of Oxygen Enriched Air by Rapid Pressure Swing Adsorption. *Adsorption*, **1**, 313 (1995).

Sircar, S., Kumar, R.: Equilibrium Theory for Adiabatic Desorption of Bulk Binary Gas Mixtures by Purge. *Ind. Eng. Chem. Process Des. Dev.*, **24**, 2, 358 – 364 (1985).

Sircar, S., Myers, A.L.: Gas Separation by Zeolites. In *Handbook of Zeolite Science and Technology*; Auerbach, S.M., Carrado, K.A.; Dutta, P.K., Eds.; Marcel Dekker: New York, Chap. 22; pp 1063-1105 (2003).

Sircar, S., Rao, M.B., Golden, T.C.: Fractionation of Air by Zeolites. In *Adsorption and its Applications in Industry and Environmental Protection*; Dabrowski, A., Ed; Elsevier: New York, **120**, Part 1; pp 395-423 (1999).

Skarstrom, C.W.: Method and Apparatus for Fractionating Gaseous Mixtures by Adsorption. U.S. Patent 2,944,627 (1960).

Skarstrom, C.W.: Oxygen Concentration Process. U.S. Patent 3,237,377 (1966).

Skarstrom, C.W.: Recent Development in Separation Science. CRC Press Cleveland, **2**, 95 (1972).

Sundaram, N., Wankat, P.C.: Pressure Drop Effects in the Pressurization and Blowdown Steps of Pressure Swing Adsorption. *Chem. Eng. Sci.*, **43**, 1, 123 – 129 (1988).

Taylor, R.J., Drago, R.S., George, J.E.: Characterization of a Cobalt(II) Cyanide Complex inside Zeolite Y that Reversibly Binds Oxygen. *Inorg. Chem.*, **28**, 6610 (1989).

Taylor, R.J., Drago, R.S., Hage, J.P.: A Reversible O2-binding System: Co(CN)5-3 Inside Zeolite. *Inorg. Chem*., **31**, 253 (1992).

Teague, K.G. Jr., Edgar, T.F.: Predictive Dynamic Model of a Small Pressure Swing Adsorption Air Separation Unit. *Ind. Eng. Chem. Res.*, **38**, 3761 (1999).

Wang, S.B., Li, H.T., Xie, S.J., Liu, S.L., Xu, L.Y.: Physical and Chemical Regeneration of Zeolitic Adsorbents for Dye Removal in Wastewater Treatment. *Chemosphere*, **65**, 82-87 (2006).

Yang, J., Park, M.W., Chang, J.W., Ko, S.M., Lee, C.H.: Effects of Bed Pressure Drop in a PSA Process. *Korean J. Chem. Eng.*, **15**, 2, 211 – 216 (1998).

Z. Yuwen, W. Yuyuan, G. Jianying, Z. Jilin: The Experimental Study on the Performance of a Small-Scale Oxygen Concentration by PSA. *Sep. Pur. Technol.*, **42**, 123-127 (2005).

Chapter 3

Preliminary Work II: Separation Performances of Packed Beds in Capillary and Conventional Tubes

The objective of this preliminary work is to experimentally assess the practicality of small scale pressure swing adsorption (PSA) for producing medical grade oxygen, before elaborate design and development are carried out to meet the research goal. A capillary tube for packed bed, which is inspired by the successful application of Micro-Electro-Mechanical System of gas chromatograph, is tested and its performance compared with that of a conventional tube for packed bed. A simple experimental setup was built to perform the tests.

3.1 MEMS Gas Chromatograph for Application of Adsorption Synthesis

Recent advances in the field of Micro-Electro-Mechanical System (MEMS) research has led to proposal of incorporating micro-channel adsorption into traditional PSA systems (Galbraith 2007). This idea is related to the successful application of MEMS gas chromatograph (GC), which is a micro-channel device for identifying different components of a gas mixture by separating them in space and time (Agah et al. 2006). The high-speed temperature programmed microfabricated GC column by Agah et al. is 25-cm-long, 150-μm-wide, and 250-μm-deep. This high surface-to-volume ratio design significantly reduces heat and mass transfer resistances, thus cutting down analysis time from minutes to seconds without compromising separation efficiency. Though GC and PSA are both gas separation devices, they are somewhat different as the prior is for sample analysis that handles small fluid volume, while the latter is for product synthesis thus may need to handle large fluid volume for high throughout. However, the advantage of improved heat and mass transfer in small column can be exploited for O_2 concentration via PSA.

Following the improved heat transfer efficiency in small column, heat effect from the adsorption and desorption can be minimized. Temperature shifts of about 20°C are common in PSA air drying and O_2-separation from air. A shift of this magnitude significantly affects O_2 separation in large units, but it does not affect the performance in small units (Knaebel). Moreover, small scale operation improves safety and lowers power consumption. Geometry wise, low dead volume is attainable from miniature O_2 concentrator. With lower dead volume, better process control and higher product flow rate per unit volume of separator may be achieved. A further advantage is that integration of a micro fabricated heat exchanger into the separator is made feasible (Velasquez-Garcia et al. 2007). This integrated heater can assist in the desorption process and minimize product loss in purging. To sum up, system miniaturization is very attractive because some physical phenomena perform much better or are more efficient when miniaturized to micro scale (Judy 2001).

3.2 Initial Assessment via Experimental Testing

In this section, experimental works and initial assessment results are reported. They are presented in this order: (1) experimental setup, measurement method, and operational conditions, (2) performance of miniature O_2 concentrator in different packed bed geometries, (3) performance of miniature O_2 concentrator versus gas flow rates, (4) performance comparison of current design pieces with large scale O_2 concentrators, (5) conclusion for preliminary assessment, and (6) future directions.

3.2.1 Experimental Setup, Measurement Method, and Operational Conditions

Figure 3.1 is a schematic diagram of experimental setup to test functionality and gauge performances of miniature O_2 concentrators via PSA concept. To simplify the setup, only a single bed was tested with traditional Skarstrom cycle: co-current pressurization with air, co-current production, counter-current depressurization, and counter-current purging using product gas, see Figure 3.2. Using dry feed air at 21 O_2/78 N_2/1 Ar and purge gas at 90 O_2/10N_2 directly from gas cylinders reduced piping and instrumentation complexity. Note that in real practice, when atmospheric air is used as feed gas, the presence of moisture can drastically reduce adsorbent capacity. Therefore, pre-drying of feed air is often achieved by using a layer of alumina silica in the feed end. However, this experimental work only considers small scale adsorption on the dry basis. Helium gas was the carrier and reference gas for mass spectrometry. Delivery pressures of feed air, purge gas and helium gas from gas cylinders were regulated at 50, 5 and 20 psig, respectively, while their flow rates were controlled by mass flow controllers. Three-way solenoid valves for flow switching were actuated by programmable logic controller. Pressure sensors were installed at the upstream and downstream of PSA column, and back pressure regulator was installed downstream at the production line. The amount of UOP Oxy LiX zeolite needed to fill a stainless steel tube was weighted by high precision analytical balance after an overnight sitting in the oven at 200^{o}C for moisture removal. Glass wool was used as particle filter at both ends of the tube. This adsorbent was then thermally activated for 8 hours using heating tape at 350^{o}C while flowing bone dry N_2 gas through the tube. Upon installing the activated zeolite column into the experimental setup with Swagelok fittings and silicone adhesive at the joints, soap leak test was applied to all the gas lines at high pressure gas flow to confirm zero leakage from the system. Essentially, for dealing with very small system, extra or redundant volume from the instrument and piping should be minimized so that system performance can be apprehended more accurately. This was achieved by using miniature solenoid valves, inserting small glass beads in the connector, and using small tubes in the process line.

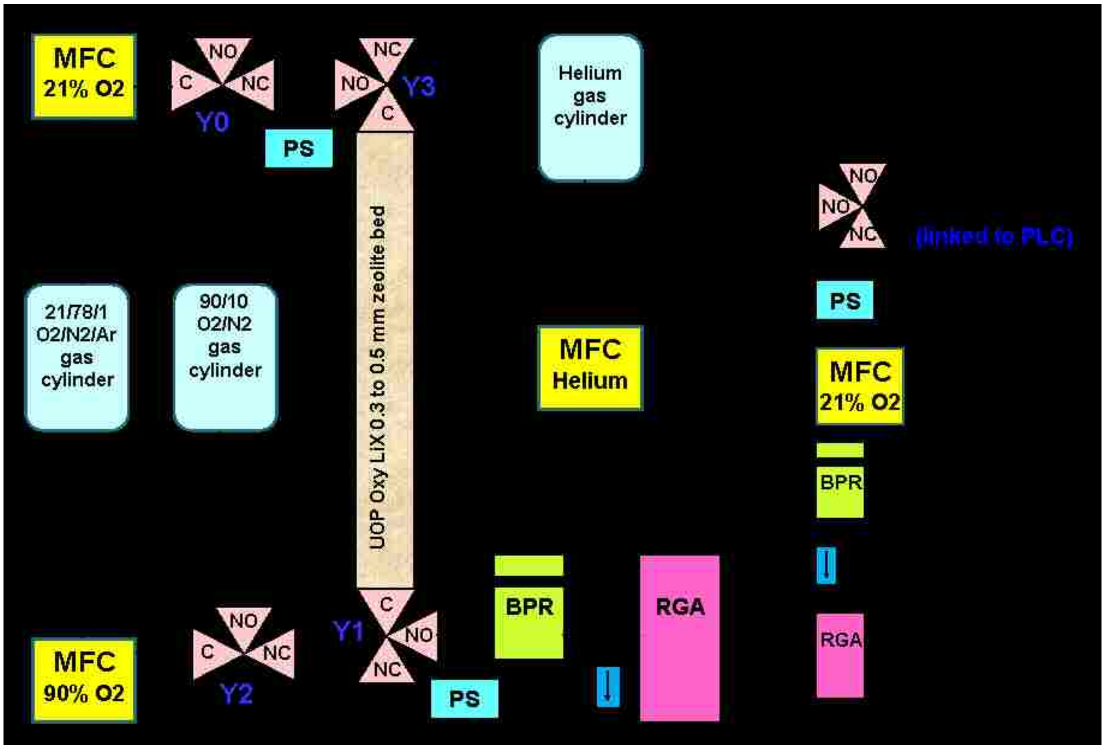


Figure 3.1: Experimental setup for testing miniature O_2 concentrator via PSA.

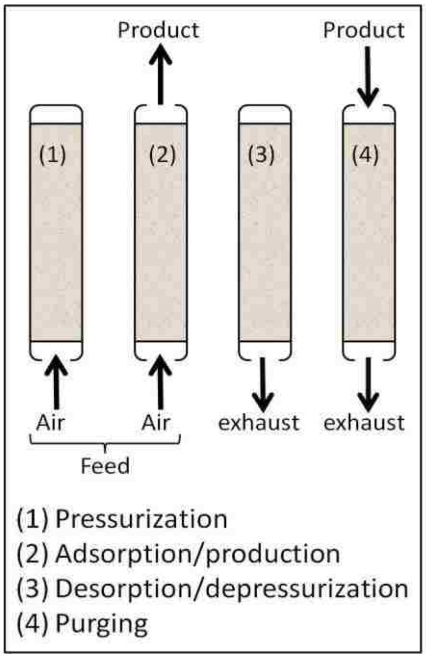
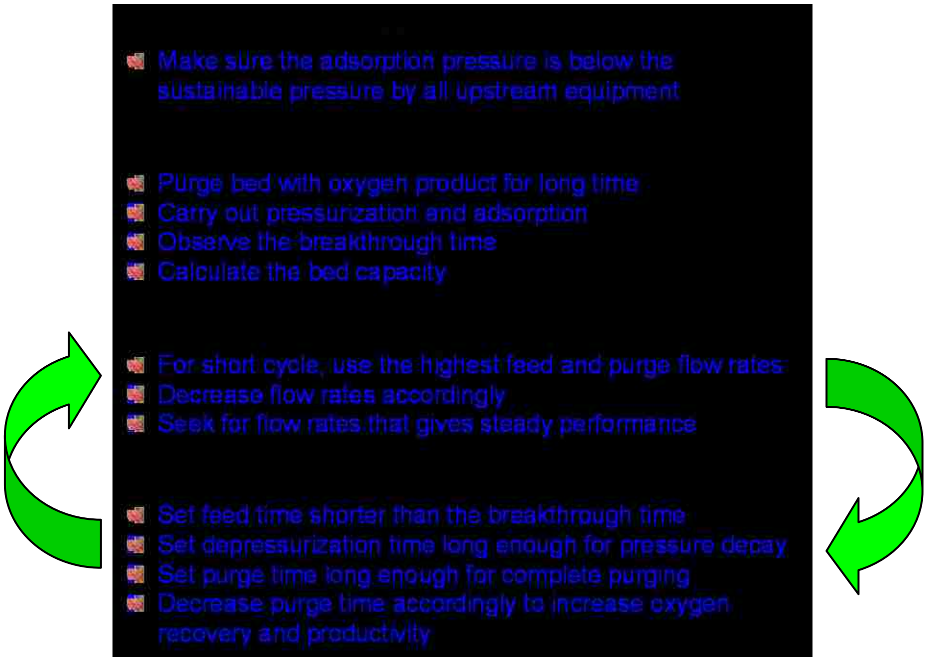


Figure 3.2: Skarstrom cycle for single bed.

Reliable flow rate and composition measurement is very crucial for this miniaturized system. Since the production rate from this miniaturized PSA was less than the sampling rate of an oxygen analyzer, mass spectrometer, which is able to measure the change in every second, was heavily depended for production line measurement. However, mass spectrometer is sensitive to the variations of the sampled gas pressure and its measurement tends to drift along time. To account for this reliability issue, careful calibration of mass spectrometer was performed according to the operation range and conditions (pressure, flow rates, compositions). More importantly, calibration correction curves were generated by measuring standard gas mixtures at the start of the day and at the end of day in between which experiments were carried out. All measurements of the day would be corrected using these correction curves. This method is able to give comparatively reliable analysis as repeating measurement results without correction scattered but with correction the results overlapped on each other. The plots with and without measurement correction will be given in Section 3.2.3.

Characterization of the system with operational conditions (e.g. pressure, flow rates, step times) for a new PSA system was performed by following the 4 modules given in Figure 3.3. It is a methodology with high degree of freedom, with 3rd and 4th modules interchangeable. High adsorption pressure gives higher loading capacity, but requires longer pressurization time and increases compression cost. Flow rates for air feed and purge gas must be set appropriately to avoid high pressure drop. Step times for depressurization, pressurization, adsorption and purging must be set correctly too to avoid bed exhaustion, incomplete- or over- regeneration. The purpose of this practice is to locate the suitable operating window that gives optimal performance.



- Make sure the adsorption pressure is below the sustainable pressure by all upstream equipment

- Purge bed with oxygen product for long time
- Carry out pressurization and adsorption
- Observe the breakthrough time
- Calculate the bed capacity

- For short cycle, use the highest feed and purge flow rates
- Decrease flow rates accordingly
- Seek for flow rates that gives steady performance

- Set feed time shorter than the breakthrough time
- Set depressurization time long enough for pressure decay
- Set purge time long enough for complete purging
- Decrease purge time accordingly to increase oxygen recovery and productivity

Figure 3.3: Locating the window of operating conditions for new PSA system.

3.2.2 Performance of Miniature Oxygen Concentrator in Different Adsorber Packed Bed Geometries

In this section the performances of miniature O_2 concentrators in different packed bed geometries are compared. In Figure 3.4: System A is a miniature PSA column with packed bed geometry of 14.6-cm-length and 0.4-cm-diameter, while System B is a PSA tube with packed bed of 151-cm-length and 0.15-cm-diameter. Both systems contain 1.6819 g dry UOP Oxy LiX zeolite with particle size ranging in between 300 – 500 μm. System B device feature is considered based on the successful application of long micro-channel GC in micro total analysis system (μ-TAS). The feasibility of this device feature for miniaturized bulk synthesis such as O_2 concentration via PSA is yet to be determined.

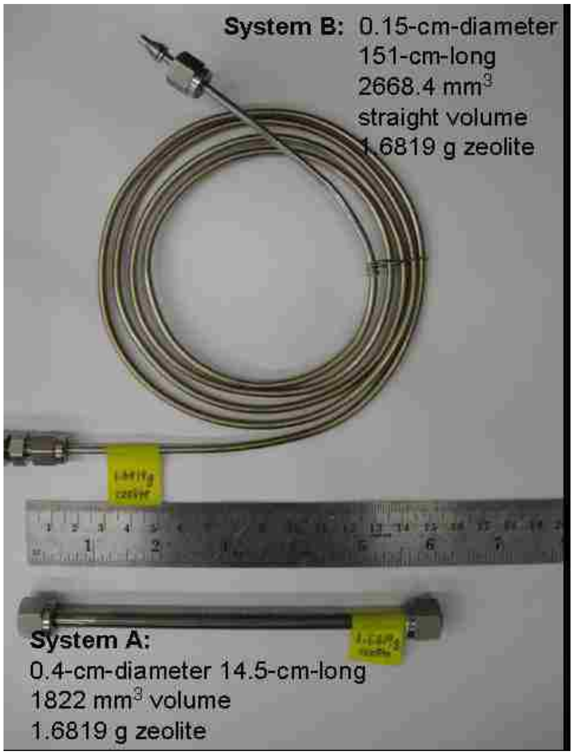


Figure 3.4: Systems A and B with different adsorber packed bed geometries and volumes but contained same amount of zeolite with the same particle sizes.

Table 3.1: Operating conditions and performances for Systems A and B.

Operating Conditions		
Parameters	**System A**	**System B**
Adsorption pressure (psig)	45	45
Depressurization time (sec)	4	15
Air Feed (20.85 mole % O_2)		
flow rate $(mL/min)_{STP}$	27	9.5
pressurization time (sec)	84	308
adsorption time (sec)	38	52
Purge with product (89.9 mole % O_2)		
flow rate $(mL/min)_{STP}$	27.35	9.6
purge time (sec)	9	11
Product per adsorption step:		
average product purity (O_2 mole %)	90.7	90.1
average product flow rate $(mL/min)_{STP}$	10.52	3.94
Pressure drop across bed (psi)	0.4	1

Performances		
Parameters	**System A**	**System B**
O_2 recovery (%)	20.57	12.57
Productivity (mmole O_2/g zeolite/cycle)	0.0625	0.0396
Productivity (mmole O_2/g zeolite/hr)	1.6668	0.3698

Note: Product per adsorption step = the product delivered during adsorption step
STP conditions: 1 atma, 273.15 K

Systematic exploration of optimum flow rates and step times for each system were carried out. While System A faced very little difficulty in characterization, System B was severely constrained by operating flow rate, it only gave positive productivity and O_2 recovery at very low flow rates. Table 3.1 summarizes the optimum operating conditions and performances of Systems A and B. Here, best conditions is defined as the parameters which are able to yield product equal to or above 90 mole % O_2 at cyclic steady-state and to give optimal productivity without experiencing bed exhaustion for hours. Both PSA systems were carried out at 45 psig (~ 4 atma) and experienced different pressure drops.

System A performed better than System B. Particularly, System B gave very low O_2 recovery and productivity. There are several reasons why System B performed poorer. First of all, the cycle time for System B was ~ 3 times the cycle time of System A, due to the feed and purge flow rates of System B which were 1/3 those of System A. The long cycle time of System B had resulted in its low productivity. Shortening the cycle time for System B by introducing higher flow rates caused early breakthrough and resulted in negative O_2 recovery.

Since the pressure drop in both systems were very small (0.4 and 1 psi), poor performance of System B was not caused by this issue, instead, it was caused by inadequate adsorbent packing. With the same capillary tube, a straight tube was able to contain ~ 3 g zeolite, but after bended into spiral shape, the tube was only packed with ~ 1.7 g of zeolite with the same particle size due to packing difficulty. Moreover, the ratio of tube diameter to particle diameter, D_{tube}/d_p, was only ~ 3.75, which gave rise to large interstitial void in the system. The impact from this poor packing include: (1) micro-channeling followed by earlier breakthrough and (2) much O_2 from the void was lost during depressurization. Figure 3.5 graphically depicts the poor packing in System B. An adequate ratio of D_{tube}/d_p should be at least 10 so that void space is reduced and gas channeling is minimized. Note that for this small diameter tube, extremely small particle size must be adopted to achieve the adequate diameter ratio, which will bring forth high pressure drop problem. Figures 3.6 and 3.7 present the consequences of channeling versus gas flow rates in System B. With the same feed amount, high gas flow rate at shorter step time channeled between particles instead of penetrating into the adsorbent. Low gas flow rate alleviated this problem a little thus System B was only characterizable at low flow rates.

On the other hand, System A was superior because its geometry was more operational friendly, as easily noticed from the depressurization time. For the same pressure difference, System A only took 4 sec but System B took 15 sec for the pressure to decay from high to low level, see Figure 3.8. Based on all these reasons, the proposal of using long micro-channel PSA is not appealing.

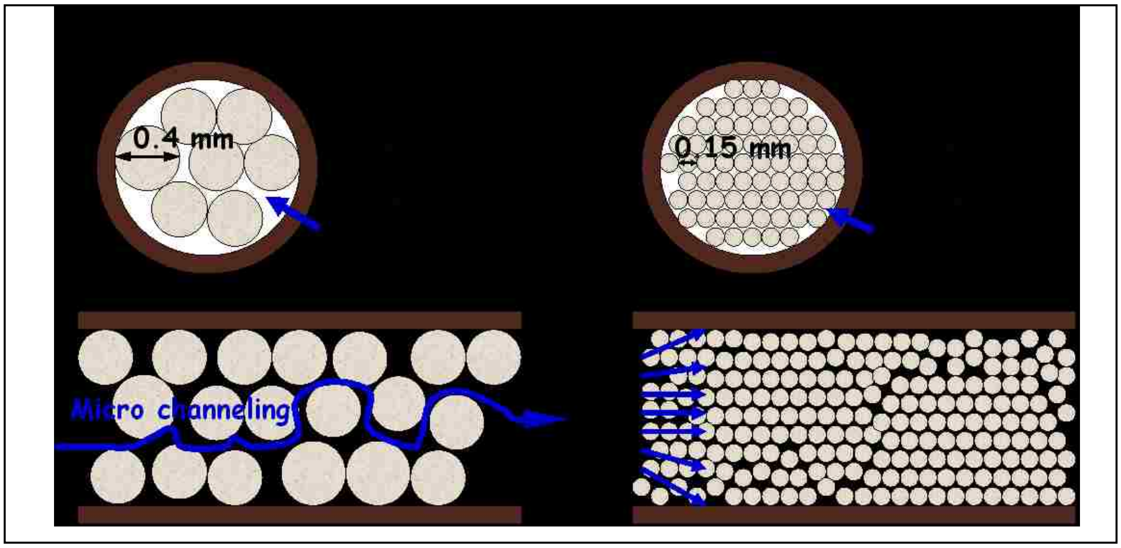
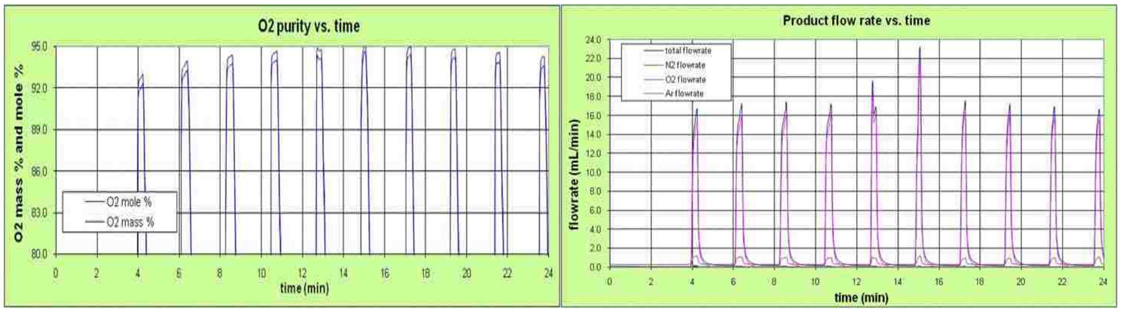
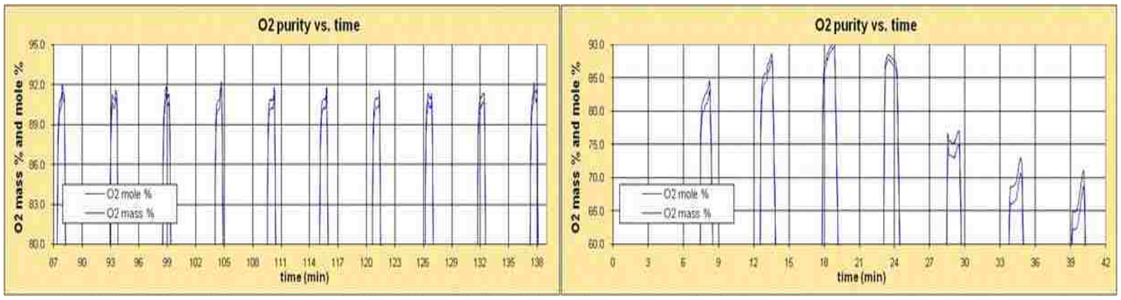


Figure 3.5: Poor performance of System B due to inadequate adsorbent packing. Improved packing density minimizes micro-channeling and product loss problems.



Figures 3.6 (a) and (b) show the product purity and product flow rate from System B subjected to 15 mL/min air flow rate. Production time reduced from 27 sec to 22 sec as a result of channeling.



Figures 3.7 (a) and (b) show the product purity from System B subjected to different purge flow rates. (a) When subjected to 10 mL/min purge flow rate for 11 sec, product purity is maintained at about 90 mole % O_2. (b) When subjected to 20 mL/min purge flow rate for 5.5 sec, product purity degraded.

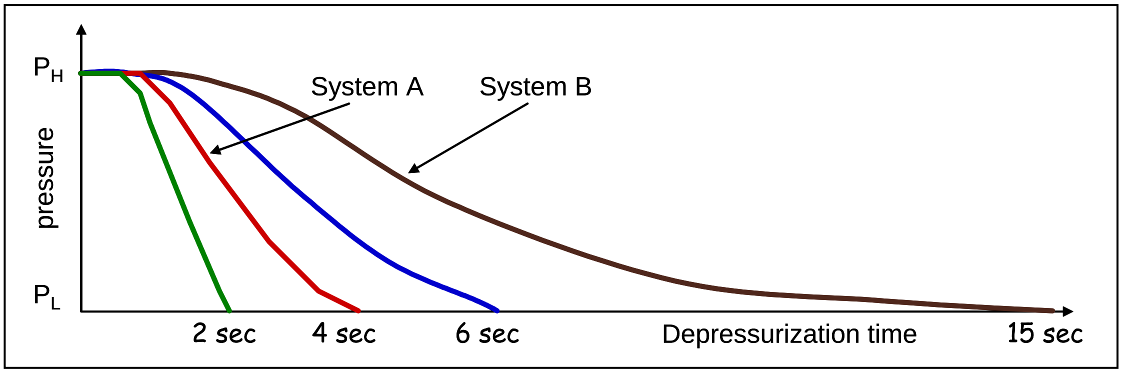


Figure 3.8: Depressurization times required for pressure decay by Systems A and B.

Finally, instrument constraints also play a major role in system performance. Figure 3.9 is a close snap shot of miniature solenoid valves connected to the feed end of PSA system and to the upstream pressure sensor. Note that the exhaust port at the solenoid valve is very small, it had restricted the blow down process and lengthened the depressurization time. On the other hand, for dealing with very small system, extra or redundant volume from the instrument and piping should be minimized so that system performance can be apprehended more accurately. This was achieved by using miniature solenoid valves, inserting small glass beads in the connector, and using small tubes in the process line where operating efficiency would not be affected.

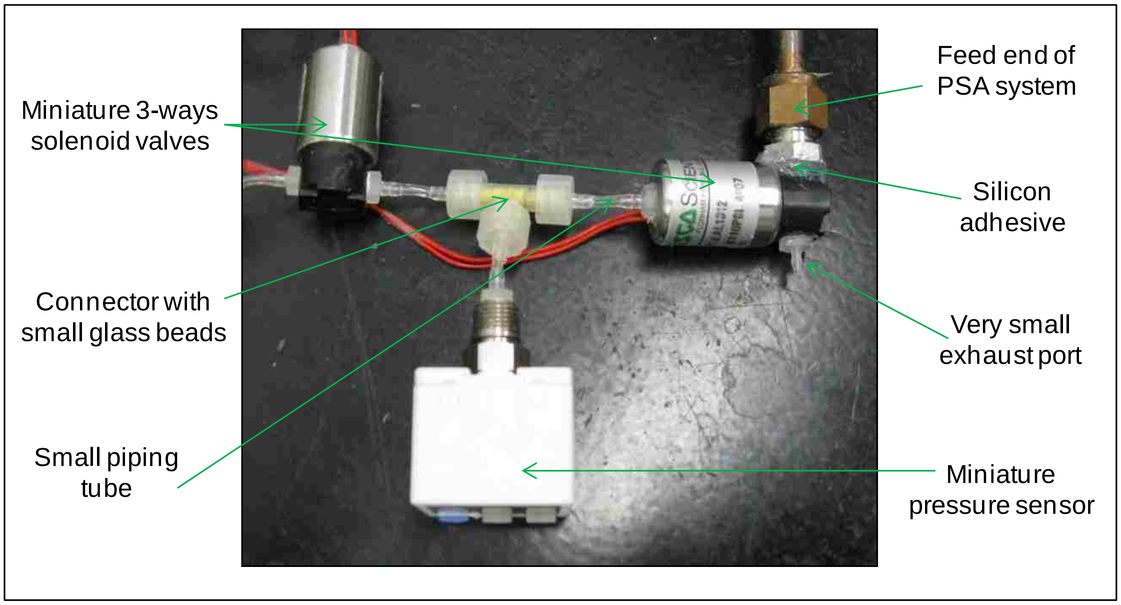

Miniature 3-ways solenoid valves
Feed end of PSA system
Silicon adhesive
Connector with small glass beads
Very small exhaust port
Small piping tube
Miniature pressure sensor

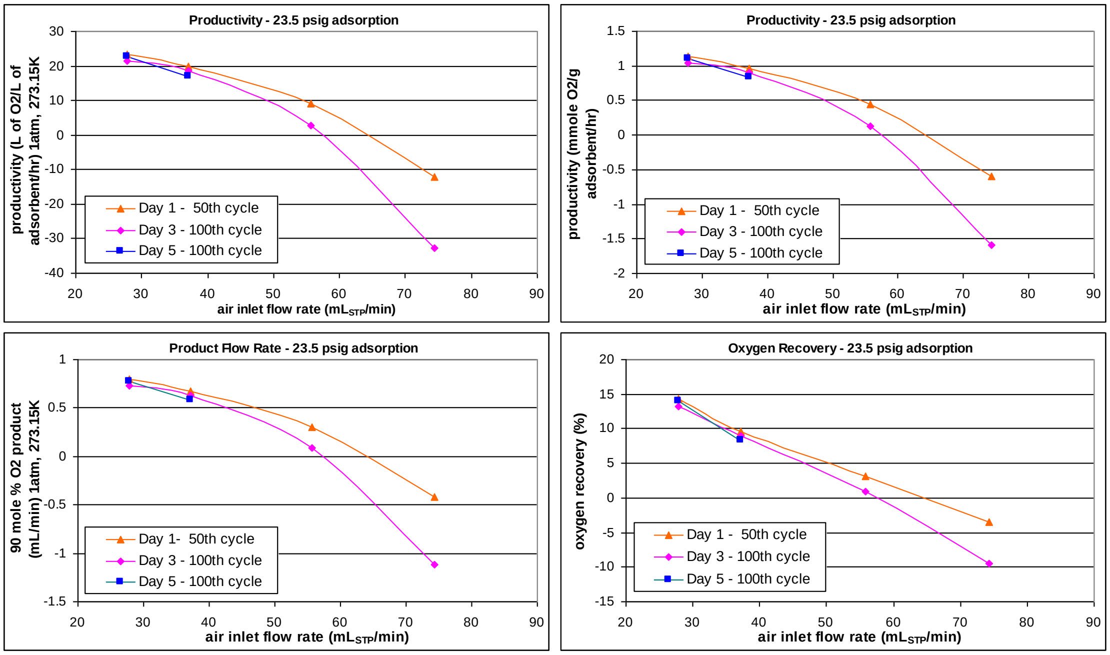
Figure 3.9: Instrument constraint and minimizing dead volume.

3.2.3 Performance of Miniature Oxygen Concentrator versus Gas Flow Rates

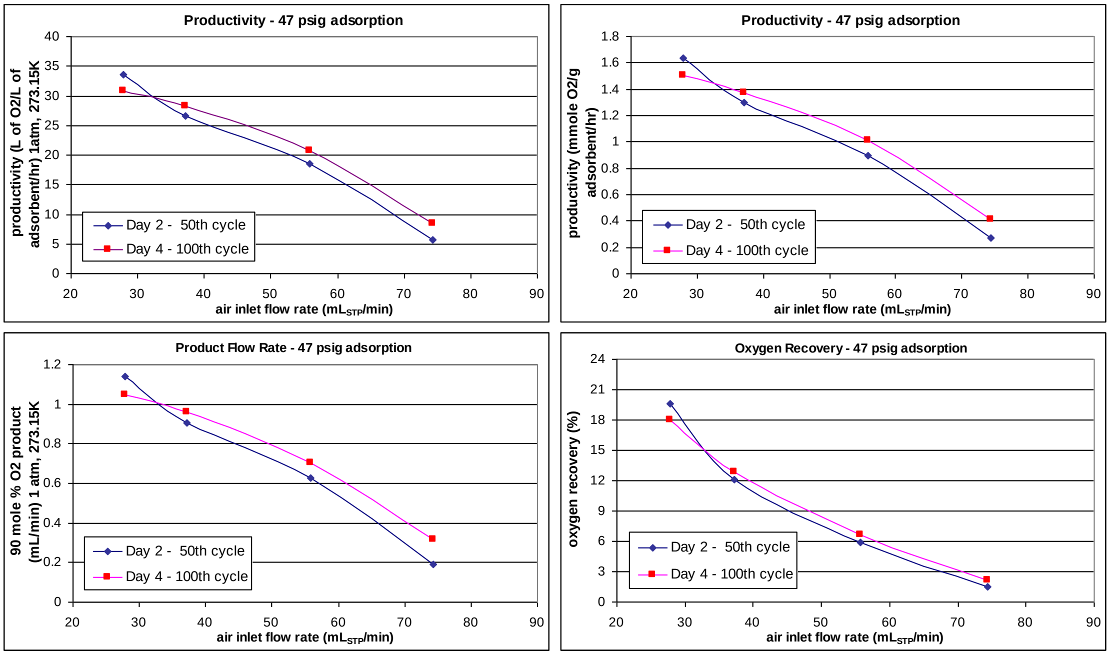
An unexpected observation from the experimental testing of miniature O_2 concentrator was the decrease of productivity along with the increase of air inlet flow rate, even though the pressure drop across column subjected to the said high flow rate was negligible. Poor packing in System B may be used to explain this occurrence, but a root cause for this occurrence in System A is yet to be determined. In order to confirm this phenomenon in the miniature PSA system, System A with the same packed bed geometry (14.6-cm-length and 0.4-cm-diameter) and same amount of zeolite (1.6819 g dry UOP Oxy LiX zeolite) with 350 µm particle size was tested. The ratio of tube diameter to particle diameter, D_{tube}/d_p = 4 mm/0.35 mm = 11.43, was beyond the threshold of poor packing. Characterization was performed at different air inlet flow rates, ranging from 30 mL_{SATP}/min to 80 mL_{SATP}/min, and at 23.5 psig and 47 psig adsorption pressures. Each condition was allowed to reach steady-state condition of 50 cycles (Day 1 and Day 2) and 100 cycles (Day 3, Day 4, Day 5). All data regarding adsorption at 23.5 psig were generated on Day 1, 3, 5, while data regarding 47 psig adsorption on Day 2 and 4.

For adsorption at 23.5 psig, the miniature adsorption column in all runs received the same amount of air feed (41.3 mL_{SATP}/cycle). Same amount of product gas (5.5 mL_{SATP}/cycle) was used to purge the column at all runs, except those runs subjected to 80 mL_{SATP}/min air inlet flow rate. They received 5.75 mL_{SATP}/cycle product purge gas because the column was unable to produce 90 mole % O_2 product if purged with less than 5.75 mL_{SATP}/cycle product gas. Pressure drop across column was between 0.2 psi and 0.8 psi, with the highest pressure drop for 80 mL_{SATP}/min air inlet flow rate; these pressure drops are considered negligible to affect adsorbent equilibrium capacity. The performances are given in Figures 3.10, all volumetric flow rates in these figures were converted from SATP conditions (standard ambient temperature and pressure, 1 atma, $21^{o}C$) to STP conditions (1 atma, $0^{o}C$). Note that the column performance degraded from Day 1 onwards. Column was exhausted on Day 5 and could not produce 90 mole % O_2 product when subjected to 60 mL_{SATP}/min and 80 mL_{SATP}/min air inlet flow rates. The unexpected observation of decreased productivity along with the increased air inlet flow rate still prevailed in these test runs. Moreover, when product gas flow rate for purging was elevated from 30 mL_{SATP}/min (11 sec step time) to 40 mL_{SATP}/min (8.25 sec step time), the bed was exhausted and gave rise to poorer performances.

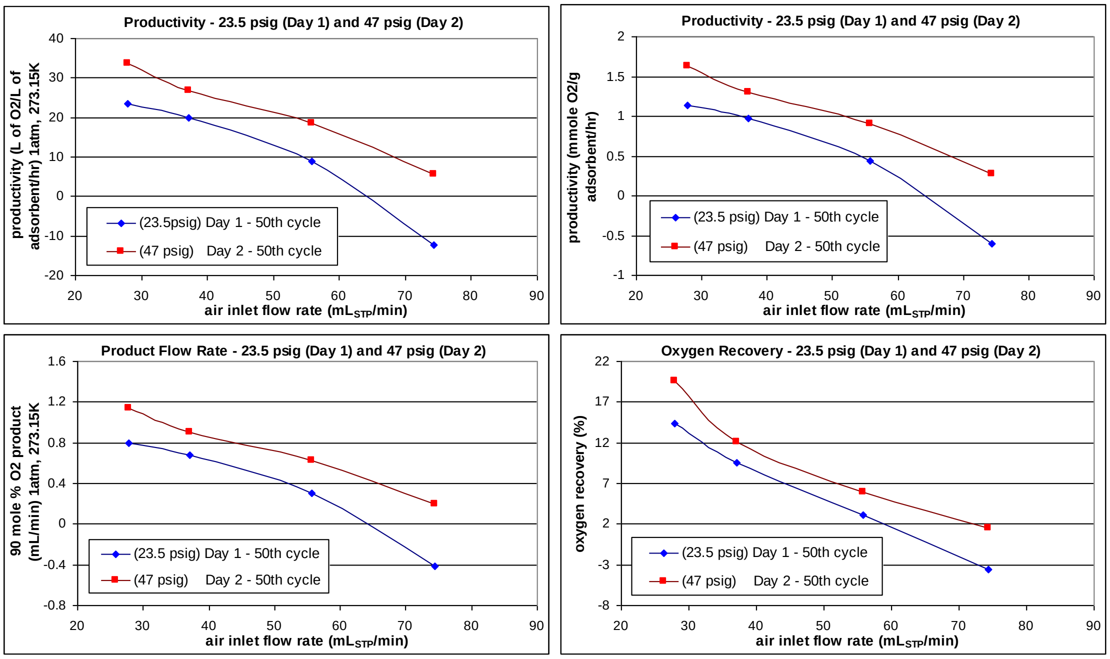
For adsorption at 47 psig, the miniature adsorption column received 63.3 mL_{SATP}/cycle air feed on Day 2 but 62.5 mL_{SATP}/cycle air feed on Day 4. Same amount of product gas (5.5 mL_{SATP}/cycle) was used to purge the column at all runs, except those runs subjected to 80 mL_{SATP}/min air inlet flow rate which received 5.75 mL_{SATP}/cycle product purge gas. Pressure drop across column was between 0.2 psi and 0.8 psi. The performances are given in Figures 3.11. Note that the column performances on Day 4 were generally better than those on Day 2. This might due to the fact that column is less exhaustive upon receiving a little lesser air feed amount per cycle. Again, the unexpected observation of decreased productivity along with the increased air inlet flow rate still prevailed in these test runs.



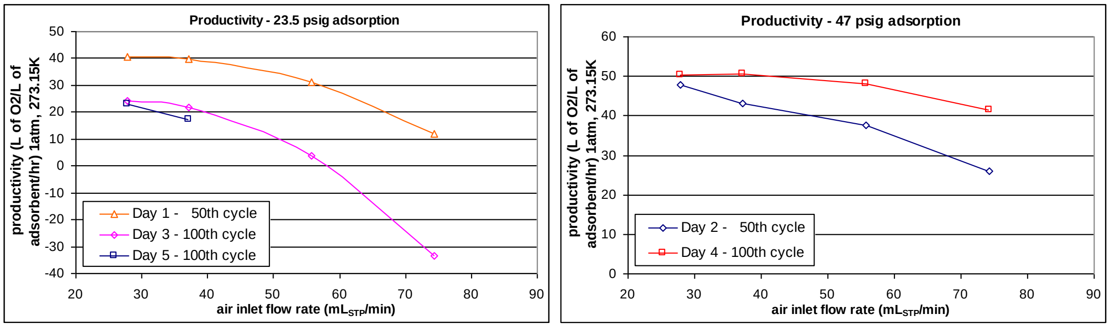
Figures 3.10: Performances of miniature O_2 concentrator at 23.5 psig adsorption pressure.



Figures 3.11: Performances of miniature O_2 concentrator at 47 psig adsorption pressure.



Figures 3.12: Performances of miniature O_2 concentrator at 23.5 psig and 47 psig adsorption pressure.



Figures 3.13: Productivities of miniature O_2 concentrator at 23.5 psig and 47 psig adsorption pressure without measurement correction.

Figures 3.12 compare the performances of miniature O_2 concentrator at 23.5 psig of Day 1 and at 47 psig of Day 2. Note that productivity, O_2 recovery and product flow rate were higher at 47 psig because the bed offered higher equilibrium capacity at elevated adsorption pressure.

The performances in Figures 3.10, 3.11 and 3.12 are the results from corrected measurement using calibration correction curves; all results demonstrated good consistency with repeating runs. Compare the productivities of corrected measurement in Figures 3.10 and 3.11 to the productivities of uncorrected measurement in Figures 3.13. The measurement results without correction scattered. Therefore, inclusion of calibration correction in the measurement method is very essential to produce consistent results especially for characterizing miniature system.

3.2.4 Performance of Miniature versus Large Scale Oxygen Concentrators

In order to deduce the feasibility of Systems A and B, their performances are compared to those of experimental or practical large scale O_2 concentrators from the literatures (Matz and Knaebel 1988; Sircar and Hanley 1995; Mendes et al. 2000; Santos et al. 2007) as given in Table 3.2. Note that parallel comparison cannot be directly inferred because all other processes were carried out using different adsorbent types, different number of beds and different process cycles. Nonetheless, all these O_2 concentrators are used to produce about 90 mole % O_2-enriched air.

Table 3.2: Compare performances of miniature oxygen concentrator (Systems A on Day 1 and 2) and large scale oxygen concentrators from the literatures.

Different systems and performances	**This Invention – miniature oxygen concentrator**		**Matz and Knaebel 1988**	**Sircar and Hanley 1995**	**Mendes et al. 2000 and 2001**	**Santos et al. 2007**
Process cycle	Skarstrom cycle: •Feed pressurization •Adsorption •Depressurization •Purge		PVSA: •Product pressurization (10 – 70 s) •Adsorption (100 s) •Depressurization (30 s, exhaust flows into a sub-atmospheric tank) •Purge (0 - 60 s)	RPSA: •Feed pressurization + adsorption (6 s) •Depressurization + purge (6 s)	Skarstrom cycle: •Feed pressurization •Adsorption •Depressurization •Purge	•Feed pressurization •Adsorption •Provide equalization •Depressurization •Purge •Receive equalization
Total cycle time (sec)	140.6	96.1	200	12	180	N/A
Zeolite type	LiX	LiX	5A	NaX	5A	AgLiLSX
Particle diameter (mm)	0.35	0.35	N/A	0.5	1.7	1
Adsorbent amount (gram)	1.6819	1.6819	184	2 x 2041.2	2 x 308.5	2 x 101.6
Bed volume (cm^3)	1.835	1.835	246.3	2 x 3030.7	2 x 412.33	2 x 84.82
Pressure (psig)	2 – 47	2 – 23.5	-12 – 36	2 – 32	0 – 84	0 – 29
Feed air	78 N_2 /21 O_2 /1 Ar		78.3 N_2 /21.7 O_2 + Ar	Dry air	78 N_2 /22 O_2 + Ar	78 N_2 /21 O_2/1 Ar
Product O_2 purity (mole %)	90	90	~94% O_2 + 5% Ar	90	93.7 (O_2 + Ar)	98.7
Net product flow rate (mL$_{STP}$ /min)	1.14	0.80	74.06 – 123.3 (with ~94%O_2)	~2413	55.2	30 – 50
O_2 recovery (%)	19.6	14.4	21.8 – 41.9	~8.3	3.6	5.6 – 7.6
O_2 Productivity (mmole O_2/ g adsorbent/cycle)	0.0639	0.0305	0.0563 – 0.0937	~0.00475	0.0108 – 0.0112 (product of 90 – 93.7 % O_2)	N/A
O_2 Productivity (mmole O_2/ g adsorbent/hr)	1.6350	1.1427	1.0130 – 1.6868	~1.42	0.2156 – 0.2245 (product of 90 – 93.7 % O_2)	0.3902 – 0.6503
O_2 Productivity (L$_{STP}$/L dsorbent/hr)	33.58	23.47	16.64 – 27.71	~21.5	3.61 – 3.76	10.47 – 17.45
Bed Size Factor (lb of adsorbent /TPD$_c$)	1594	2271	2571 – 1544	~1830	12075 - 11598	6672 - 4003

STP = 1 atm abs, 273.15K

Notice that some large scale O_2 concentrators used superior adsorbent such as AgLiLSX by Santos et al. (2007) and some used superior process cycle such as PVSA by Matz and Knaebel (1988). Even so, miniature O_2 concentrator performed better especially in terms of productivity (L_{STP}/L adsorbent/hr) and bed size factor (lb of adsorbent /TPD_c).

From these comparison results, three possible reasons for the poorer performance of the large systems were deduced: (1) poorer gas distribution, (2) incomplete utilization of adsorbent bed due to mass transfer resistance (2) significant non-isothermal condition in large bed. Therefore, though not overall conclusive, System A demonstrated better separation efficiency due to miniaturized scale.

3.2.5 Conclusions of Preliminary Assessment

Miniature O_2 concentration via PSA concept was demonstrated. A consistent calibration and correction method was used to measure product flow rate and composition. Though it was faced with unexpected phenomenon of decreased productivity at increased gas flow rates, O_2 productivity (L_{STP}/L adsorbent/hr) and bed size factor (lb of adsorbent /TPD_c) were relatively larger and smaller, respectively, than those of larger systems from the literatures.

System A was superior to System B due to its operational friendly geometry and adequate packing. Nonetheless, product throughput of System A is still far from meeting the requirement, multiple units in parallel are needed to increase product flow rate. From this experimental work, several factors were identified that affect system performance, they were, cycle time, packing density (particle size and void), pressure drop (if packing density and bed length are very high), bed geometry and instrumental constraints. Poor gas distribution (significant dead volume), incomplete utilization of adsorbent, high mass and heat transfer resistances, and non-isothermal operation might have contributed to poor performance in large beds from the literatures. This work was the first effort to evaluate feasibility of miniature PSA for O_2 concentration. The physical and operational designs was not optimal thus there are plenty rooms for improvement to yield higher O_2 recovery and productivity. Special attention in checking the overall and component mass balance at cyclic steady-state was not committed which might have given rise to misinterpretation of the experimental data.

3.2.6 Future Directions

Future development and elaborated design of miniature O_2 concentrator can focus on these areas: improving bed geometry, modifying process steps, reducing cycle time (by introducing higher flow rate without compromising separation efficiency), using different adsorption pressure, achieve complete depressurization (so that expanded purge volume can reduce the amount of purge gas required), and minimize instrumental limitations. These works should be accompanied by theoretical study and simulation. Special care must be given to accurate measurement of all the product purity and flow rate data, in addition to checking mass balances of the system.

3.3 References

Agah, M., Lambertus, G.R., Sacks, R., Wise, K.: High-Speed MEMS-Based Gas Chromatography. *J. MEMS*, **15**, 1371-1378 (2006).

Galbraith, S.D.: Meso-Frequency Traveling Wave Electro-Kinetic Continuous Adsorption System. U.S. Patent 7,291,271(2007).

Hutson, N.D., Rege, S.U., Yang, R.T.: Mixed Cation Zeolites: LixAgy-X as a Superior Adsorbent for Air Separation. *AIChE J.*, **45**, 724-734 (1999).

Judy, J.W.: MEMS: Fabrication, Design, and Applications. *Smart Materials and Structures*, **10**, 1115-1134 (2001).

Knaebel, K.S.A.: "How To" Guide for Adsorber Design. Adsorption Research, Inc. http://www.adsorption.com/publications/AdsorberDes2.pdf

Matz, M.J., Knaebel, K.S.: Pressure Swing Adsorption: Effects of Incomplete Purge. *AIChE J*., **34**, 1486 – 1492 (1988).

Mendes, A.M.M., Costa, C.A.V., Rodrigues, A.E.: Analysis of Nonisobaric Steps in Nonlinear Bicomponent Pressure Swing Adsorption Systems. Application to Air Separation. *Ind. Eng. Chem. Res.,* **39**, 138-145 (2000).

Santos, J.C., Cruz, P., Regala, T., Magalhaes, F.D., Mendes, A.M.M.: High-purity Oxygen Production by Pressure Swing Adsorption. *Ind. Eng. Chem. Res.*, **46**, 591-599 (2007).

Sircar, S., Hanley, B. F.: Production of Oxygen Enriched Air by Rapid Pressure Swing Adsorption. *Adsorption*, **1**, 313-320 (1995).

Velasquez-Garcia, L.F., Hill, T.F., Wilhite, B.A., Jensen, K.F.: A MEMS Singlet Oxygen Generator – Part I: Device Fabrication and Proof of Concept Demonstration". *J. MEMS*, **16**, 1482-1491 (2007).

Chapter 4

Experimental Simulated Rapid Pressure Swing Adsorption Cyclic Process for Adsorbent Bed Size Reduction

This experimental work was carried out using improved experimental setup and polished experimental procedures compared to those of the preliminary work in Chapter 3. It was experimentally demonstrated using a Skarstrom-like PSA cycle that the bed size factor (BSF) of a PSA process cannot be reduced indefinitely by lowering the cycle time (t_c) and using very small particle size (d_p). However it was shown that a rapid pressure swing adsorption (RPSA) process can deliver a BSF of 25 – 50 lbs/TPD$_c$ with an O_2 recovery of 25 – 35% using (a) small particles of LiX zeolite (~ 0.35 mm), (b) adsorption pressure of 3 – 4 atm, and (c) total cycle time of 3 – 5 seconds. This work was filed in U.S. Patent with application number 2010/0300285 and published in Industrial & Engineering Chemistry Research (Chai et al. 2011).

4.1 Adsorbent Bed Size Reduction using Rapid Pressure Swing Adsorption

Adsorbent bed size reduction is commonly carried out using fast cycled PSA process so adsorber can be used more frequently and its amount required for a specific production rate can be reduced. However, BSF cannot be reduced indefinitely due to the non-negligible mass, energy, and momentum transfer resistances in the system, especially during rapid cycle, as elaborated in Sections 1.1.3, 1.1.7 and 1.1.8.

Therefore, it is proposed that a realistic BSF vs. total cycle time for a PSA O_2 concentrator should look like that in Figure 4.1.

4.1.1 Model Simulated Linear Relationship of Bed Size Factor versus Total Cycle Time

The ultra rapid pulsed PSA (PPSA) process simulated by Rama Rao, Farooq and Krantz (2010) used a total cycle time of only ~ 0.25 seconds. The resulting BSF for the process producing ~ 90% O_2 from air was very low (~ 5 lbs/TPD_c) with an O_2 recovery of ~ 24%. The PPSA process consisted of only two cyclic steps: (a) adsorption where compressed air was passed through a shallow zeolite bed (L/D ~ 0.1 – 0.2) of very small particles (d_p = 20 μm) while directly producing the O_2 product gas at near ambient pressure at the column exit end, and (b) countercurrent depressurization where the column pressure was reduced to a near ambient level producing a N_2 enriched waste gas. The N_2 adsorption front penetrated only a small fraction of the zeolite bed length during step (a), thereby, leaving a portion of the bed filled with product quality O_2 to be used as back purge gas for desorption of N_2 during step (b). The key model assumptions of this process are isothermal operation, linear driving force (LDF) adsorption kinetic model, Langmuir isotherms, inclusion of axial dispersion, and Darcy's law for calculating average velocity from imposed pressure drop. Table 4.1 reproduces a few examples of the simulated PPSA process performance using a dry, CO_2-free feed gas (21% O_2 + 79% N_2) at 3.5 atm and 298 K for production of ~ 5 LPM of ~ 90% O_2 employing a Ag-Li-X zeolite.

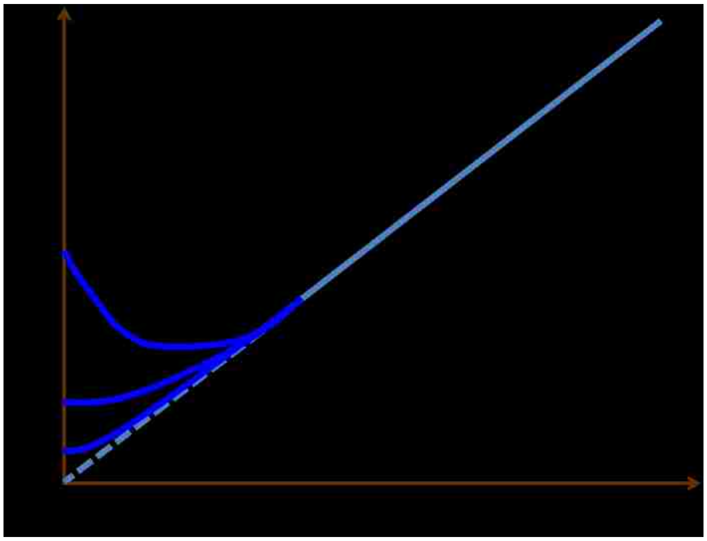


Figure 4.1: Schematic drawing of BSF vs cycle time for a RPSA process.

Table 4.1: Model simulated performances of PPSA cycle by Rama Rao et al.

Cycle times (sec)		Column Dimensions				d_p/L (x10^3)	BSF * (lbs/TPD$_c$)	O_2 recovery (%)
Total	Step (a)	Length L (cm)	Diameter D (cm)	L/D	Volume V (cm^3)			
0.235	0.017	0.7	7.34	0.095	29.6	2.86	5.2	24
0.444	0.034	1.0	8.14	0.123	52.1	2.00	8.9	26
1.538	0.137	2.0	9.96	0.201	155.9	1.00	26.4	27
3.125	0.308	3.0	11.21	0.268	295.9	0.67	50.6	30

*calculated using sorbent bulk density = 0.8 g/cc

It may be seen from Table 4.1 that the BSF for the PPSA concept progressively decreased with decreasing total cycle time which was reduced to well below 1 second in order to achieve a BSF of less than 10 lbs/TPD$_c$. A 'pancake' shaped adsorber consisting of a narrow layer (adsorber length, $L \leq 1.0$ cm) of very small particles (d_p = 20 μm) of the zeolite in conjunction with a large adsorber diameter (L/D < 0.25) was used in the model simulation to enhance the effective adsorbate mass transfer coefficient and to circumvent the negative effects of column pressure drop in order to achieve such a low BSF. The authors suggest that the BSF can be reduced indefinitely by increasing the cycle frequency as far as allowable by mechanical design of the PPSA unit.

The actual performance of the proposed PPSA concept has not been experimentally tested. Two critical hydrodynamic issues such as gas mal-distribution and particle agglomeration associated with the proposed design may impede the performance of the actual process (Porter et al. 1993; Moulijn and Van Swaaij 1976). Ad hoc reduction of the adsorbent particle size may also impede the performance due to the dominating effect of axial dispersion (Zhong et al. 2010). In addition, the simulation ignores the potentially negative influence of finite heat transfer resistance between the gas and the solid phases on the performance of a rapid PSA cycle by assuming isothermal operation (Sircar 2005).

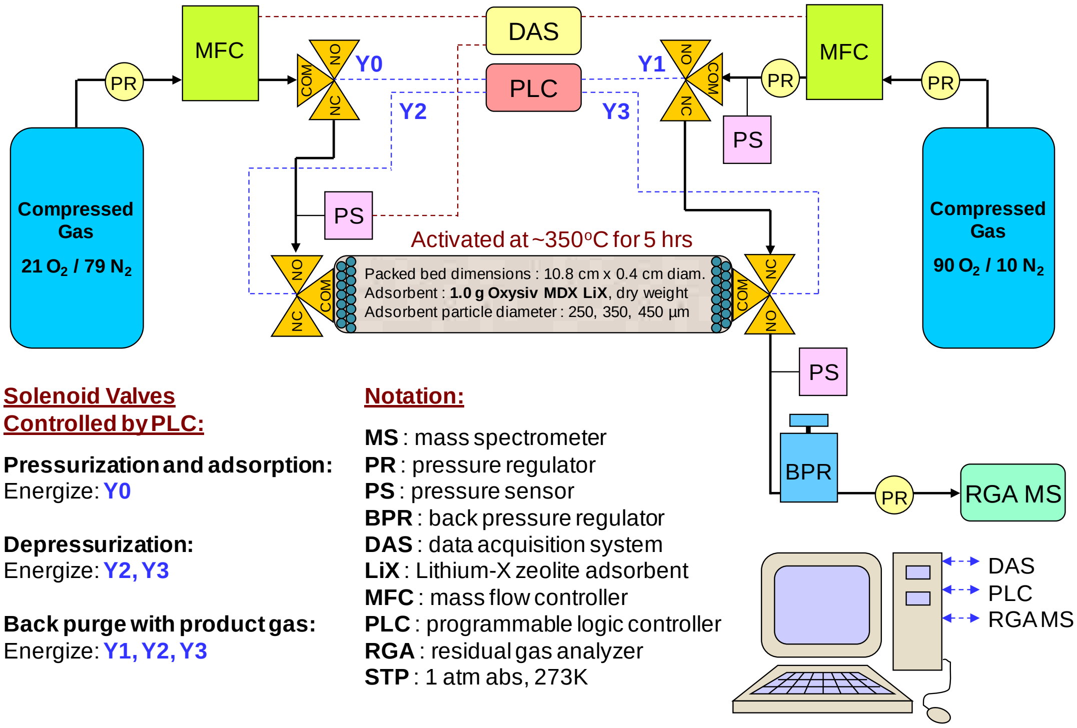
4.1.2 Experimental Simulated Nonlinear Relationship of Bed Size Factor versus Total Cycle Time

The present work experimentally demonstrates that the BSF of a rapid Skarstrom-like PSA cycle for production of ~ 90% O_2 cannot be reduced indefinitely by lowering the cycle time. The BSF vs cycle time profile exhibits a minimum value as schematically described by Figure 4.1 due to the impediments introduced by finite mass and heat transfer coefficients between the gas and solid phases as well as column pressure drop during the process steps. The tests were conducted employing small particles of LiX zeolite of different diameters, different super-atmospheric adsorption pressures in conjunction with a near ambient desorption pressure, and different cycle frequencies.

A PSA cycle was chosen for this study because a PSA mode of MOC design may have several practical advantages over a VSA or a PVSA design: (a) direct production of a compressed stream of O_2 product gas, which provides a good driving force to overcome the column pressure drop during steps like back purge and pressurization with product gas, particularly when small particle sizes are used and faster execution of these steps are needed, and (b) elimination of the need for a vacuum pump required by a VSA or a PVSA system which can be bulky and a potential source of system leakage. A PVSA system requires a compressor as well as a vacuum pump which complicates system maintenance. An air compressor, on the other hand, can be compact and efficient for small scale application. These advantages must be weighed against some potential performance deficiencies of a PSA system, such as relatively lower O_2 recovery and increased BSF (Sircar 1988).

4.2 Experimental Setup

A single column semi-continuous PSA test apparatus consisting of a mini-adsorber (0.4 cm diameter x 10.8 cm length) was constructed for the tests. It was packed with 1 gram of a commercial sample of LiLSX (Oxysiv MDX LiX) zeolite obtained from the UOP Corp. The key components of the apparatus were three-way solenoid switch valves, pressure sensors at various points, a back pressure regulator, mass flow controllers, pressure regulators and a programmable logic controller. It was also equipped with a dedicated mass spectrometer for gas analysis and a data acquisition system. Figure 4.2 is a schematic drawing of the apparatus.



Packed bed dimensions : 10.8 cm x 0.4 cm diam.
Adsorbent : **1.0 g Oxysiv MDX LiX**, dry weight
Adsorbent particle diameter : 250, 350, 450 μm

Solenoid Valves Controlled by PLC:

Pressurization and adsorption:
Energize: **Y0**

Depressurization:
Energize: **Y2, Y3**

Back purge with product gas:
Energize: **Y1, Y2, Y3**

Notation:

MS : mass spectrometer
PR : pressure regulator
PS : pressure sensor
BPR : back pressure regulator
DAS : data acquisition system
LiX : Lithium-X zeolite adsorbent
MFC : mass flow controller
PLC : programmable logic controller
RGA : residual gas analyzer
STP : 1 atm abs, 273K

Figure 4.2: Schematic drawing of test apparatus.

4.2.1 Programmable Logic Controller

ELC-PB-14NNDR programmable logic controller (PLC) from Eaton Cutler-Hammer was used to control the fast switching of solenoid valves at precise timing. Depending on the type of power supply required, wiring of the solenoid valves to PLC were completed according to the schematic diagrams given in Figure 4.3. Surge adsorbers and diodes are installed in parallel with, respectively, the alternating-current (AC) and direct-current (DC) powered solenoid valves, in order to protect internal circuitry on the relay outputs of PLC. The relay contact life decreases significantly if this is neglected.

Multiple solenoid valves can be installed in parallel fashion to a single relay output as long as the switching sequence and timing of these valves are the same and as long as the total current required does not exceed the maximum capacity of the single output.

If a big solenoid valve requires large current that exceeds the maximum capacity of a single relay output of the PLC, an intermediate relay with large output capacity can be installed as shown in Figure 4.4, then the big solenoid valve can be operated using this intermediate relay.

A sample of the PLC program is given in Figure 4.5. This cyclic sequence of loads activation is for the four-step Skarstrom cycle in the PSA system, which has depressurization time 0.50 sec, purge time 1.43 sec, pressurization and adsorption times combined 2.57 sec. Time precision is 0.01 second.

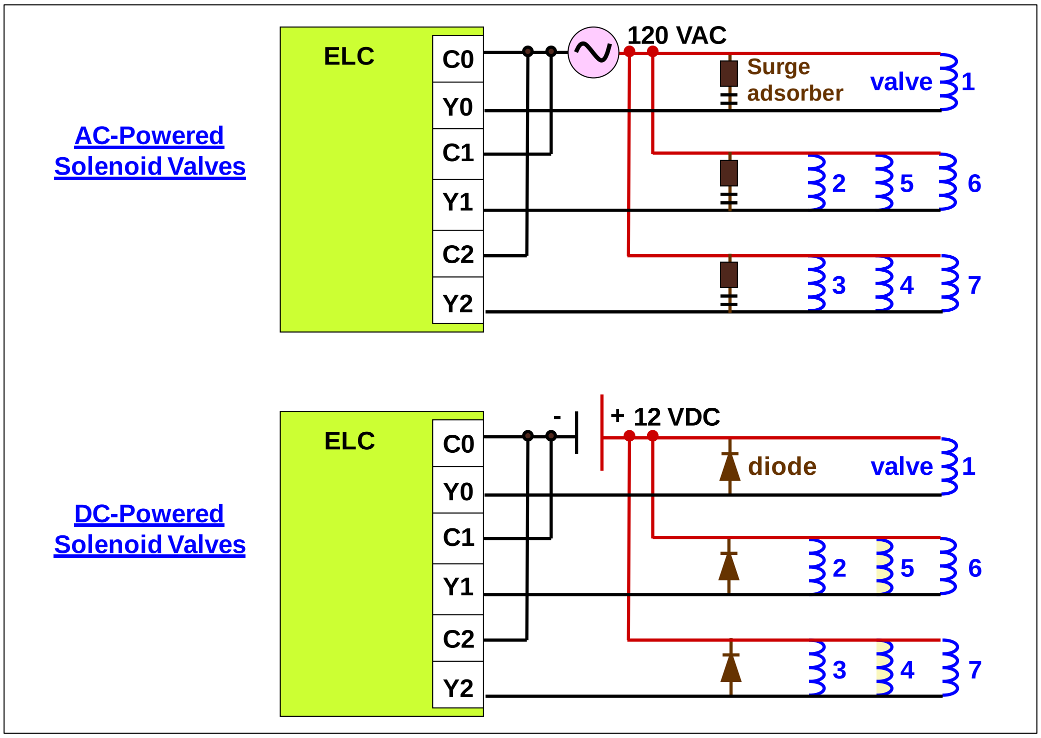


Figure 4.3: Wiring of solenoid valves to PLC relay outputs.

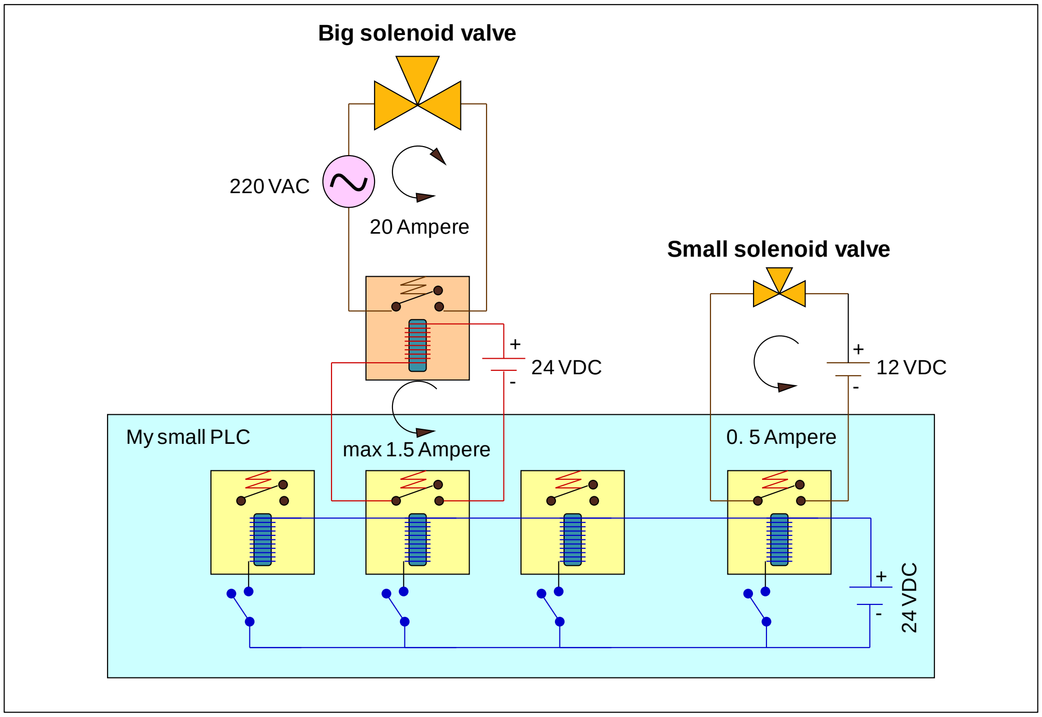
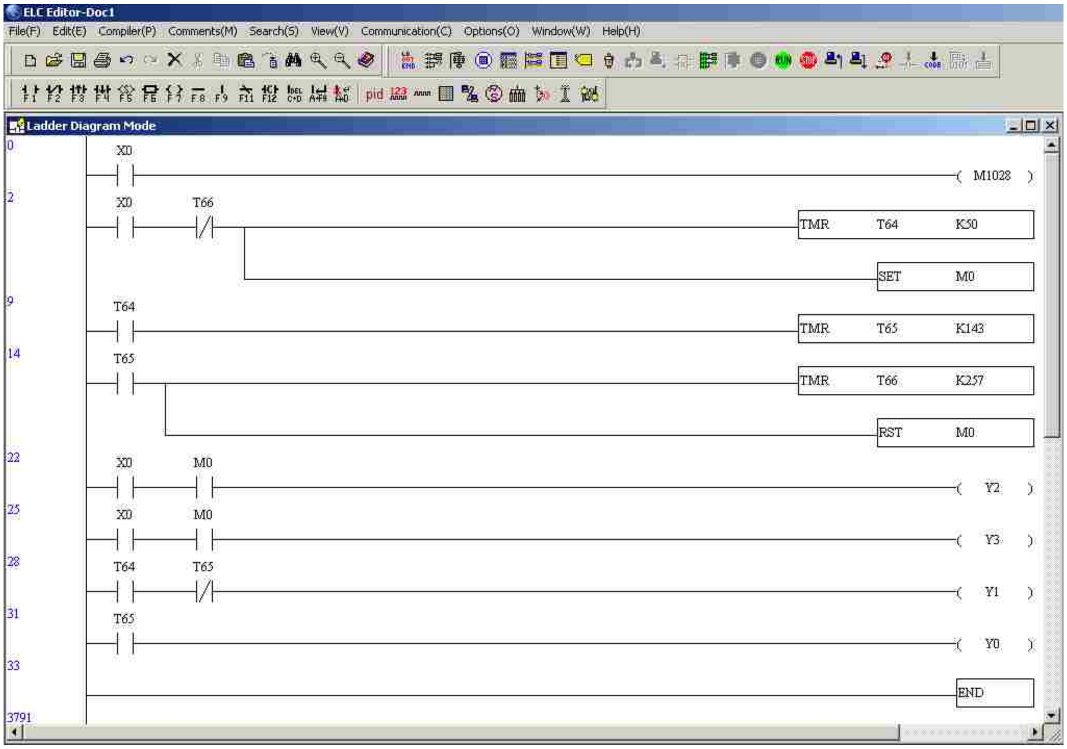
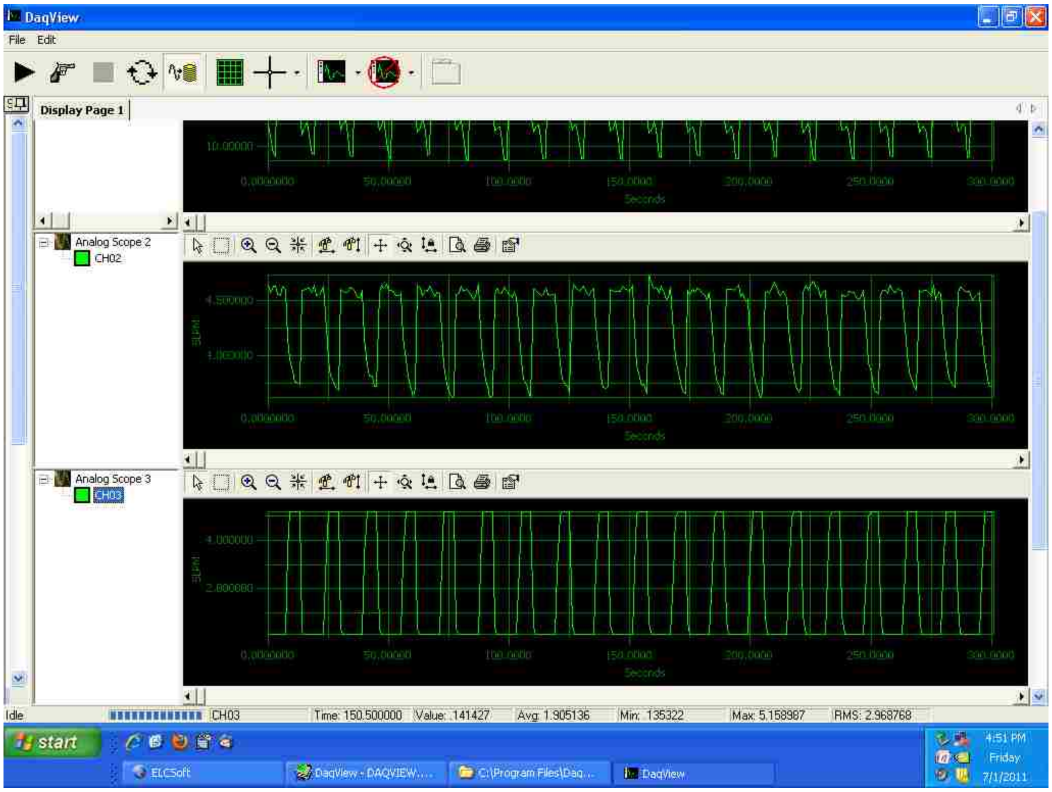


Figure 4.4: Use of intermediate relay to handle large load.


ELC Editor-Doc1
File(F) Edit(E) Compiler(P) Comments(M) Search(S) View(V) Communication(C) Options(O) Window(W) Help(H)
Ladder Diagram Mode

```
0     X0
      -| |------------------------------------------( M1028 )
2     X0      T66
      -| |----|/|--+----------------------------[TMR   T64   K50 ]
                   +----------------------------[SET   M0        ]
9     T64
      -| |--------------------------------------[TMR   T65   K143]
14    T65
      -| |--+-----------------------------------[TMR   T66   K257]
            +-----------------------------------[RST   M0        ]
22    X0      M0
      -| |----| |-------------------------------( Y2 )
25    X0      M0
      -| |----| |-------------------------------( Y3 )
28    T64     T65
      -| |----|/|-------------------------------( Y1 )
31    T65
      -| |--------------------------------------( Y0 )
33
      ------------------------------------------[END]
3791
```

Figure 4.5: A sample of the PSA switching program in the PLC.



```
Idle   CH03   Time: 150.500000   Value: .141427   Avg: 1.905136   Min: .135322   Max: 5.158987   RMS: 2.968768
```

Figure 4.6: A sample of the flow rate data logged by the data acquisition system.

4.2.2 Data Acquisition

OMB-DAQ-300116-bit/1-MHz USB data acquisition system (DAS) from Omega Engineering was used to control and acquire flow rates of mass flow controllers**.** A sample of the controlled and logged data is shown in Figure 4.6. This DAS outputs maximum, minimum, average, root mean square of the data over the time span specified in addition to the dynamic profiles. In the acquisition setup, it allows us to specify scan count and scan rate (number of scans per second).

4.2.3 Void Volume

Void volumes are considered as the empty space inside the system that do not contribute to separation but causing additional compression cost, more product loss during depressurization step, higher product dilution, and more purge gas required for desorption. All these are detrimental to O_2 recovery. Therefore, it is essential to minimize void volume in the system, which can be accomplished by using miniature solenoid valves with small internal volumes, reducing tubing length and diameter, and filling the void space in connectors with glass beads.

Loose bed ends are created as a result of powderized adsorbent particles after many shocks of PSA at high pressures. Therefore, it is necessary that packed bed adsorbent be replaced with fresh adsorbent after some time for consistent characterization.

The relevant void volumes in the apparatus (packed column void + space inside valves located at column ends + connecting tubes) were measured by helium expansion, which is elaborated in Section 4.2.7. Some of the process performance tests were conducted using valves with smaller internal void space and others were conducted using valves with larger internal void space which were powered by higher voltage for reliable switching operation. The specific helium void volumes of the apparatus for the large void (system A) and small void (system B) configurations were, respectively, ~ 6.9 and 6.1 cm^3/g of adsorbent. The estimated void volume of the packed column alone was ~ 0.88 cm^3/g. The adsorbent bulk and chemical densities were, respectively, ~ 0.74 and 2.1 g/cm^3 of adsorbent. Thus, the extraneous void in the test system was high. Consequently, there will be artificially high loss of O_2 from the system during the depressurization step of the PSA cycle, resulting in lower O_2 recovery for the PSA cycle. This is an artifact of the test system.

4.2.4 System Calibrations

SRS RGA mass spectrometer (MS) is a sensitive equipment with its analysis pressure- and temperature-dependent. In order to minimize sampling fluctuation, a balloon is used to collect sample gas up to a controlled volume within a beaker, and the equilibrated mixed gases are fed into sampling suction port of mass spectrometer for gas composition analysis. Adjust pressure regulator at the sampling inlet to get stable measurement. Different colors of balloons are used for collecting product and exhaust gases. Discard balloons which are more than a few days old.

Aalborg Mass flow controllers (MFCs) are recalibrated from time to time, for consistency of these MFCs normally does not last for one week. Do not use MFC for more than 24 hours, especially if there is always a shock or obstruction that hinders controlled flow.

Panasonic miniature pressure sensor (PS) is also calibrated using a standard barometer.

4.2.5 Adsorbent Bed Activation

The miniature tube of diameter 0.4 cm was packed with small adsorbent particles up to 10.8 cm length to give ~ 1.0 gram of LiX (dry weight). Insulator fiber tape was used as filters at both ends. Proper amount of filter is needed to prevent zeolite particles from getting into solenoid valve as well as to avoid introducing unwanted gas flow resistance.

For obtaining accurate dry weight of LiX, it is necessary to remove moisture from zeolite adsorbent prior packing. This step also prevents packed bed shrinkage and leaving much void in the tube post thermal activation. After ~ 5 hours heating on a hot plate in a fume hood, the zeolite adsorbent in beaker is weighted, replace it onto hot plate in the fume hood for heating for another while. Then, pack zeolite into the miniature tube. Measure the remaining beaker + zeolite to obtain the dry weight of zeolite adsorbent in packed bed.

For thermal activation, the packed bed was connected to a minute but continuous flow of bone dry N_2 gas. This N_2 purge flow rate should be controlled at minimum so as to avoid internal cooling but still able to inhibit air and moisture from getting into packed bed. Steadily increase heating from ambient to activation temperature in the first hour. 350^oC < activation temperature < 400^oC is a good range. Below or beyond these temperatures, the bed is not fully activated or particles disintegration might happen in which the binder material turns grey, becomes brittle and eventually powdered. Heating beyond 5 hours does not improve activation.

For consistent comparison, the adsorbent bed was regenerated by heating to ~ 350^oC for ~ 5 hours under dry N_2 purge before all test runs. Repeated adsorbent replacement and activation was also carried out to ensure that the deterioration of performance results, if any, was not due to adsorbent deactivation over time.

4.2.6 Adsorbent Bed Installation

At the end of thermal activation, N_2 purge flow through the packed bed should be increased to allow cooling as well as to prevent adsorption of air and moisture into packed bed at lower temperature. Make sure the activated bed is completely cooled to avoid post installation shrinkage thus leaving some gaps unsealed.

At the experimental system, turn on dry gas flow to purge away air that was in the system, continue its flow during the packed bed installation. Make sure BPR is reduced to lowest pressure. The whole installation process was carried out such that the freshly activated packed bed was totally sealed from air contact. First insert packed bed at the COM port of solenoid valve Y2, let air flows in but discontinue and release the N_2 flow. Then insert another packed bed end into the COM port of solenoid valve Y3. Air continues to flow from solenoid valve Y2 to Y3. Silicon adhesive is applied before (on the thread) and after the nut. See that adhesive spread out behind the nut.

After the installation, dry gas flow was continued for another hour or more to allow initial curing of the silicon adhesive. Then the packed bed can be let alone overnight in slightly pressurized condition for complete solidification of the silicon adhesive.

4.2.7 Helium Leak Test

The system must be made leak proof, or at least, with very minimum leak rate, because the separation is performed at high pressure and gas leakage can cause major loss of product. There are several precautions in the installation stage which can minimize leakage:

Proper connection of 10-32 connector to solenoid valve. Before doing this, make sure the port threads of solenoid valves and connectors are cleaned properly using the rotary steel brush at the machining workshop. This is to ensure proper and smooth locking of 10-32 to solenoid valve. Also make sure that 10-32 end is seal taped (to enlarge the diameter a little so as to create firm grip), apply silicon adhesive at the 10-32 top generously (to make sure solenoid valve and connector is closely and flatly attached). Do not assume that all joints are still intact after some time. Reconnect all fittings to solenoid valve ports with proper amount of silicon adhesive.

At the downstream and upstream of T connectors, apply silicon adhesive generously to all the joints (after cap on). After drying, seal tape the T connectors. Tube connected to back pressure regulator can be made leak proof by applying silicon adhesive and seal tape before and after the nut and at the plastic tubes.

After tubing, thread is used to tie plastic tube and barb connector. Do not overlook the ports of solenoid valves Y1 and Y3, the ones connected to product gas for back purge.

After proper experimental setup but before packed bed installation, perform a leak test by connecting upstream and downstream T connectors, pressurize it to ~ 50 psig and spot for any leakage using soap bubble test. Even with minor leak, soap bubbles are seen obviously at leak spots. During leak test, turn BPR knot to very high pressure so that gas will not secretly leak out from BPR, which might be thought as system leak. Fix the leaks before further installation.

After packed bed installation and an overnight curing of silicon adhesive, pressurize the system with helium gas. The helium leak test compound is framed within the fine dotted lines in Figure 4.7 (from NC of solenoid valve Y0 to BPR). Void volume and helium leak rate are measured by helium expansion and are calculated as follows:

- Packed bed volume (include solid + void) = π x (0.2 cm)^2 x 10.8 cm = 1.357 cm^3
- Prior each leak check, the PSA system was purged by helium gas for some time to expel all adsorbed N_2.

Calculate void volume of the pressurized system:

- Pressurize the helium leak test compound with Helium up to: 344 kPa
- Depressurize it down to: 0 kPa

- Pressure difference = 3.395 atm
- Volume of helium gas collected = 63 mL for 3 cycles at 19.5^{o}C (292.5 K) = 19.6 mL$_{STP}$ helium in the void
- Void volume of the pressurized system = (19.6 mL$_{STP}$ He x (mole He/22400 mL$_{STP}$ He) x 82.05746 cm^{3}.atm/(K.mole)) x 292.5 K / 3.395 atm = 6.186 cm^{3}

Calculate helium leak rate from the pressurized system:

- Pressurize with helium up to: 329 kPa
- After 15 hours, the pressure dropped to: 253 kPa
- Pressure difference = 0.75 atm
- Initial moles of helium in the system void = 329 kPa x void volume/ (R.T)
- Final moles of helium in the system void = 253 kPa x void volume/ (R.T)
- Moles of helium leaked from the system void = (329 – 253) kPa x void volume/(R.T) = 0.75 atm x 6.186 cm^{3} void volume /(82.05746 cm^{3}.atm/mol.K x 292.5 K) = 1.933 x 10^{-4} moles helium leak
- Helium leak rate = 1.933 x 10^{-4} moles helium leak x 22400 mL$_{STP}$ /mole /14 hours = 0.289 mL$_{STP}$ helium leak/hr.

This leak rate is very minute (0.02%) as compared to the production rate of this system, 1167 mL$_{STP}$ product/hr (for BSF = 50 lb/TPD$_{c}$). The leak check was performed for several consequent nights.

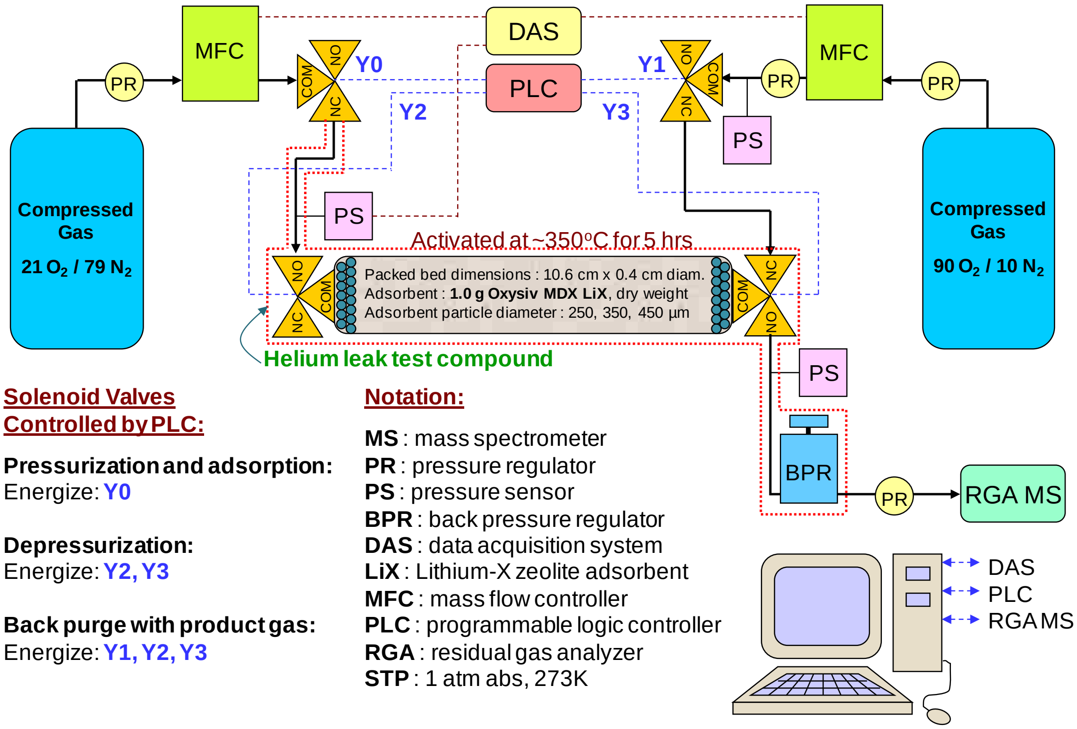

MFC
DAS
MFC
PR
Y0
PLC
Y1
PR
PR
NO
COM
NC
Y2
Y3
NO
COM
NC
PS
PS
Compressed Gas
21 O2 / 79 N2
Activated at ~350°C for 5 hrs
Compressed Gas
90 O2 / 10 N2
NO
COM
NC
Packed bed dimensions : 10.6 cm x 0.4 cm diam.
Adsorbent : 1.0 g Oxysiv MDX LiX, dry weight
Adsorbent particle diameter : 250, 350, 450 µm
NC
COM
NO
Helium leak test compound
PS
Solenoid Valves Controlled by PLC:
Pressurization and adsorption:
Energize: Y0
Depressurization:
Energize: Y2, Y3
Back purge with product gas:
Energize: Y1, Y2, Y3
Notation:
MS : mass spectrometer
PR : pressure regulator
PS : pressure sensor
BPR : back pressure regulator
DAS : data acquisition system
LiX : Lithium-X zeolite adsorbent
MFC : mass flow controller
PLC : programmable logic controller
RGA : residual gas analyzer
STP : 1 atm abs, 273K
BPR
PR
RGA MS
DAS
PLC
RGA MS

Figure 4.7: Perform helium leak test onto experimental system.

4.2.8 Measurements

Since the miniature test apparatus only handles and produces minute amount of fluid, accurate measurements are very important. For gas flow rate, product or exhaust gas was collected using a graduated cylinder for at least 10 cycles and the collected amount averaged. For product purity, gases were collected in a balloon for at least 10 cycles and up to a controlled volume within a beaker, equilibrated for about 5 minutes, then analyzed by mass spectrometer.

Upon reaching cyclic steady-state, and after the measurements described above performed, it is imperative that overall and component mass balances are checked. In this experimental work, data were all closed within 3% error, but only the data with < 2 % error are accepted.

4.3 Experimental Results

The PSA cycle used in the tests was very similar to those of the Skarstrom cycle consisting of:

(a) Adsorber pressurization to a super-ambient pressure (P_A) with air feed,

(b) Flow of compressed air at P_A to produce ~ 90% O_2 product at ~ P_A,

(c) Counter-current depressurization of column to ambient pressure level and rejection of the waste gas,

(d) Counter-current back purge of the column with a synthetic O_2-enriched product gas at near ambient pressure and rejection of the effluent gas,

(e) Repeat cycle from step (a).

Dry and CO_2-free synthetic air (79% N_2 + 21% O_2) from a compressed gas cylinder was used as the air feed source, and a synthetic product gas (90% O_2 + 10% N_2) from a compressed gas cylinder was used as the product purge gas. Cyclic test runs were conducted until steady state operation was achieved using different adsorbent particle sizes (~ 0.25, 0.35 and 0.45 mm), different adsorption pressures (P_A = 2, 3 and 4 atm), and different total cycle times (~ 2 – 10 seconds). The cycle times of steps (c) and (d) were, respectively, 0.50 and 1.43 seconds for the smaller particle tests, and, respectively, 0.10 and 1.40 seconds for the larger particle tests. The times for steps (a) and (b) varied from run to run. All entrance and exit gas flows and compositions were measured, and the BSF and the O_2 recovery were obtained as functions of total cycle times.

4.3.1 Different Particle Sizes

Figure 4.8 compares the plots of BSF vs t_c measured by using different particle sizes and an adsorption pressure of 3 atm. The parameter PR represents the pressure ratio (P_A/P_D) used in the test. The desorption pressure (P_D) was ambient in all experiments. System A (with larger internal void space) was used for this test. Figure 4.9 shows the corresponding plots of R vs t_c.

It may be seen from Figure 4.8 that the BSF decreases with decreasing cycle time when t_c is large, goes through a minimum value, and then rises again as t_c is further reduced. The minimum value of BSF is smaller and it occurs at a lower cycle time for the smaller particle case. This strongly suggests that the adsorption kinetics is the cause of this behavior because the adsorbate mass transfer coefficient for the larger particle is smaller than that for the smaller particle, which impedes the process performance at lower cycle times. However, as particle size is reduced to 0.25 mm, the overall BSF profile is higher compared to those with d_p = 0.35 mm and 0.45 mm. It is apparent that reduced resistance of pore diffusion does not help because other impediments, such as pressure drop and axial dispersion, arise in the system due to very small particle size.

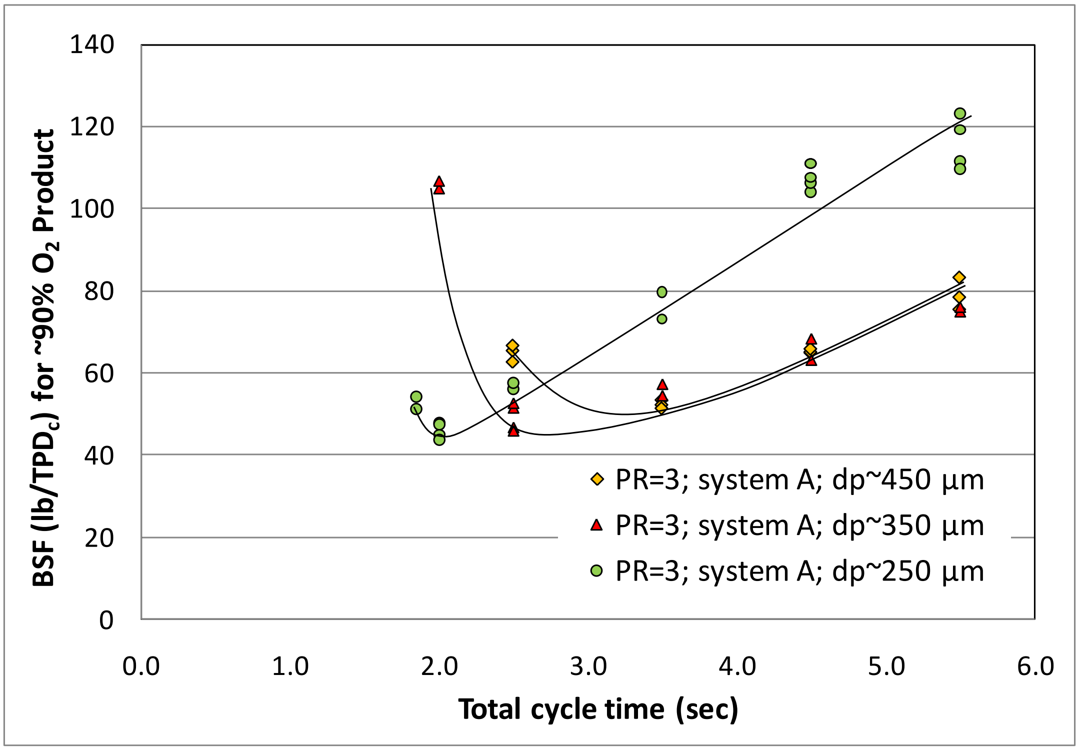


Figure 4.8: BSF vs t_c for different particle sizes.

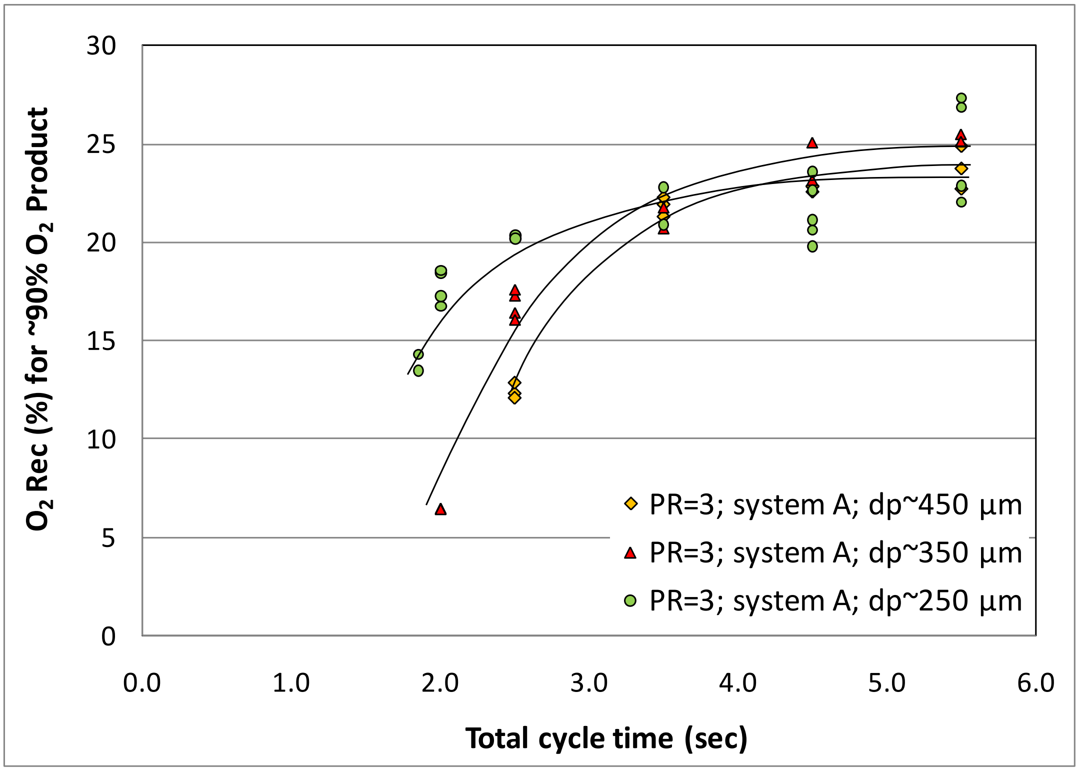


Figure 4.9: R vs t_c for different particle sizes.

Figure 4.9 shows that the functional dependence of O_2 recovery on cycle time is similar for all particle sizes. The recovery is fairly constant at larger cycle times and then it decreases as the cycle time is reduced. It may also be seen from Figures 4.8 and 4.9 that the smaller adsorbent particle produces a smaller value of BSF at minimum and the corresponding O_2 recovery is higher than that for the larger particle. This generally inferior separation efficiency exhibited by the larger particle in the RPSA cycle must be caused by the relatively slower kinetics of sorption on the larger particles.

It should be noted that improved performance cannot be achieved indefinitely by reducing the particle size as demonstrated by these experimental results, this may be due to ineffective cleaning and ineffective pressure swing in a high pressure drop system (Alpay et al. 1994), and that rate-limiting mechanism shifting from pore diffusion to axial dispersion as smaller particles are used had caused pressure drop increasing at a rate disproportionately higher than the gain in mass transfer rate coefficient (Zhong et al. 2010). Therefore, continual decrease of particle size for achieving infinite mass and heat transfer coefficient is not feasible. This clearly demonstrates that BSF cannot be indefinitely reduced by lowering the cycle time of a RPSA process as schematically suggested in Figure 4.1.

It is emphasized that the relatively low values of the O_2 recoveries shown by Figure 4.9 are artifacts of artificially high external void volumes of the test system. The recoveries are expected to be higher if the external void volume of the process system is reduced. However, the general functional dependence of O_2 recovery on total cycle time will be similar to that described by Figure 4.9 irrespective of the relative void volume of the system.

4.3.2 Different Adsorption Pressures

Figures 4.10 and 4.11, respectively, show the effects of adsorption pressure on BSF vs t_c and R vs t_c plots. They were measured using test system B (with smaller internal void space) with adsorbent particle diameter of 0.35 mm. The data in Figure 4.10 show that the BSF decreases with decreasing total cycle time approximately linearly, when t_c > 4 seconds. Thereafter the BSF reaches a minimum value, and finally it starts increasing again when t_c is further reduced. The figure also shows that the BSF for a given t_c decreases as the adsorption pressure is increased. However, the pressure coefficient of BSF decreases as the absolute adsorption pressure is increased above 3 atm. The minimum value of the BSF decreases as the feed gas pressure is increased but the cycle time for the minimum BSF is not affected much by the adsorption pressure. The rise of BSF at lower values of t_c is not clearly evident from the higher adsorption pressure data in Figure 4.10 because data in that region could not be gathered by our experimental system.

Figure 4.11, on the other hand, shows that the O_2 recovery increases as the feed pressure is increased in the range of the data. For a given feed gas pressure, the recovery initially increases rapidly with increasing total cycle time and then the rate of increase diminishes at higher cycle times.

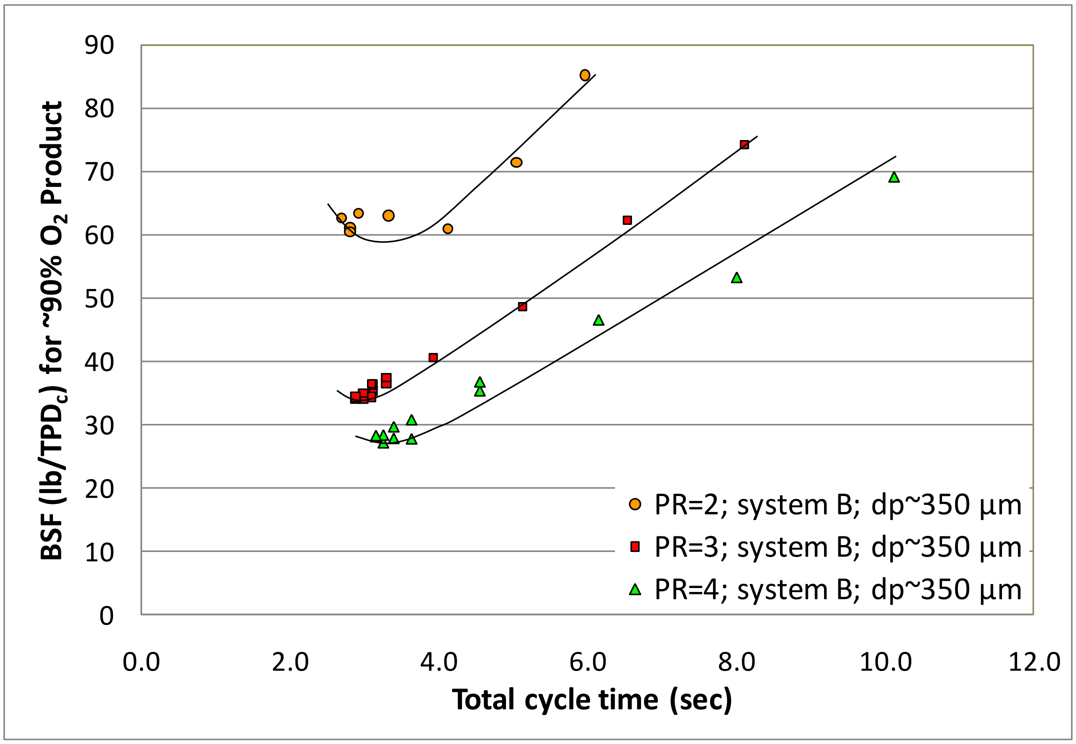


Figure 4.10: Effects of adsorption pressure on BSF vs t_c plots.

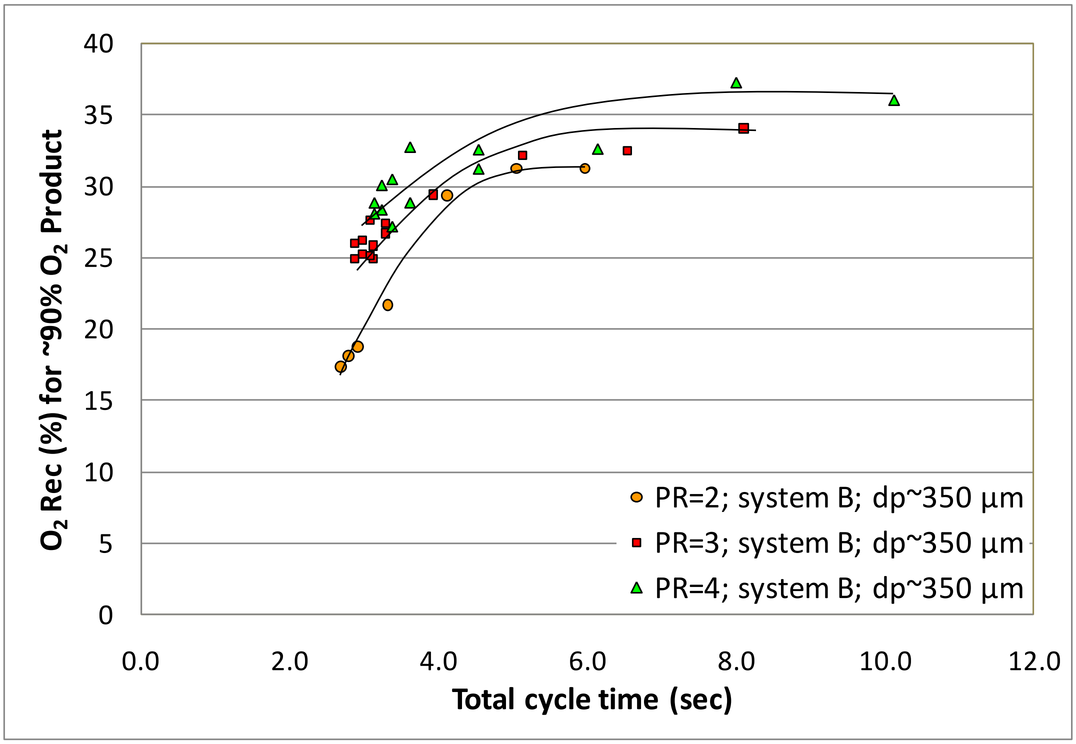


Figure 4.11: Effects of adsorption pressure on R vs t_c plots.

4.3.3 An Optimum Case

A very important conclusion from the data of Figures 4.10 and 4.11 is that the simple PSA process tested in this work is capable of producing ~ 90% O_2 product gas with a BSF of 25 – 50 lbs/TPD_c with an O_2 recovery of at least 25 – 35% by employing LiX zeolite particles of ~ 0.35 mm diameter, adsorption pressure of 3 – 4 atm, and a total cycle time of 3 – 5 seconds.

4.3.4 Compare Performance with Other Oxygen Concentrators

Figures 4.12 and 4.13, respectively, compare the experimental BSF and O_2 recovery as functions of cycle time measured for the PSA process tested in this work (data from Figures 4.10 and 4.11 for d_p = ~ 0.35 mm and PR = 3) with those for other rapid cycle PSA, VSA, and PVSA O_2 concentrators reported in Table 1.3 (Ackley and Zhong 2003; Jagger et al. 2006). All of these processes used LiX zeolite as the sorbent and a PR value of 3.

Figures 4.12 and 4.13 show that the performance of the PSA or PVSA processes of Table 1.3 cannot match the performance reported in this work (BSF < 50 lbs/TPC_c at ~ 30% O_2 recovery) for production of ~ 90% O_2 for MOC application. Only a VSA process, which was developed by Jagger et al. (2006), could better the performance (BSF ~ 30 lbs/TPD_c, O_2 recovery ~ 60%) of the present work, but it would require a very fast cycle ($t_c \leq 1$ second). That would need a large vacuum pump and may impose mechanical operational difficulty. Moreover the potential advantages of operating the MOC in the PSA mode will be lost.

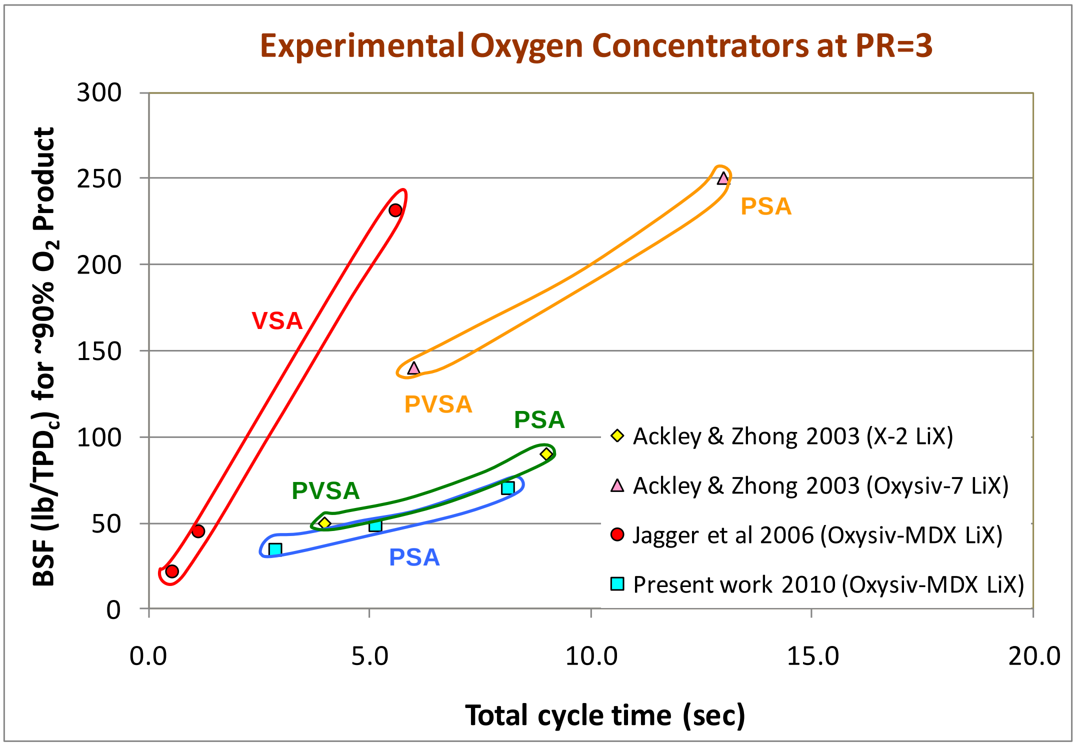


Figure 4.12: Comparative performance of O_2 concentrators (BSF vs t_c).

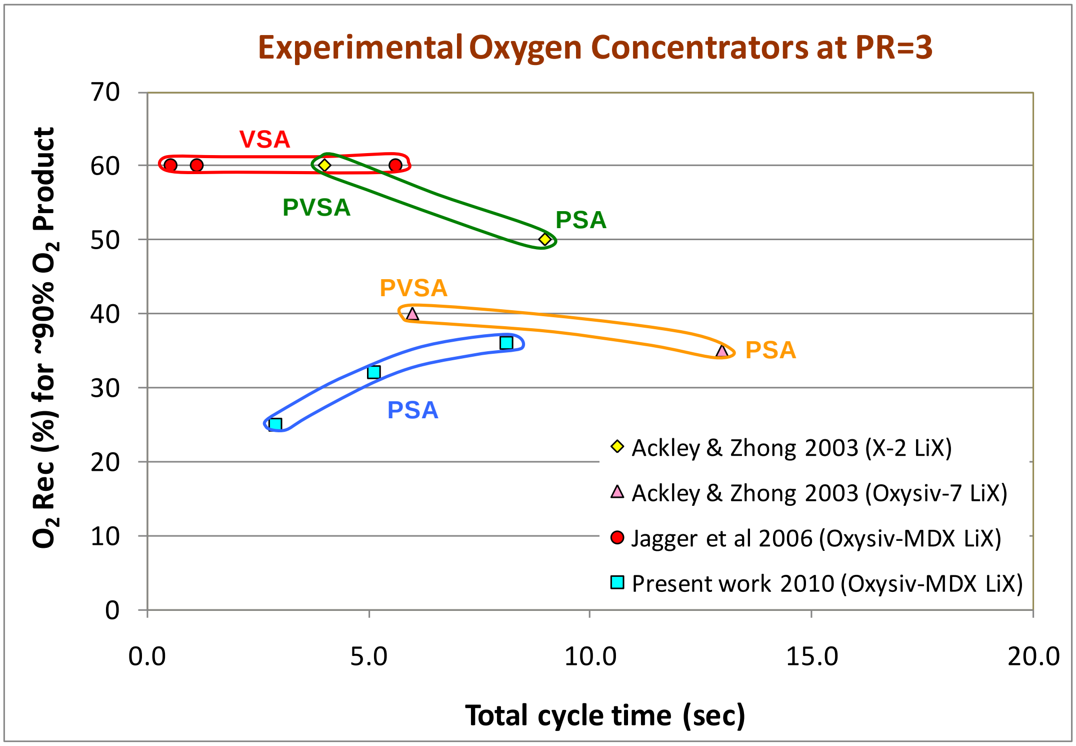


Figure 4.13: Comparative performance of O_2 concentrators (R vs t_c).

4.4 Potential Application

The feasibility of achieving a BSF of 25 – 50 lbs/TPD$_c$ by a simple RPSA system which was experimentally simulated in this work lead us to propose a "snap on" concept for a compact, light-weight, and portable MOC for individual use at locations where piped, compressed air is available, such as in civil and military hospitals, hospitals at remote locations, air craft cabins, cruise ships, trains, public transportations, etc. A schematic drawing of the 'snap on' RPSA unit is given by Figure 4.14. It consists of a single adsorbent column surrounded by a product O_2 tank.

The adsorber of the 'snap on' MOC is packed with a layer (~ 20%) of a desiccant (e.g. activated alumina) and a layer (~ 80%) of LiX zeolite of small particle diameter (0.35 mm). The unit is fitted with one way switch valves and check valves to direct gas flow appropriately during a cycle. Compressed air at 3 – 4 atm is passed through the adsorber to pressurize the adsorber to the feed gas pressure and to produce ~ 90% O_2 product gas to be stored in the tank. The column is then counter-currently depressurized to ambient pressure, and finally purged and partially pressurized with a part of the stored O_2 product gas. A continuous stream of the product gas can be withdrawn for breathing from the storage tank. The process cycle is electronically controlled (not shown in the Figure).

It can be estimated using the RPSA performance data of Figures 4.12 and 4.13 that an adsorber [3.175 cm in diameter and 55 cm long] containing ~ 230 g of LiX zeolite will be adequate to produce ~ 5 LPM of ~ 90% O_2 enriched product gas. The compressed feed air (~ 3 atm) flow rate needed is ~ 67 LPM. The outside diameter of the gas tank is assumed to be ~ 15.2 cm which will be sufficient to hold ~ 15.8 liters of usable O_2 product (withdrawn between 3 and 1 atm pressures).

The configuration of Figure 4.14 is an example of the potential compactness of the ‘Snap On’ concept for a MOC. There will be no dedicated moving machine (compressor, blower or vacuum pump). Locally available compressed air will be fed to the PSA system by snapping on the O_2 concentrator to the air source through a quick connect fitting. Obviously, other system designs and adsorber sizes can be postulated using the data of Figures 4.12 and 4.13.

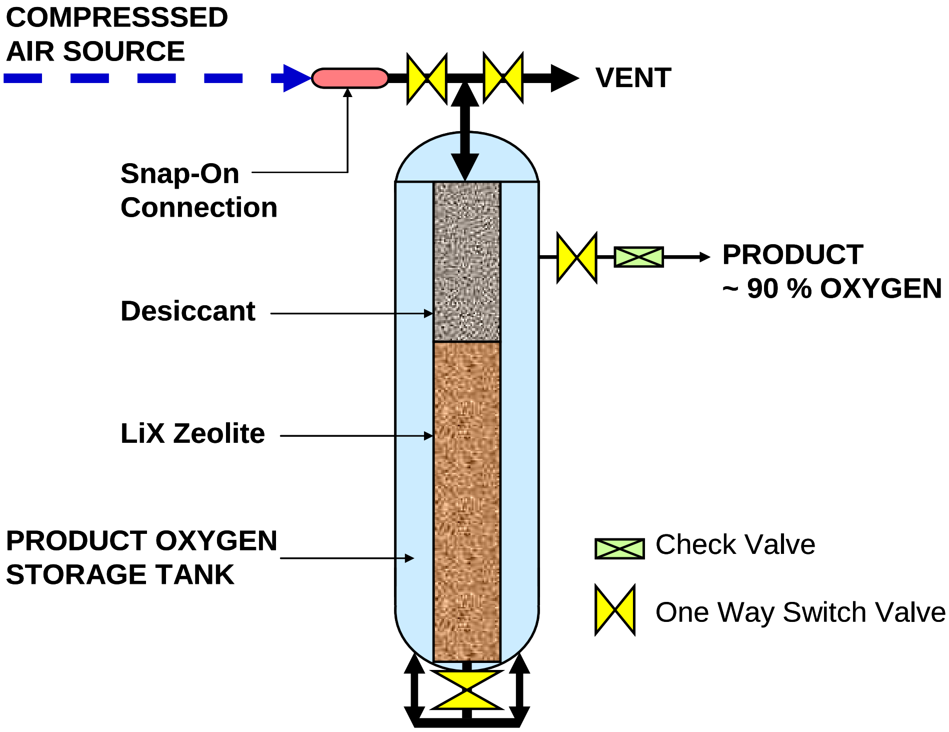


Figure 4.14: Schematic drawing of a ‘Snap On’ MOC.

4.5 Conclusions of Adsorbent Bed Size Reduction

It was experimentally demonstrated using a Skarstrom-like PSA cycle that the bed size factor (BSF) of a PSA process cannot be reduced indefinitely by lowering the total cycle time (t_c) and adsorbent particle size (d_p), due to impediments introduced by adsorbate mass transfer resistance, gas-solid heat transfer resistance, column pressure drop, and axial dispersion, etc. However it was shown that a PSA process can deliver a BSF of 25 – 50 lbs/TPD$_c$ with an O_2 recovery of 25 – 35% using (a) small particles of LiX zeolite (~ 0.35 mm), (b) adsorption pressure of 3 – 4 atm, and (c) total cycle times of 3 – 5 seconds. A novel compact, light-weight and highly portable 'snap on' concept for a medical oxygen concentrator (MOC) is proposed.

4.6 References

Ackley, M.W., Zhong, G.M.: Medical Oxygen Concentrator. U.S. Patent 6,551,384 (2003).

Alpay, E., Kenney, C.N., Scott, D.M.: Adsorbent Particle Size Effects in the Separation of Air by Rapid Pressure Swing Adsorption. *Chem. Eng. Sci.*, **49**, 3059 (1994).

Chai, S.W., Sircar, S., Kothare, M.V.: Miniature Oxygen Concentrators and Methods. U.S. Patent Application 2010/0300285 (2010).

Chai, S.W., Kothare, M.V., Sircar, S.: Rapid Pressure Swing Adsorption for Reduction of Bed Size Factor of a Medical Oxygen Concentrator. *Ind. Eng. Chem. Res.* (2011).

Jagger, T.W., Van Brunt, N.P., Kivisto, J.A., Lonnes, P.B.: Personal Oxygen Concentrator. U.S. Patent 7,121,276 (2006).

Moulijn, J.A., Van Swaaij, W.P.M.: The Correlation of Axial Dispersion Data for Beds of Small Particles. *Chem. Eng. Sci.*, **31**, 845 (1976).

Porter, K.E., Ali, Q.H., Hassan, A.O., Aryan, A.F.: Gas Distribution in Shallow Packed Beds. *Ind. Eng. Chem. Res.*, **32**, 2408 (1993).

Rama Rao, V., Farooq, S., Krantz, W.B.: Design of a Two-Step Pulsed Pressure Swing Adsorption Based Oxygen Concentrator. *AIChE J. Separations*, **56**, 354 (2010).

Sircar, S.: Air Fractionation by Adsorption. *Sep. Sci. Tech.*, **23**, 2379 (1988).

Sircar, S.: Influence of Gas-Solid Heat Transfer on Rapid PSA. *Adsorption*, **11**, 509 (2005).

Zhong, G.M., Rankin. P.J., Ackley, M.W.: High Frequency PSA Process for Gas Separation. U.S. Patent 7,828,878 (2010).

Chapter 5

Mathematical Model

Experimental results of pressure swing adsorption (PSA) system are combination of design, process and operating conditions, which yield overall performance [bed size factor (BSF), O_2 recovery (R)] but the phenomenon at microscopic level are unknown. A mathematical model of PSA system is useful for investigating the process in depth. With such mathematical tool available, the dynamic interaction among pressure, temperature, gas and solid phase concentrations, gas-solid mass and heat transfer kinetics, and molecular gas mixing can be observed, and their effects on ad(de)sorption efficiency can be analyzed. Great flexibility in changing the parameters and fast computation time avoid costly and time consuming experimental testing.

A detailed mathematical model of coupled mass, energy and momentum partial differential equations together with equilibrium and kinetic properties for a PSA process has been developed. This chapter mainly focuses on model development and its conversion from dimensional to non-dimensional model. A chapter section discussing the appropriate use of Ergun equation and momentum balance for adsorption and desorption steps has also been included.

This model will be used for parametric analysis of desorption-by-purge, a regeneration step that prepares the adsorbent column for production in next cycle, also a critical step that determines BSF and O_2 recovery of a PSA process in producing O_2-enriched product gas.

5.1 Model Assumptions

A detailed mathematical model is developed based on the assumptions below:

1. Ideal gas
2. Non-isobaric; Ergun equation describes local pressure drop in the column
3. Non-isothermal; perfect column adiabaticity
4. Non-equilibrium gas-solid adsorption kinetics; linear driving force (LDF) model describes the mass transfer kinetics
5. Non-equilibrium gas-solid heat transfer kinetics; correlations are used to calculate the gas-solid heat transfer coefficients
6. Mixed gas empirical adsorption isotherms describes the adsorption equilibria
7. Include mass and thermal axial dispersions in the gas phase
8. Exclude thermal axial conduction in the solid phase
9. Absence of radial distribution of mass and heat
10. Absence of gas mal-distribution or particle agglomeration

Dimensional mathematical model is first developed and have all its equations checked for uniform units. Then non-dimensional mathematical model is formulated based on the defined dimensionless variables, this model is used in numerical programming and analysis.

5.2 Dimensional Mathematical Model

The mathematical model of PSA system for numerical simulation of the ‘desorption-by-purge’ process is formulated by a set of partial differential equations (PDE) which describe (a) gas phase mass balance for each component (1 = N_2; 2 = O_2), (b) gas phase overall mass balance, (c) adsorbed phase mass balance for each component, (d) gas and solid phase energy balances, and (e) momentum balance or Ergun equation for column pressure drop. They are given by equations (5.1) to (5.7):

Gas phase mass balance for component i:

$$\varepsilon \frac{\partial}{\partial t}\; y_i\rho_g \;\; = -\frac{\partial}{\partial z}\; Qy_i \;\; - \rho_b \frac{\partial n_i}{\partial t} + \varepsilon D_L \frac{\partial^2}{\partial z^2}\; y_i\rho_g \tag{5.1}$$

Gas phase overall mass balance:

$$\varepsilon \frac{\partial}{\partial t}\; \rho_g \;\; = -\frac{\partial}{\partial z}\; Q \;\; - \rho_b \quad \frac{\partial n_i}{\partial t} \tag{5.2}$$

Solid phase mass balance for component i (LDF model):

LDF kinetic model provides a simple but decent representation of ad(de)sorption rate by linear difference between equilibrium loading capacity $n_i^{\infty}(T, P, y_i)$, and dynamic loading $n_i(t, z)$, with the use of effective mass transfer coefficient, k_i, which sums up all the mass transfer resistances.

$$\frac{\partial n_i}{\partial t} = k_i \;\, n_i^{\infty} - n_i \tag{5.3}$$

Gas phase energy balance:

$$\varepsilon C_g \frac{\partial}{\partial t}\; \rho_g\theta_g \;\; = -C_g \frac{\partial}{\partial z}\; Q\theta_g \;\; + ha \;\; T_s - T_g \;\; + \varepsilon C_g D_g \frac{\partial^2}{\partial z^2}\; \rho_g\theta_g \tag{5.4}$$

$$\theta_g(z, t) = (T_g - T_0)$$

Solid phase energy balance:

$$\rho_b C_s \frac{\partial}{\partial t}\ \theta_s\ = \rho_b\quad q_i \frac{\partial n_i}{\partial t} - ha\ \ T_s - T_g\ \ + C_s \rho_b D_s \frac{\partial^2}{\partial z^2}\ \theta_s \tag{5.5}$$

$$\theta_s(z,t) = (T_s - T_0)$$

Momentum balance:

$$\frac{\partial}{\partial t}\ \frac{M_g \rho_g u_s}{\varepsilon}\ = -\frac{\partial}{\partial z}\ \frac{M_g \rho_g u_s\ u_s}{\varepsilon^2}\ \ -\frac{\partial P}{\partial z} - \frac{150\mu}{d_p^2}\frac{1-\varepsilon^{\ 2}}{\varepsilon^3} u_s - \frac{1.75 M_g \rho_g}{d_p}\frac{1-\varepsilon}{\varepsilon^3} u_s\ u_s \tag{5.6}$$

Ergun equation:

$$\frac{\partial P}{\partial z} = -\frac{150\mu}{d_p^2 \rho_g}\frac{1-\varepsilon^{\ 2}}{\varepsilon^3} Q - \frac{1.75 M_g}{d_p \rho_g}\frac{1-\varepsilon}{\varepsilon^3} Q\ Q \tag{5.7}$$

Gas phase component mass balances add up together to give the gas phase density, $\rho_g\ \ z, t\ $. Equation (5.2) is primarily used for getting the overall mass flux, $Q(z,t)$, in an isobaric-isothermal model.

Equation (5.6) is a relatively complete description of the momentum transfer in the system since it includes the transient parts of superficial velocity. However, Ergun equation (5.7) is a relatively simple empirical formulation to be used in place of momentum balance. It models pressure drop versus fluid velocity in packed bed up to the point of fluidization. Fluidization is when packed material's weight equals to the upward drag force from the fluid. Ergun equation relates pressure drop to flow rate, particle properties such as particle packing density, and shape and uniformity of the particle's shape, and fluid properties such as viscosity. Todd and Webley (2005) shows that Ergun equation accurately reproduces dynamic pressure profiles in a packed bed, error of < 0.1% were observed between the Ergun equation and the momentum balance, which justifies the use of equation (5.7) in their model.

Note that $\varepsilon = \varepsilon + (1 - \varepsilon)\ \varepsilon_p$ is the helium or total void fraction in the packed bed with inter-particle void fraction ε and intra-particle void fraction ε_p. It is used in the accumulation terms of gas phase mass and energy balances. The gas amount inside particle void is not negligible and could account for nearly 50% of the total gas amount, which would result to erroneous simulation results if not included in the model (Chahbani and Tondeur 2000). On the other hand, ε is used in gas-phase mass and thermal axial dispersion terms, in momentum balance, and in Ergun equation (Ergun 1952; Hartzog and Sircar 1995; Knaebel). $\varepsilon = 0.33$ is an ideal value for a packed bed of uniform spherical adsorbent, in reality, $0.33 \leq \varepsilon \leq 0.43$ may be used.

The initial and boundary conditions used for numerical solution of the model for desorption-by-purge are summarized below:

$$y_1\ \ z,0\ \ = 0.79;\ y_2\ \ z,0\ \ = 0.21;\ y_1\ \ L,t\ \ = 0;\ y_2\ \ L,t\ \ = 1.0 \tag{5.8}$$

$$P\ \ z,0\ \ = 1\text{ atm};\ P\ \ 0,t\ \ = 1\text{ atm} \tag{5.9}$$

$$\theta_g\ \ z,0\ \ = \theta_s\ \ z,0\ \ = 0;\ \theta_g\ \ L,t\ \ = 0 \tag{5.10}$$

$$\varepsilon C_g \frac{\partial}{\partial t}\ \ \rho_g\theta_g\ \ = -C_g \frac{\partial}{\partial z}\ \ Q\theta_g\ \ + ha\ \ T_s - T_g \qquad \text{at } z = 0 \text{ for all } t \tag{5.11}$$

$$\varepsilon \frac{\partial}{\partial t}\ \ y_i\rho_g\ \ = -\frac{\partial}{\partial z}\ \ Qy_i\ \ - \rho_b \frac{\partial n_i}{\partial t} \qquad \text{at } z = 0 \text{ for all } t \tag{5.12}$$

The last two boundary conditions in equations (5.11) and (5.12) are based on a proposal by Schiesser (1996) for minimum reduction of the PDEs and subsequent stability of the numerical solutions. They were used in place of the conventional Wilhelm-Wehner-Danckwerts boundary conditions $\frac{\partial}{\partial z}\ \ \theta_g\ \ = 0$ and $\frac{\partial}{\partial z}\ \ y_i\ \ = 0$ at the column exit at $z = 0$ for all t which is physically unrealistic.

Other auxiliary equations used in conjunction with equations (5.1) to (5.12) in the PSA model are state variables, given by equations (5.13) to (5.17), and variable-dependent coefficients, given by equations (5.18) to (5.21), and equations (5.23) to (5.25):

Adsorption equilibria:

An empirical model from Rege and Yang (1997) was used for calculating pure and mixed gas equilibrium adsorption isotherms for N_2 and O_2 adsorption on LiX zeolite at different temperatures, given by equation (5.13). When treated strictly as binary Langmuir isotherms, it is in the form of equation (5.14).

$$n_i^{\infty} = \frac{K_i P y_i}{1+ \quad B_j P y_j}; K_i = k_1 exp \;\; \frac{k_2}{T_s} \; ; B_i = k_3 exp \;\; \frac{k_4}{T_s} \qquad (5.13)$$

$$n_i^{\infty} = \frac{m B_i P y_i}{1+ \quad B_j P y_j}; K_i = K_i^o exp \;\; \frac{Q_{Ki}}{RT_s} \; ; B_i = B_i^o exp \;\; \frac{Q_{Bi}}{RT_s} \qquad (5.14)$$

Ideal gas law:

$$P = \rho_g R T_g = \rho_g R(\theta_g + T_0) \qquad (5.15)$$

Overall mass flux and superficial velocity:

$$Q = \rho_g u_s \qquad (5.16)$$

Average gas phase molecular weight:

$$M_g = \quad M_{gi} y_i \qquad (5.17)$$

LDF mass transfer coefficients:

$$k_i = \frac{60 D_i^e}{d_p^2} \qquad (5.18)$$

Effective intra-particle diffusion coefficient:

Effective Knudsen diffusivities of O_2 and N_2, $D_{K,i}^{e}$, effective molecular diffusivity between O_2 and N_2, $D_{m,1-2}^{e}$, on Zeochem LiLSX sorbent were obtained from Todd and Webley (2006). They can otherwise be determined from Knudsen's diffusion law and Chapman-Enskog kinetic theory, respectively (Bird et al. 1960; Ruthven 1984; Yang 1987). Effective intra-particle diffusion coefficient, D_i^e , is described by a parallel combination of molecular and Knudsen diffusion (Alpay, Kenney and Scott 1994) .

$$D_i^e = \frac{\varepsilon_p}{\tau_p} \quad \frac{1}{\frac{1}{D_{K,i}^{e}} + \frac{1}{D_{m,1-2}^{e}}} \tag{5.19}$$

Gas phase mass axial dispersion coefficient:

The contributions of molecular diffusion and turbulent mixing arising from stream splitting and recombination around the sorbent particles can be considered additive (Langer et al. 1978). The first and second terms in equation (5.20) account for, respectively, molecular diffusion with coefficient between 0.64 – 0.73 (Ruthven 1984) and mixing with coefficient 0.5 (Wakao and Funazkri 1978).

$$D_L = 0.7D_{m,1-2}^{e} + \frac{0.5 d_p Q}{\varepsilon \rho_g} \tag{5.20}$$

Gas phase thermal axial dispersion coefficient:

$$D_g = \frac{k_g}{\rho_g C_g} \tag{5.21}$$

Gas-solid heat transfer coefficient:

Model studies have indicated that a finite gas-solid heat transfer resistance can substantially influence the column dynamics of adsorption (Kumar and Sircar

1984a,b) as well as the separation efficiency of a rapid differential PSA system (Sircar 2005). So far, the influence of h on the efficiency of desorption by purge is not studied.

Different empirical correlations between the Nusselt number $Nu = hd_p/k_g$ and the Reynolds number $Re = Qd_pM_g/\mu_g$ for calculation of surface gas-solid heat transfer coefficient, h, are found in the literature. Some account for single particle in fluid flow while others account for packed particles in fluid flow. Some correlations are used together with thermal axial dispersion in the gas-phase. Omission of gas-phase thermal axial dispersion in a packed column can lead to a very low value of Nu in the low Re (< 5) region (Wakao et al. 1979; Dhingra et al. 1984).

Ranz and Marshall (1952) proposed the following correlation for heat transfer of a single sphere on the basis of a variety of experimental data on the evaporation of liquid droplets in air.

$$Nu = 2 + 0.6Pr^{1/3}Re^{1/2} \tag{5.22}$$

where Nu, Pr, and Re are, respectively, the Nusselt, Prandtl, and Reynolds numbers.

The basic idea of Ranz and Marshall correlation is that at low Reynolds number regime the correlation must approach asymptotically the theoretical value for the stationary fluid, whereas in the large Reynolds number regime the flow around the sphere is turbulent. This correlation combines two extreme cases, stationary fluid and very large free stream velocity, thus it encompasses a wide range of Reynolds number with fairly good accuracy.

In Ranz equation, a limiting Nu of 2 is given for gas-solid heat transfer in the absence of convection. However, many experimental data with model fitting at low Re number show that Nu decreases rapidly to zero. This behavior was explained by Kunii and Suzuki (1967) and Martin (1978) as channeling and mal-distribution problem. Gunn and De Souza (1974) explained that behavior is due to exclusion of thermal axial dispersion in model fitting. Nelson and Galloway (1975) explained that behavior is due to there is hardly any heat transport by convection at low Re. Wijngaarden and Westerterp (1993) measured local and global values of gas-solid heat transfer coefficients in packed beds and refuted the hypotheses by Kunii and Suzuki, Martin, and Gunn and De Souza, but supported the hypothesis by Nelson and Galloway.

Littman et al. (1968) presented data for the gas-solid Nu number that takes into account the effects of both axial dispersion of heat in the gas phase and axial conduction in the solid phase. This conduction-dispersion model is an appropriate representation for packed bed heat transfer in the Re number range investigated. The heat transfer coefficients obtained are reliable in the range $6 \leq \text{Re} \leq 99.2$, while for the range $2 \leq \text{Re} \leq 6$, the Nusselt numbers are of the right order of magnitude.

Wakao et al. (1979) corrected the published steady and nonsteady heat transfer data in packed bed for the axial fluid thermal dispersion coefficients proposed by Wakao (1976). The correction for gas phase axial effective dispersion coefficients gives higher mass transfer coefficients particularly at decreased Re number. The reevaluated heat transfer data on the modified dispersion-conduction model are correlated by the analogous form of mass correlation proposed by Wakao and Funazkri (1978).

The following correlations for gas-solid heat transfer coefficients were used in the model simulation of desorption-by-purge step:

Correlation with gas phase thermal axial dispersion (Wakao et al. 1979):

$$Nu = \frac{hd_p}{k_g} = 2.0 + 1.1Pr^{1/3}Re^{0.6} = 2.0 + 1.1 \ {\frac{C_g\mu_g}{k_gM_g}}^{1/3} \ {\frac{Qd_pM_g}{\mu_g}}^{0.6} \qquad (5.23)$$

$15 \leq Re \leq 8500$

Correlation without gas phase thermal axial dispersion (Kunii and Suzuki 1967):

$$Nu = \frac{hd_p}{k_g} = 0.032 \ Pr \times Re \ ^{1.5} = 0.032 \quad \frac{C_g\mu_g}{k_gM_g} \ {\frac{Qd_pM_g}{\mu_g}}^{1.5} \qquad (5.24)$$

Volumetric gas-solid heat transfer coefficient in packed bed:

The dependency of the volumetric heat transfer coefficient on particle size is less clear due to a number of complicating factors which include the non-uniformity of the bed and the uncertainty in estimating available surface area on the particle size per bed volume (Seesee and Thomson 1977). If a sphere is considered, the volumetric heat transfer coefficient can be formulated as

$$ha = h(1 - \varepsilon)\frac{6}{d_p} \qquad (5.25)$$

Sphere surface area $= 4\pi r_p^2 = 4\pi \ {\frac{d_p}{2}}^2 = \pi d_p^2$

Sphere volume $= \frac{4}{3}\pi r_p^3 = \frac{4}{3}\pi \ {\frac{d_p}{2}}^3 = \frac{1}{6}\pi d_p^3$

Sphere surface area/sphere volume $= \frac{6}{d_p}$

Total surface area available for heat transfer/bed volume = (sphere volume/bed volume) × (sphere surface area/ sphere volume) = $a = (1-\varepsilon)\frac{6}{d_p}$

5.2.1 Uniform Units for Dimensional Mathematical Model

Equations (5.1) to (5.7) are checked for their units uniformity in each individual equation and shown below. Similarly, equations (5.8) to (5.25) have also been checked for their units consistency but are not shown here.

Gas phase mass balance for component i:

$$\varepsilon \frac{\partial}{\partial t}\, y_i \rho_g = -\frac{\partial}{\partial z}\, Q y_i - \rho_b \frac{\partial n_i}{\partial t} + \varepsilon D_L \frac{\partial^2}{\partial z^2}\, y_i \rho_g \tag{5.1}$$

$$\frac{\text{cm}^3\text{ total void}}{\text{cm}^3\text{ bed volume}}\ \frac{1}{\text{sec}}\ \frac{\text{gas phase mmole component i}}{\text{gas phase total mmole}}\ \frac{\text{gas phase total mmole}}{\text{cm}^3\text{ total void}}$$

$$= -\ \frac{1}{\text{cm}}\ \frac{\text{gas phase total mmole}}{\text{cm}^2\text{. sec}}\ \frac{\text{gas phase mmole component i}}{\text{gas phase total mmole}}$$

$$-\ \frac{\text{g adsorbent}}{\text{cm}^3\text{ bed volume}}\ \frac{1}{\text{sec}}\ \frac{\text{mmole component i adsorbed}}{\text{g adsorbent}}$$

$$+\ \frac{\text{cm}^3\text{ interstitial void}}{\text{cm}^3\text{ bed volume}}\ \frac{\text{cm}^2}{\text{sec}}\ \frac{1}{\text{cm}^2}\ \frac{\text{gas phase mmole component i}}{\text{gas phase total mmole}}\ \frac{\text{gas phase total mmole}}{\text{cm}^3\text{ total void}}$$

Gas phase overall mass balance:

$$\varepsilon \frac{\partial}{\partial t}\ \rho_g\ = -\frac{\partial}{\partial z}\ Q\ - \rho_b\ \frac{\partial n_i}{\partial t} \qquad (5.2)$$

$$\frac{\text{cm}^3\ \text{total void}}{\text{cm}^3\ \text{bed volume}}\ \frac{1}{\text{sec}}\ \frac{\text{gas phase total mmole}}{\text{cm}^3\ \text{total void}}$$

$$= -\ \frac{1}{\text{cm}}\ \frac{\text{gas phase total mmole}}{\text{cm}^2.\text{sec}}$$

$$-\ \frac{\text{g adsorbent}}{\text{cm}^3\ \text{bed volume}}\ \frac{1}{\text{sec}}\ \frac{\text{total mmole adsorbed}}{\text{g adsorbent}}$$

Solid phase mass balance for component i (LDF model):

$$\frac{\partial n_i}{\partial t} = k_i\ n_i^{\infty} - n_i \qquad (5.3)$$

$$\frac{1}{\text{sec}}\ \frac{\text{mmole component i adsorbed}}{\text{g adsorbent}} = \frac{1}{\text{sec}}\ \frac{\text{mmole component i adsorbed}}{\text{g adsorbent}}$$

Gas phase energy balance:

$$\varepsilon C_g \frac{\partial}{\partial t}\ \rho_g\theta_g\ = -C_g \frac{\partial}{\partial z}\ Q\theta_g\ + ha\ T_s - T_g\ + \varepsilon C_g D_g \frac{\partial^2}{\partial z^2}\ \rho_g\theta_g \qquad (5.4)$$

$$\frac{\text{cm}^3\ \text{total void}}{\text{cm}^3\ \text{bed volume}}\ \frac{\text{gas phase cal}}{\text{gas phase total mmole. K}}\ \frac{1}{\text{sec}}\ \frac{\text{gas phase total mmole}}{\text{cm}^3\ \text{total void}}\ \text{K}$$

$$= -\ \frac{\text{gas phase cal}}{\text{gas phase total mmole. K}}\ \frac{1}{\text{cm}}\ \frac{\text{gas phase total mmole}}{\text{cm}^2\ \text{cross sec area . sec}}\ \text{K}$$

$$+\ \frac{\text{gas} - \text{solid heat transfer cal}}{\text{cm}^3\ \text{bed volume. K. sec}}\ \text{K}$$

$$+\ \frac{\text{cm}^3\ \text{interstitial void}}{\text{cm}^3\ \text{bed volume}}\ \frac{\text{gas phase cal}}{\text{gas phase total mmole. K}}\ \frac{\text{cm}^2}{\text{sec}}\ \frac{1}{\text{cm}^2}\ \frac{\text{gas phase total mmole}}{\text{cm}^3\ \text{total void}}\ \text{K}$$

Solid phase energy balance:

$$\rho_b C_s \frac{\partial}{\partial t}\ \theta_s\ = \rho_b\quad q_i \frac{\partial n_i}{\partial t} - ha\ \ T_s - T_g\ \ + C_s \rho_b D_s \frac{\partial^2}{\partial z^2}\ \theta_s \qquad (5.5)$$

$$\frac{\text{g adsorbent}}{\text{cm}^3\text{ bed volume}}\ \ \frac{\text{solid phase cal}}{\text{g adsorbent. K}}\ \ \frac{1}{\text{sec}}\ \ \text{K}$$

$$= \ \frac{\text{g adsorbent}}{\text{cm}^3\text{ bed volume}}\ \ \frac{\text{isoteric heat adsorption cal}}{\text{mmole adsorbed}}\ \ \frac{1}{\text{sec}}\ \ \frac{\text{mmole adsorbed}}{\text{g adsorbent}}$$

$$-\ \frac{\text{gas} - \text{solid heat transfer cal}}{\text{cm}^3\text{ bed volume. K. sec}}\ \text{K} + \ \frac{\text{solid phase cal}}{\text{g adsorbent. K}}\ \ \frac{\text{g adsorbent}}{\text{cm}^3\text{ bed volume}}\ \ \frac{\text{cm}^2}{\text{sec}}\ \ \frac{1}{\text{cm}^2}\ \ \text{K}$$

Momentum balance:

$$\frac{\partial}{\partial t}\ \frac{M_g \rho_g u_s}{\varepsilon}\ = -\frac{\partial}{\partial z}\ \frac{M_g \rho_g u_s\ u_s}{\varepsilon^2}\ \ - \frac{\partial P}{\partial z} - \frac{150\mu}{d_p^2}\ \frac{1-\varepsilon\ ^{2}}{\varepsilon^3} u_s - \frac{1.75 M_g \rho_g}{d_p}\ \frac{1-\varepsilon}{\varepsilon^3} u_s\ u_s \qquad (5.6)$$

$$\frac{1}{\text{sec}}\ \ \frac{\text{g}}{\text{mmol}}\ \ \frac{\text{mmol}}{\text{cm}^3}\ \ \frac{\text{cm}}{\text{sec}}$$

$$= \ \frac{1}{\text{cm}}\ \ \frac{\text{g}}{\text{mmol}}\ \ \frac{\text{mmol}}{\text{cm}^3}\ \ \frac{\text{cm}}{\text{sec}}\ \ \frac{\text{cm}}{\text{sec}}\ - \ \frac{\text{g}}{\text{cm. sec}^2}\ \ \frac{1}{\text{cm}}$$

$$-\ \frac{\text{g}}{\text{cm. sec}}\ \ \frac{\text{cm}}{\text{sec}}\ \ \frac{1}{\text{cm}^2}\ - \ \frac{\text{g}}{\text{mmol}}\ \ \frac{\text{mmol}}{\text{cm}^3}\ \ \frac{\text{cm}}{\text{sec}}^{\ 2}\ \ \frac{1}{\text{cm}}$$

Ergun equation:

$$\frac{\partial P}{\partial z} = -\frac{150\mu}{d_p^2 \rho_g}\frac{1-\varepsilon\ ^{2}}{\varepsilon^3} Q - \frac{1.75 M_g}{d_p \rho_g}\frac{1-\varepsilon}{\varepsilon^3} Q\ \ Q \qquad (5.7)$$

$$\frac{\text{g}}{\text{cm. sec}^2}\ \ \frac{1}{\text{cm}}\ = -\ \frac{\text{g}}{\text{cm. sec}}\ \ \frac{\text{mmol}}{\text{cm}^2\text{. sec}}\ \ \frac{\text{cm}^3}{\text{mmol}}\ \ \frac{1}{\text{cm}^2}\ - \ \frac{\text{g}}{\text{mmol}}\ \ \frac{\text{mmol}}{\text{cm}^2\text{. sec}}^{\ 2}\ \ \frac{\text{cm}^3}{\text{mmol}}\ \ \frac{1}{\text{cm}}$$

5.3 Non-Dimensional Mathematical Model

Non-dimensional equations (5.26) to (5.39) below and other auxiliary equations developed in Section 5.2 are to be used in numerical model.

Gas phase mass balance for component i:

$$\frac{\partial}{\partial t}\ y_i\rho_g\ = -\frac{\partial}{\partial z}\ \frac{Q}{\varepsilon}\cdot y_i\ - \frac{n_i^\infty - n_i}{\varepsilon} + \frac{\varepsilon D_L t_{feed}}{\varepsilon L^2}\frac{\partial^2}{\partial z^2}\ y_i\rho_g \qquad (5.26)$$

Gas phase overall mass balance:

$$\frac{\partial}{\partial t}\ \rho_g\ = -\frac{\partial}{\partial z}\ \frac{Q}{\varepsilon}\ - \frac{n_1^\infty - n_1}{\varepsilon} - \frac{n_2^\infty - n_2}{\varepsilon} \qquad (5.27)$$

Solid phase mass balance for component i (LDF model):

$$\frac{\partial n_i}{\partial t} = k_i t_{feed}\ n_i^\infty - n_i \qquad (5.28)$$

Gas phase energy balance:

$$\frac{\partial}{\partial t}\ \rho_g\theta_g\ = -\frac{\partial}{\partial z}\ \frac{Q}{\varepsilon}\cdot\theta_g\ + \frac{haL}{\varepsilon C_g Q_{feed}}\ \theta_s - \theta_g\ + \frac{\varepsilon D_g t_{feed}}{\varepsilon L^2}\ \frac{\partial^2}{\partial z^2}\ \rho_g\theta_g \qquad (5.29)$$

$$\theta_g(z,t) = (T_g - T_0)\ /\ T_{max} - T_{min}$$

Solid phase energy balance:

$$\frac{\partial}{\partial t}\ \theta_s\ = \frac{Q_{feed}t_{feed}}{\rho_b C_s L\ T_{max} - T_{min}}\ q_1\ n_1^\infty - n_1\ + q_2\ n_2^\infty - n_2\ \ - \frac{hat_{feed}}{\rho_b C_s}\ \theta_s - \theta_g\ +$$

$$\frac{D_s t_{feed}}{L^2}\frac{\partial^2}{\partial z^2}\ \theta_s \qquad (5.30)$$

$$\theta_s(z,t) = (T_s - T_0)\ /\ T_{max} - T_{min}$$

Momentum balance:

$$\frac{\partial}{\partial t}\ M_g\rho_g u_s\ = -\frac{\partial}{\partial z}\ \frac{u_s}{\varepsilon}\cdot M_g\rho_g u_s\ - \frac{\varepsilon t_{feed}\ P_{max} - P_{min}}{Q_{feed}L}\frac{\partial P}{\partial z} - \frac{150\mu L}{d_p^2 Q_{feed}}\frac{1-\varepsilon\ ^2}{\varepsilon^2}u_s -$$

$$\frac{1.75 M_g\rho_g L}{d_p}\frac{1-\varepsilon}{\varepsilon^2}u_s\ u_s \qquad (5.31)$$

Ergun equation:

$$\frac{\varepsilon t_{feed}\ P_{max}-P_{min}}{Q_{feed}L}\frac{\partial P}{\partial z}=-\frac{150\mu L}{d_p^2\rho_g Q_{feed}}\frac{1-\varepsilon\ ^2}{\varepsilon^2}Q-\frac{1.75M_gL}{d_p\rho_g}\frac{1-\varepsilon}{\varepsilon^2}Q\ Q \tag{5.32}$$

Initial and boundary conditions:

$$y_1\ \ z,0\ \ =0.79;\ y_2\ \ z,0\ \ =0.21;\ y_1\ \ 1,t\ \ =0;\ y_2\ \ 1,t\ \ =1.0 \tag{5.33}$$

$$P\ \ z,0\ \ =\frac{P_0}{P_{max}-P_{min}}=\frac{P_0}{P_{out}};\ P\ \ 0,t\ \ =\frac{P_{out}}{P_{max}-P_{min}}=\frac{P_{out}}{P_{out}}=1 \tag{5.34}$$

$$\theta_g\ \ z,0\ \ =\theta_s\ \ z,0\ \ =0;\ \theta_g\ \ 1,t\ \ =0 \tag{5.35}$$

$$\frac{\partial}{\partial t}\ \rho_g\theta_g\ \ =-\frac{\partial}{\partial z}\ \frac{Q}{\varepsilon}\cdot\theta_g\ \ +\frac{haL}{\varepsilon C_g Q_{feed}}\ \theta_s-\theta_g \qquad \text{at } z=0 \text{ for all } t \tag{5.36}$$

$$\frac{\partial}{\partial t}\ y_i\rho_g\ \ =-\frac{\partial}{\partial z}\ \frac{Q}{\varepsilon}\cdot y_i\ \ -\frac{n_i^{\infty}-n_i}{\varepsilon} \qquad \text{at } z=0 \text{ for all } t \tag{5.37}$$

Ideal gas law:

$$P=\frac{Q_{feed}t_{feed}R}{L\ P_{max}-P_{min}}\rho_g\ \theta_g\ T_{max}-T_{min}\ \ +T_0 \tag{5.38}$$

Overall mass flux and superficial velocity:

$$Q=\rho_g u_s \tag{5.39}$$

5.3.1 Non-Dimensional Variables

Given here are the non-dimensional variables used to formulate the non-dimensional mathematical model above.

$$t = t/t_{feed} \tag{5.40}$$

$$z = z/L \tag{5.41}$$

$$y_1 = y_1 \tag{5.42}$$

$$\rho_g = \rho_g L/\ Q_{feed} t_{feed} \tag{5.43}$$

$$n_1 = n_1 k_1 \rho_b L/Q_{feed} \tag{5.44}$$

$$n_2 = n_2 k_2 \rho_b L/Q_{feed} \tag{5.45}$$

$$n_1^{\infty} = n_1^{\infty} k_1 \rho_b L/Q_{feed} \tag{5.46}$$

$$n_2^{\infty} = n_2^{\infty} k_2 \rho_b L/Q_{feed} \tag{5.47}$$

$$\theta_g = \ \ T_g - T_{ref}\ \ /\ T_{max} - T_{min} \tag{5.48}$$

$$\theta_s = \ \ T_s - T_{ref}\ \ /\ T_{max} - T_{min} \tag{5.49}$$

$$P = P/\ P_{max} - P_{min} \tag{5.50}$$

$$Q = Q/Q_{feed} \tag{5.51}$$

$$u_s = u_s t_{feed}/L \tag{5.52}$$

5.4 Transient Superficial Velocity vs Ergun Equation in Momentum Balance

Todd and Webley (2005), Sereno and Rodrigues (1993) claimed that Ergun equation can be used to estimate pressure drop in packed bed column. Indeed, many simulation works were reported to use Ergun equation or Darcy's law for pressure drop in packed bed, as in Sundaram and Wankat (1988), Lu et al. (1993), Scott (1993), Yang et al. (1998), Chahbani and Tondeur (2001), Ko et al. (2003).

Since numerical Method of Lines (MOL) and the ODE solver in Matlab are used for this model simulation work, an ordinary differential equation is preferred over the empirical Ergun equation in order to benefit from the MOL framework and ODE solver. With reference to Whitaker (1996), Todd and Webley (2005) and Petrov (2006), a momentum equation describing the dynamic superficial velocity due to convection, frictional pressure drop, and ad/desorption was derived, given by equations (5.6) and (5.6b) below. Kikkinides and Yang (1993) and Kupiec et al. (2009) presented a similar momentum balance but they neglected the ad/desorption contribution to the transient superficial velocity.

Momentum balance used for this model is equation (5.6), when expanded gives rise to equation (5.6b).

$$\frac{\partial}{\partial t}\ \frac{M_g\rho_g u_s}{\varepsilon}\ = -\frac{\partial}{\partial z}\ \frac{M_g\rho_g u_s\ u_s}{\varepsilon^2}\ \ -\frac{\partial P}{\partial z}-\frac{150\mu}{d_p^2}\frac{1-\varepsilon\ ^{2}}{\varepsilon^3}u_s-\frac{1.75M_g\rho_g}{d_p}\frac{1-\varepsilon}{\varepsilon^3}u_s\ u_s \qquad (5.6)$$

$$M_g\rho_g\ \ \frac{1}{\varepsilon}\frac{\partial u_s}{\partial t}+\frac{u_s}{\varepsilon^2}\frac{\partial u_s}{\partial z}\ = -\frac{\partial P}{\partial z}-\frac{150\mu}{d_p^2}\frac{1-\varepsilon\ ^{2}}{\varepsilon^3}u_s-\frac{1.75M_g\rho_g}{d_p}\frac{1-\varepsilon}{\varepsilon^3}u_s\ u_s\ +\rho_b\frac{M_g u_s}{\varepsilon^2}\ \ \ \frac{\partial n_i}{\partial t}$$

(5.6b)

If the momentum balance is purely used with positive superficial velocity, for example, during adsorption step, the pair $u_s\ u_s$ is not required but u_s^2 is adequate. However, for desorption step where negative superficial velocity is used, the pair $u_s\ u_s$ in the momentum equation is compulsory to ensure consistent and correct evaluation of the variables.

Table 5.1 tabulates some of the values from the transient superficial velocity (with respect to time and space) and from the Ergun equation (frictional pressure drop) that compose the non-dimensional momentum balance [equation (5.31)] in a simulation case, in which a bed packed with d_p = 300 μm adsorbent and initially saturated with 79% N_2 + 21% O_2 at P_0 = 1 atm and T = 298 K was back purged by 100% O_2 at t_{feed} = 3 sec, Q_{feed} = 3.88752 mmol/cm^2/sec. It is obvious that the values from Ergun equation part are generally five orders of magnitude larger than those from the transient superficial velocity part. Therefore, the transient part in momentum balance is usually dropped off but leaving the empirical steady-state Ergun equation (5.32) to be solved analytically to get the gas mass flux, Q.

Table 5.2, on the other hand, tabulates the numerical values of momentum balance for adsorption step: a bed packed with d_p = 300 μm adsorbent and initially saturated with 100% O_2 at P_0 = 3 atm and T = 298 K was displaced by 79% N_2 + 21% O_2 at t_{feed} = 3 sec, Q_{feed} = 3.88752 mmol/cm^2/sec. It is observed that the values from Ergun equation part are only two orders of magnitude larger than those from the transient superficial velocity part. Therefore, the use of momentum balance for adsorption step might be important for attaining solutions of higher accuracy.

Table 5.1: Compare the contributions of transient superficial velocity and Ergun equation to momentum equation for desorption-by-purge step.

Transient superficial velocity: $\frac{\partial}{\partial t} M_g \rho_g u_s + \frac{\partial}{\partial z} \frac{u_s}{\varepsilon} \cdot M_g \rho_g u_s$	**Ergun equation:** $-\frac{\varepsilon t_{feed}\ P_{max} - P_{min}}{Q_{feed} L} \frac{\partial P}{\partial z} - \frac{150\mu L}{d_p^2 Q_{feed}} \frac{1-\varepsilon\ ^2}{\varepsilon^2} u_s - \frac{1.75 M_g \rho_g L}{d_p} \frac{1-\varepsilon}{\varepsilon^2} u_s\ u_s$
2.180907	10811.73
2.414571	11160.12
-0.21867	11027.29
2.785421	11591.20
0.098667	11499.96
3.206974	12127.32
0.427492	12080.07
3.607463	12771.87

Table 5.2: Compare the contributions of transient superficial velocity and Ergun equation to momentum equation for adsorption step.

Transient superficial velocity: $\frac{\partial}{\partial t} M_g \rho_g u_s + \frac{\partial}{\partial z} \frac{u_s}{\varepsilon} \cdot M_g \rho_g u_s$	**Ergun equation:** $-\frac{\varepsilon t_{feed}\ P_{max} - P_{min}}{Q_{feed} L} \frac{\partial P}{\partial z} - \frac{150\mu L}{d_p^2 Q_{feed}} \frac{1-\varepsilon\ ^2}{\varepsilon^2} u_s - \frac{1.75 M_g \rho_g L}{d_p} \frac{1-\varepsilon}{\varepsilon^2} u_s\ u_s$
743	36045
610	13026
63	2909
646	35865
438	7729
154	3282
467	22915
164	3319
133	4134
96	9041

5.5 Nomenclature

A, bed cross sectional area, cm^2

C_g, gas phase heat capacity, cal/mmol/K

C_s, solid phase heat capacity, cal/g/K

d_p, particle diameter, cm

D, bed diameter, cm

D_i^e, effective diffusivity, cm^2/sec

$D_{K,i}^e$, effective Knudsen diffusivity, cm^2/sec

$D_{m,1-2}^e$, effective Molecular diffusivity, cm^2/sec

$D_g = k_g/\rho_g C_g$, gas phase thermal axial dispersion coefficient, cm^2/sec

D_L, mass axial dispersion coefficient, cm^2/sec

f_{N2}, fraction of initial N_2 (adsorbed and void) desorbed

h, surface gas-solid heat transfer coefficient, cal/cm^2/sec/K

ha, volumetric gas-solid heat transfer coefficient, cal/cm^3/sec/K

k_1, O_2 mass transfer coefficient, 1/sec

k_2, N_2 mass transfer coefficient, 1/sec

k_g, gas phase thermal conductivity, cal/cm/sec/K

L, bed length, cm

M_g, gas phase molecular weight, g/mmol

M_{g1}, O_2 molecular weight, g/mmol

M_{g2}, N_2 molecular weight, g/mmol

n_1, specific amount of O_2 adsorbed on solid phase, mmol/g

n_2, specific amount of N_2 adsorbed on solid phase, mmol/g

n_1^∞, equilibrium specific amount of O_2 adsorbed on solid phase, mmol/g

n_2^∞, equilibrium specific amount of N_2 adsorbed on solid phase, mmol/g

Nu, Nusselt number

$P = P(atm) \times 1013250$, absolute pressure, g/cm/sec^2

P_0, absolute pressure at initial condition, g/cm/sec^2

P_{out}, absolute pressure at bed exit, g/cm/sec^2

$P_{max} = P_{out}$, g/cm/sec^2

$P_{min} = 0$

Pe, Peclet number

Pr, Prandtl number

q_1, O_2 isosteric heat of adsorption, cal/mmol

q_2, N_2 isosteric heat of adsorption, cal/mmol

Q, gas mass flux, mmol/cm^2/sec

Q_{in}, gas mass flux in feed, +ve/-ve value, mmol/cm^2/sec

Q_{feed}, gas mass flux in feed, absolute value, mmol/cm^2/sec

r_p, particle radius, cm

$R = 8.314472 \times 10^4$, gas constant, g.cm^2/K/mmol/sec^2

Re, Reynolds number

t, time, sec

t_{feed}, feed time, sec

T_{feed}, feed temperature, K

T_0, initial temperature, K

$T_{ref} = T_0$

$T_{max} = T_{feed}$

$T_{min} = 273$ K

u_s= $\varepsilon\ u_z$, superficial velocity, cm/sec

u_z, interstitial velocity, cm/sec

y_1, O_2 mole fraction in gas phase

y_2, N_2 mole fraction in gas phase

y_{10}, O_2 mole fraction in gas phase at initial condition

y_{1feed}, O_2 mole fraction in gas phase for feed

z, axial position, cm

ε = $\varepsilon_p(1-\varepsilon) + \ \varepsilon$, total or helium void fraction

ε, inter-particle void fraction

ε_p, intra-particle void fraction

μ, gas phase dynamic viscosity, g/cm/s

ρ_b, adsorbent bulk density, g/cm^3

$\rho_g = P/RT_g$, gas phase density, mmol/cm^3

τ_p, particle tortuosity factor

$\theta_g = T_g - T_{ref}$, gas phase temperature, K

$\theta_s = T_s - T_{ref}$, solid phase temperature, K

5.6 References

Alpay, E., Kenney, C.N., Scott, D.M.: Adsorbent Particle Size Effects in the Separation of Air by Rapid Pressure Swing Adsorption. *Chem. Eng. Sci*.. **49**, 18, 3059-3075 (1994).

Bird, R.B., Steward, W.E., Lightfoot, E.N.: *Transport Phenomena*, Wiley, New York (1960).

Chahbani, M.H., Tondeur. D.: Mass Transfer Kinetics in Pressure Swing Adsorption. *Sep. Pur. Technol*., **20**, 185-196 (2000).

Chahbani, M.H., Tondeur, D.: Pressure Drop in Fixed Bed Adsorbers. *Chem. Eng. J.*, **81**, 23 – 24 (2001).

Cornelio, A.A.: Dynamic Modeling of an Industrial Ethylene Oxide Reactor. *Indian Chemical Engineering*, A, **48**, 3 (2006).

Dhingra, S.C., Gunn, D.J., Narayanan, P. V.: The Analysis of Heat Transfer in Fixed Beds of Particles at Low and Intermediate Reynolds Numbers. *Int. J. Heat Mass Transfer*, **27,** 2377 (1984).

Ergun, S.: Fluid Flow through Packed Columns. *Chem. Eng. Prog*., **48**, 89 (1952).

Gunn, D.J., De Souza, J.F.C.: Heat Transfer and Axial Dispersion in Packed Beds. *Chem. Eng. Sci*., 29, 1363-1371 (1974).

Hartzog D.G., Sircar S.: Sensitivity of PSA Process Performance to Input Variables. *Adsorption*, **1**, 133-151 (1995).

Iliuta, I., Larachi, F.: Fines Deposition Dynamics in Packed Bed Bubble Reactors. *Ind. Eng. Chem. Res*., **42**, 2441-2449 (2003).

Kikkinides, E.S., Yang, R.T.: Effects of Bed Pressure Drop on Isothermal and Adiabatic Adsorber Dynamics. *Chem. Eng. Sci.*, **48** (9), 1545-1555 (1993).

Knaebel, K.S.: A "How To" Guide for Adsorber Design. Adsorption Research, Inc. Dublin, Ohio 43016. http://www.adsorption.com/publications/AdsorberDes2.pdf

Ko, D., Siriwardane, R., Biegler, L.T.: Optimization of a Pressure Swing Adsorption Using Zeolite 13X for CO_2 Sequestration. *Ind. Eng. Eng. Res*., **42**, 339-348 (2003).

Kumar, R., Sircar, S.: Adiabatic Sorption of Bulk Single Adsorbate from an Inert Gas – Effect of Gas-Solid Mass and Heat Transfer Coefficients. *Chem. Eng. Commun.,* **26**, 319 (1984).

Kumar, R., Sircar, S.: Adiabatic Sorption of Dilute Single Adsorbate from an Inert Gas – Effect of Gas-Solid Mass and Heat Transfer Coefficients. *Chem. Eng. Commun.*, **26**, 339 (1984).

Kunii, D., Suzuki, M.: Particle-to-Fluid Heat and Mass Transfer in Packed Beds of Fine Particles. *Int. J. Heat Mass Transfer*, **10**, 845 (1967).

Kupiec, K., Rakoczy, J., Lalik, E.: Modeling of PSA Separation Process including Friction Pressure Drop in Adsorbent Bed. *Chem. Eng. and Processing*, **48**, 1199-1211 (2009).

Langer, G., Roethe, A., Roethe, K.P., Gelbin, D.: Heat and Mass Transfer in Packed Beds—III. Axial Mass Dispersion. *Int. J. Heat and Mass Transfer,* **21**, 751-759 (1978).

Littman, H., Barile, R.G., Pulsifer, A.H.: Gas-Particle Heat Transfer Coefficients in Packed Beds at Low Reynolds Numbers. *Ind. Eng. Chem. Fundamentals*, **7**, 4, 554-561 (1968).

Lu, Z.P., Loureiro, J.M., Rodrigues A.E.: Pressurization and Blowdown of Adsorption Beds – II. Effect of the Momentum and Equilibrium Relations on Isothermal Operation. *Chem. Eng. Sci.*, **48** (9), 1699-1707 (1993).

Martin, H.: Low Peclet Number Particle-to-Fluid Heat and Mass Transfer in Packed Beds. C*hem. Eng. Sci.,***33**, 913-919 (1978).

Nelson, P.A., Galloway, T.R.: Particle-to-Fluid Heat and Mass Transfer in Dense Systems of Fine Particles. *Chem. Eng. Sci*., 30, 1-6 (1975).

Petrov, A.S.: Momentum and Heat Transfer in a Packed Bed. Department of Mechanical and Aerospace Engineering, University of California, San Diego. December 2006. http://courses.ucsd.edu/rherz/mae221a/reports/Petrov_221A_F06.pdf

Rama Rao, V., Farooq, S., Krantz, W.B.: Design of a Two-Step Pulsed Pressure Swing Adsorption Based Oxygen Concentrator. *AIChE J.*, **56**, 2, 354-370 (2010).

Ranz, W.E., Marshall, W.R.: Evaporation from Drops: II. *Chem. Eng. Prog*., **48**: 173-180 (1952).

Rege, S.U., Yang, R.T.: Limits for Air Separation by Adsorption with LiX Zeolite. *Ind. Eng. Chem. Res*., **36**, 12, 5358-5365 (1997).

Ruthven, D.M.: *Principles of Adsorption and Adsorption Processes*, Wiley, New York (1984).

Saucez, P., Schiesser, W.E., Wouwer, A.V.: Upwinding in the Method of Lines. *Mathematics and Computers in Simulation*, **56**, 171-185 (2001).

Schiesser, W.E.: PDE Boundary Conditions from Minimum Reduction of the PDE. *Applied Numerical Mathematics*, **20**, 171 (1996).

Schiesser, W.E., Griffiths, G.W.: *A Compendium of Partial Differential Equation Models: Method of Lines Analysis with Matlab,* Cambridge University Press (2009).

Scott, D.M.: Effects of Bed Pressure Drop on Adsorption and Desorption with Langmuir Isotherms. *Chem. Eng. Sci*. **48**, 17, 3001-3006 (1993).

Seesee, T.A., Thomson, W.J.: Gas-Solid Heat Transfer Coefficients in Beds of Crushed Oil Shale. *Ind. Eng. Chem. Proc. Des. Res*., **16**, 2:243-248 (1977).

Sereno, C., Rodrigues, A.: Can Steady-State Momentum Equations be used in Modeling Pressurization of Adsorption Beds? *Gas Separation & Purification*, **3**, 167-174 (1993).

Sircar, S.: Influence of Gas-Solid Heat Transfer on Rapid PSA. *Adsorption*, **11**, 509-513 (2005).

Sircar, S.: Linear-Driving-Force Model for Non-isothermal Gas Adsorption Kinetics. *J. Chem. Soc., Faraday Trans*., **1**, 79, 785-796 (1983).

Sundaram, N., Wankat, P.C.: Pressure Drop Effects in the Pressurization and Blowdown Steps of Pressure Swing Adsorption. *Chem. Eng. Sci.,* **43** (1), 123-129 (1988).

Todd, R.S., Webley, P.A.: Mass Transfer Models for Rapid Pressure Swing Adsorption Simulation. *AIChE J.,* **52**, 9, 3126-3145 (2006).

Todd, R.S., Webley, P.A.: Pressure Drop in a Packed Bed under Nonadsorbing and Adsorbing Conditions. *Ind. Eng. Chem. Res*., **44**, 7234-7241 (2005).

Wakao, N.: Particle-to-Fluid Transfer Coefficients and Fluid Diffusivities at Low Flow Rate in Packed Beds. *Chem. Eng. Sci*., **31**, 1115-1122 (1976).

Wakao, N., Funazkri, T.: Effect of Fluid Dispersion Coefficients on Particle-to-Fluid Mass Transfer Coefficients in Packed Beds. Correlation of Sherwood Numbers. *Chem. Eng. Sci*., **33**, 1375-1384 (1978).

Wakao, N., Kaguei, S., Funazkri, T.: Effect of Fluid Dispersion Coefficients on Particle-to-Fluid Heat Transfer Coefficients in Packed Beds. Correlation of Nusselt Numbers. *Chem. Eng. Sci.*, **34,** 325 (1979).

Whitaker, S.: The Forchheimer Equation: A Theoretical Development. *Transport in Porous Media*, **25**, 27-61 (1996).

Wijngaarden, R.J., Westerterp, K.R.: Pellet Heat Transfer Coefficients in Packed Beds: Global and Local Values. *Chem. Eng. Technol*. **16**, 363-369 (1993).

Yang, J., Park, M.W., Chang, J.W., Ko, S.M., Lee, C.H.: Effects of Bed Pressure Drop in a PSA Process. *Korean J. Chem. Eng.*, **15** (2), 211- 216 (1998).

Yang, R.T.: *Gas Separation by Adsorption Processes,* Butterworths, Boston (1987).

Chapter 6

Numerical Methods

The rigorous mathematical model developed in Chapter 5 is solved via numerical method of lines (MOL) using the ordinary differential equation (ODE) solver in Matlab. Approximation of convective terms in partial differential equations (PDEs) using linear upwind and center finite differences are derived and discussed. Adsorption using different conditions (model resistances, t_{feed} and d_p) are simulated to compare the solutions of 1st order upwind finite volume method (FVM), 2nd order upwind FMV, and Superbee FMV; it is demonstrated that nonlinear FVM using Superbee flux limiter gives the best steep-fronts-tracking performance.

6.1 Method of Lines and ODE Solver in Matlab

The numerical program is divided into two routines, Method of Lines (MOL) routine and main routine, in which Matlab ode solver for stiff system is used to solve the ordinary differential equations (ODEs) simplified from partial differential equations (PDEs) using MOL (Griffiths and Schiesser 2011; Schiesser and Griffiths 2009; Wouwer, Saucez and Schiesser 2001).

6.1.1 Method of Lines

MOL routine converts PDEs of continuous temporal and spatial domains into ODEs of time domain only by discretizing all the convective and diffusive terms. In the process of converting PDEs to ODEs, MOL routine needs to deal with the boundary conditions (BCs) and any other algebraic equations (AEs) necessary for ODEs formulation. The ODEs at BCs must again be taken care of at the end of MOL routine. The major functions in MOL routine are:

- Assign Y variables into recognizable vectors
- Define boundary conditions
- Establish algebraic equations
- Define initial and boundary conditions for algebraic equations, if required
- Discretize convective terms using flux conserving formulation and Superbee flux limiter
- Formulate complete ODEs
- Take care of BCs again

6.1.2 Ordinary Differential Equation Solver

ODE solver is called in the main routine. It receives ODEs formed by MOL routine, initial conditions (ICs) defined earlier, and time series to perform integration. The outputs from ODE solver are matrices of variables in temporal and space domains, in which the space domain was created by MOL routine. Whatever AEs used in the MOL routine will not be conserved in ODE solver. Solutions from AEs must be reformulated after the ODE solver using the output solutions from ODE solver. The major functions in Main routine are:

- Input of system parameters

- Define spatial domain for MOL, z
- Define time series for integration, t
- Define initial conditions for integration, Y_o
- ODE solver calls MOL routine for ODEs, uses t and Y_o to perform integration
- The output solutions are in the matrix form $Y(t, z)$
- Fix BCs
- If AEs used in the MOL routine are necessary for the solutions, they are reformulated here
- Plot the graphs

6.2 Linear Approximation of Convective Terms

Listed below are some upwind and center finite differences for approximating convective terms in ODEs.

- 1st order (two points) upwind finite difference
- 2nd order (three points) upwind finite difference
- 3rd order (four points) upwind finite difference
- 4th order (five points) center finite difference

With reference to Saucez et al. (2001), Schiesser and Griffiths (2009) and Lin (1997), upwind finite differences in Sections 6.2.1 and 6.2.2 are derived according to the direction of information propagation, while center finite differences in Section 6.2.3 are not directional dependent. Note that they are not finite volume methods.

6.2.1 For Propagation of Information in Positive Direction

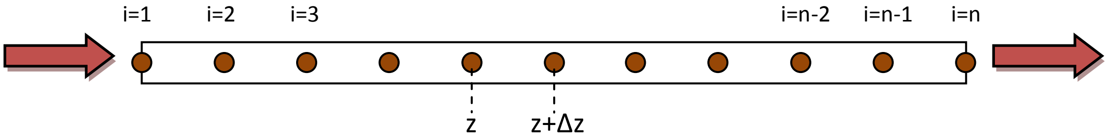


Figure 6.1: Information propagation in positive direction.

1st Order Upwind Finite Difference:

$$\frac{dY(i)}{dz} = \frac{Y\ \ i\ \ -Y\ \ i-1}{dz}$$

$$\frac{dY(1)}{dz} = \frac{Y\ \ 2\ \ -Y\ \ 1}{dz}$$

2nd Order Upwind Finite Difference:

$$\frac{dY(i)}{dz} = \frac{Y\ \ i-2\ \ -4Y\ \ i-1\ \ +3Y\ \ i}{2!\,dz}$$

$$\frac{dY(1)}{dz} = \frac{-3Y\ \ 1\ \ +4Y\ \ 2\ \ -Y\ \ 3}{2!\,dz}$$

$$\frac{dY(2)}{dz} = \frac{-Y\ \ 1\ \ +0Y\ \ 2\ \ +Y\ \ 3}{2!\,dz}$$

3rd Order Upwind Finite Difference:

$$\frac{dY(i)}{dz} = \frac{-2Y\ \ i-3\ \ +9Y\ \ i-2\ \ -18Y\ \ i-1\ \ +11Y\ \ i}{3!\,dz}$$

$$\frac{dY(1)}{dz} = \frac{-11Y\ \ 1\ \ +18Y\ \ 2\ \ -9Y\ \ 3\ \ +2Y\ \ 4}{3!\,dz}$$

$$\frac{dY(2)}{dz} = \frac{-2Y\ \ 1\ \ -3Y\ \ 2\ \ +6Y\ \ 3\ \ -Y\ \ 4}{3!\,dz}$$

$$\frac{dY(3)}{dz} = \frac{Y\ \ 1\ \ -6Y\ \ 2\ \ +3Y\ \ 3\ \ +2Y\ \ 4}{3!\,dz}$$

6.2.2 For Propagation of Information in Negative Direction

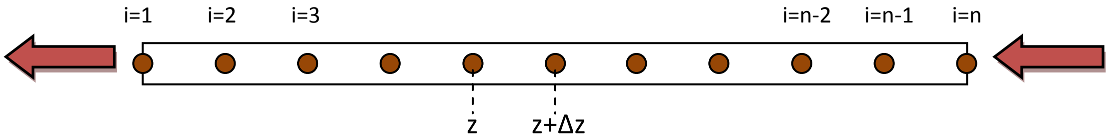


Figure 6.2: Information propagation in negative direction.

1st Order Upwind Finite Difference:

$$\frac{dY(i)}{dz} = \frac{Y\ \ i+1\ \ -Y\ \ i}{dz}$$

$$\frac{dY(n)}{dz} = \frac{Y\ \ n\ \ -Y\ \ n-1}{dz}$$

2nd Order Upwind Finite Difference:

$$\frac{dY(i)}{dz} = \frac{-3Y\ \ i\ \ +4Y\ \ i+1\ \ -Y\ \ i+2}{2!\,dz}$$

$$\frac{dY(n)}{dz} = \frac{Y\ \ n-2\ \ -4Y\ \ n-1\ \ +3Y\ \ n}{2!\,dz}$$

$$\frac{dY(n-1)}{dz} = \frac{-Y\ \ n-2\ \ +0Y\ \ n-1\ \ +Y\ \ n}{2!\,dz}$$

3rd Order Upwind Finite Difference:

$$\frac{dY(i)}{dz} = \frac{-11Y\ \ i\ \ +18Y\ \ i+1\ \ -9Y\ \ i+2\ \ +2Y\ \ i+3}{3!\,dz}$$

$$\frac{dY(n)}{dz} = \frac{-2Y\ \ n-3\ \ +9Y\ \ n-2\ \ -18Y\ \ n-1\ \ +11Y\ \ n}{3!\,dz}$$

$$\frac{dY(n-1)}{dz} = \frac{Y\ \ n-3\ \ -6Y\ \ n-2\ \ +3Y\ \ n-1\ \ +2Y\ \ n}{3!\,dz}$$

$$\frac{dY(n-2)}{dz} = \frac{-2Y\ \ n-3\ \ -3Y\ \ n-2\ \ +6Y\ \ n-1\ \ -Y\ \ n}{3!\,dz}$$

6.2.3 For Propagation of Information in Positive or Negative Direction

4th Order Center Finite Difference:

$$\frac{dY(i)}{dz} = \frac{Y\ \ i-2\ \ -8Y\ \ i-1\ \ +0Y\ \ i\ \ +8Y\ \ i+1\ \ -Y\ \ i+2}{12dz}$$

$$\frac{dY(1)}{dz} = \frac{-25Y\ \ 1\ \ +48Y\ \ 2\ \ -36Y\ \ 3\ \ +16Y\ \ 4\ \ -3Y\ \ 5}{12dz}$$

$$\frac{dY(2)}{dz} = \frac{-3Y\ \ 1\ \ -10Y\ \ 2\ \ +18Y\ \ 3\ \ -6Y\ \ 4\ \ +Y\ \ 5}{12dz}$$

$$\frac{dY(n-1)}{dz} = \frac{-Y\ \ n-4\ \ +6Y\ \ n-3\ \ -18Y\ \ n-2\ \ +10Y\ \ n-1\ \ +3Y\ \ n}{12dz}$$

$$\frac{dY(n)}{dz} = \frac{3Y\ \ n-4\ \ -16Y\ \ n-3\ \ +36Y\ \ n-2\ \ -48Y\ \ n-1\ \ +25Y\ \ n}{12dz}$$

Upwind finite differences are used for numerical approximation of convective terms from the available upwind information. If the propagation of information is not made in the correct direction, numerical stability suffers. On the other hand, center finite difference can be used to approximate diffusive terms which are not directional sensitive.

Though 2nd and 3rd order upwind finite differences reduce numerical diffusion generated by 1st order upwind difference, they carry inherit problem of numerical oscillation. Disregard the upwind formulation, 2nd and 3rd order upwind finite differences produce oscillated numerical solutions due to the mix of upwind and downwind at boundary points. Likewise, 4th order center finite difference produces oscillated numerical solutions. Nonetheless, 2nd and higher order upwind finite difference schemes produce spurious oscillations which remain bounded compared to those generated by the unstable centered-differencing scheme of the same order. Numerical oscillation and instability problem is more rigorous for longer step time due to increased error accumulation. When the nonphysical oscillation becomes too severe, or numerical values generated are out of the integration stability, the model simulation becomes faulty and stops running.

Therefore it is desired to have high order scheme that does not produce spurious oscillations. But according to Godunov Theorem (Godunov 1954, 1959), "*Linear numerical schemes for solving partial differential equations (PDE's), having the property of not generating new extrema (monotone scheme), can be at most first-order accurate."* Henceforth, the solution method is switched to nonlinear approximation scheme, which is introduced in the next section.

6.3 Nonlinear Approximation of Convective Terms

Previous section discussed that first order upwind finite difference is the least accurate approximation method while higher order finite differences produce unrealistic oscillatory numerical solutions. A nonlinear approximation tool that combines higher order accuracy and non-oscillating solutions is desired. This useful tool will allow the use of high order approximation at smooth solutions while limiting its use at high gradient regions to prevent overshoots and excessive oscillation.

6.3.1 2nd Order TVD Superbee Flux Limiter

Finite volume method using flux conserving scheme is introduced here using the course material from Dullemond (2009). The idea of flux conserving creates cells out of the grid, where the cell interfaces are located at

$$z_{i+\frac{1}{2}} = \frac{1}{2}(z_i + z_{i+1})$$

A hyperbolic partial differential equation

$$\frac{\partial Y}{\partial t} = -\frac{\partial (uY)}{\partial z}$$

is formulated into flux conserving equation

$$\frac{\partial Y}{\partial t} = -\frac{\partial (F)}{\partial z}$$

$$\frac{Y_i^{m+1} - Y_i^m}{\Delta t} = -\frac{F_{i+\frac{1}{2}}^{m+\frac{1}{2}} - F_{i-\frac{1}{2}}^{m+\frac{1}{2}}}{\Delta z} = \frac{F_{i-\frac{1}{2}}^{m+\frac{1}{2}} - F_{i+\frac{1}{2}}^{m+\frac{1}{2}}}{\Delta z}$$

which is represented in the figure below, that the change of a conserved quantity Y_i from time *m* to *m+1* is caused by flux in $F_{i-\frac{1}{2}}$ and flux out $F_{i+\frac{1}{2}}$.

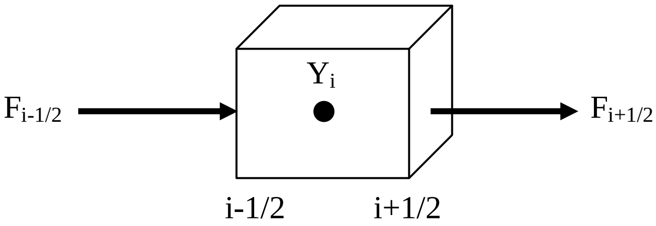


Figure 6.3: Flux conservation in finite volume.

These fluxes are formulated by high and low resolution schemes so that the flux limiter, $\emptyset(r)$, can switch between these schemes depending on the gradients close to the particular cell, as follow.

$$F_{i-\frac{1}{2}} = f^{low}_{i-\frac{1}{2}} - \emptyset(r_{i-\frac{1}{2}}) \quad f^{low}_{i-\frac{1}{2}} - f^{high}_{i-\frac{1}{2}}$$

$$F_{i+\frac{1}{2}} = f^{low}_{i+\frac{1}{2}} - \emptyset(r_{i+\frac{1}{2}}) \quad f^{low}_{i+\frac{1}{2}} - f^{high}_{i+\frac{1}{2}}$$

The fluxes below are formed in terms of the convective average velocity, u, and the conserved quantity, Y.

$$F^{m+\frac{1}{2}}_{i-\frac{1}{2}} = \frac{1}{2} u_{i-\frac{1}{2}} \quad 1 + \theta_{i-\frac{1}{2}} \;\; Y^m_{i-1} + \quad 1 + \theta_{i-\frac{1}{2}} \;\; Y^m_i$$

$$+ \frac{1}{2} \; u_{i-\frac{1}{2}} \quad 1 - \; u_{i-\frac{1}{2}} \frac{\Delta t}{\Delta z} \quad \emptyset(r^m_{i-\frac{1}{2}}) \; Y^m_i - Y^m_{i-1}$$

$$F^{m+\frac{1}{2}}_{i+\frac{1}{2}} = \frac{1}{2} u_{i+\frac{1}{2}} \quad 1 + \theta_{i+\frac{1}{2}} \;\; Y^m_i + \quad 1 + \theta_{i+\frac{1}{2}} \;\; Y^m_{i+1}$$

$$+ \frac{1}{2} \; u_{i+\frac{1}{2}} \quad 1 - \; u_{i+\frac{1}{2}} \frac{\Delta t}{\Delta z} \quad \emptyset(r^m_{i+\frac{1}{2}}) \; Y^m_{i+1} - Y^m_i$$

The fluxes formulated above account for regular or non-regular space grid, Δz, and for varying average velocity, u, in positive and negative convective direction. Direction sense is captured by the flip-flop function, θ, represented by

$$\theta_{i-\frac{1}{2}} = \begin{matrix} +1 & for\, u_{i-\frac{1}{2}} \geq 0 \\ -1 & for\, u_{i-\frac{1}{2}} \leq 0 \end{matrix}$$

$$\theta_{i+\frac{1}{2}} = \begin{matrix} +1 & for\, u_{i+\frac{1}{2}} \geq 0 \\ -1 & for\, u_{i+\frac{1}{2}} \leq 0 \end{matrix}$$

r is a smoothness indicator near the cell interface and is represented by the successive gradients of solution

$$r^{m}_{i-\frac{1}{2}} = \begin{array}{ll} \dfrac{Y^{m}_{i-1} - Y^{m}_{i-2}}{Y^{m}_{i} - Y^{m}_{i-1}} & \text{for } u_{i-\frac{1}{2}} \geq 0 \\ \dfrac{Y^{m}_{i+1} - Y^{m}_{i}}{Y^{m}_{i} - Y^{m}_{i-1}} & \text{for } u_{i-\frac{1}{2}} \leq 0 \end{array}$$

$$r^{m}_{i+\frac{1}{2}} = \begin{array}{ll} \dfrac{Y^{m}_{i} - Y^{m}_{i-1}}{Y^{m}_{i+1} - Y^{m}_{i}} & \text{for } u_{i+\frac{1}{2}} \geq 0 \\ \dfrac{Y^{m}_{i+2} - Y^{m}_{i+1}}{Y^{m}_{i+1} - Y^{m}_{i}} & \text{for } u_{i+\frac{1}{2}} \leq 0 \end{array}$$

Note that interface average velocities, $u_{i-\frac{1}{2}}$ and $u_{i+\frac{1}{2}}$, should be computed beforehand by linear interpolation of cell-centered values u_i. Finally, r serves as an input parameter in Superbee function to determine the flux limiting value.

$$\emptyset \quad r^{m}_{i-\frac{1}{2}} \quad = max \quad 0, \min \quad 1, 2r^{m}_{i-\frac{1}{2}} \quad , \min \quad 2, r^{m}_{i-\frac{1}{2}}$$

$$\emptyset \quad r^{m}_{i+\frac{1}{2}} \quad = max \quad 0, \min \quad 1, 2r^{m}_{i+\frac{1}{2}} \quad , \min \quad 2, r^{m}_{i+\frac{1}{2}}$$

Superbee flux limiter is 2$^{\text{nd}}$ order Total Variance Diminishing (TVD). It is 2$^{\text{nd}}$ order accurate and it guarantees stability of the scheme by passing through the TVD region. For further information about flux limiters please refer to Dullemond (2009), Griffiths and Schiesser (2011), Wouwer et. al. (2001), Saucez et. al. (2001), Toro (1999), Alhumaizi (2007).

6.4 Finite Volume Methods with and without Superbee Flux Limiter

To demonstrate the solution differences using finite volume methods (FVM) for 1st order upwind, 2nd order upwind, and nonlinear approximation using Superbee flux limiter, adsorption step is simulated because it presents clear steep fronts.

A LiX column with dimensions L = 8.382 cm, D = 6.0325 cm, A = 28.5815 cm^2, packed with adsorbent particle size d_p = 200 and 400 μm, is initially saturated with 100% O_2 at P_o = 3 atm. It is then fed with 79% N_2 + 21% O_2 at constant gas fluxes of Q_o = 4.3735 and 0.8747 mmole/cm^2/s, respectively, for feed durations t_{feed} = 2 and 10 seconds, at inlet temperature T = 298 K. The output pressure is P_{out} = 3 atm. The detailed mathematical model based on all the assumptions in Section 5.1 (named as model F), and the isobaric-isothermal model with non-equilibrium adsorption kinetic (named as model B), are used in this simulation. Number of spatial nodes used is n = 101.

Figures 6.4 – 6.9 show the adsorption fronts for dimensionless time t = t/t_{feed} = 0, 0.2, 0.4, 0.6, 0.8, 1.0. It is shown that cases with 1st order upwind FVM generated diffusive steep fronts, cases with 2nd order upwind FVM generated oscillatory steep fronts, and cases with nonlinear FVM with Superbee flux limiter generated non-dispersed and bounded steep fronts. The third numerical method gives the best solution because it can switch between high and low order of flux conserved approximations depending on the smoothness/steepness of the immediate solutions.

It is also shown that when model resistances are kept minimum, as given in Figures 6.4 and 6.7, respectively for, d_p = 200 µm and 400 µm, the adsorption fronts are steep, thus oscillations are obvious using 2nd order upwind FVM. When model resistances have increased, as given in Figures 6.5 and 6.8, respectively for, d_p = 200 µm and 400 µm, the adsorption fronts are more dispersive, thus oscillations are less obvious using 2nd order upwind FVM. Note also that increased model resistances result in earlier breakthrough of adsorption column capacity.

Furthermore, when adsorption times are reduced from 10 sec to 2 sec (thus the increased of air flux, Q_o, from 0.8747 mmole/cm^2/s to 4.3735 mmole/cm^2/s), as given in Figures 6.6 and 6.9, respectively for, d_p = 200 µm and 400 µm, the effects of resistances are magnified in these rapid adsorption steps. As a result, more dispersive adsorption fronts are generated and the oscillatory phenomenon of 2nd order upwind FVM is furthered dampened. The oscillation phenomenon in Figure 6.9 essentially diminished.

The adsorption using d_p = 400 µm creates more dispersive fronts than those of 200 µm because smaller particle size gives higher pore diffusion rates, thus mass transfer zones are narrowed with the absence of mass axial dispersion in these cases.

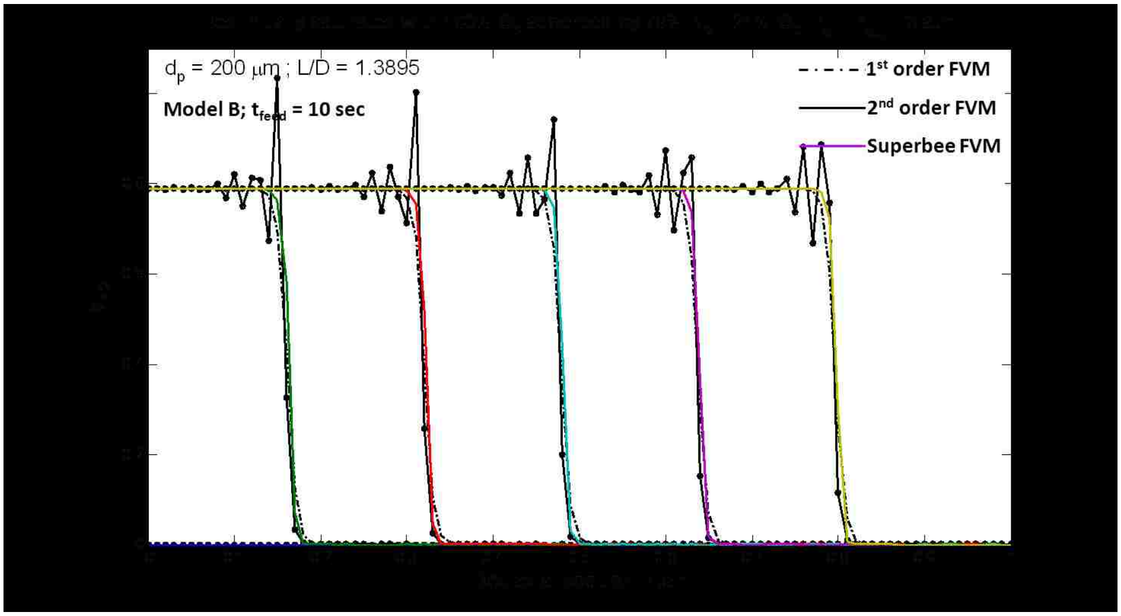


Figure 6.4: Adsorption steep fronts for model B (t_{feed} = 10 sec, d_p = 200 μm), using 1st order upwind, 2nd order upwind, and Superbee finite volume methods, with CPU times = 20, 21, 22 min, respectively.

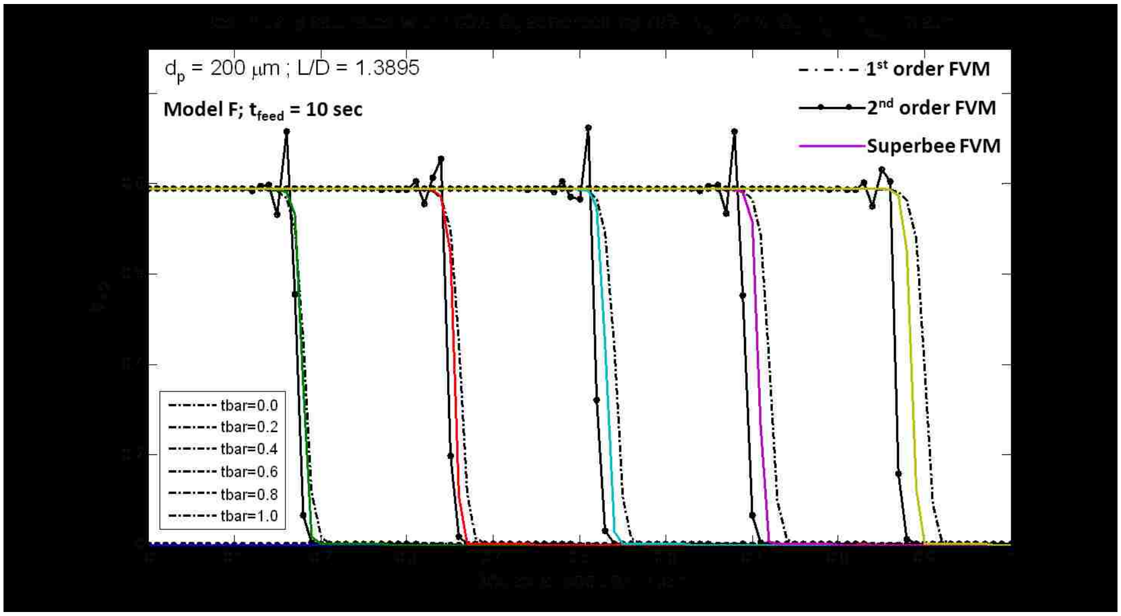


Figure 6.5: Adsorption steep fronts for model F (t_{feed} = 10 sec, d_p = 200 μm), using 1st order upwind, 2nd order upwind, and Superbee finite volume methods, with CPU times = 24, 26, 37 min, respectively.

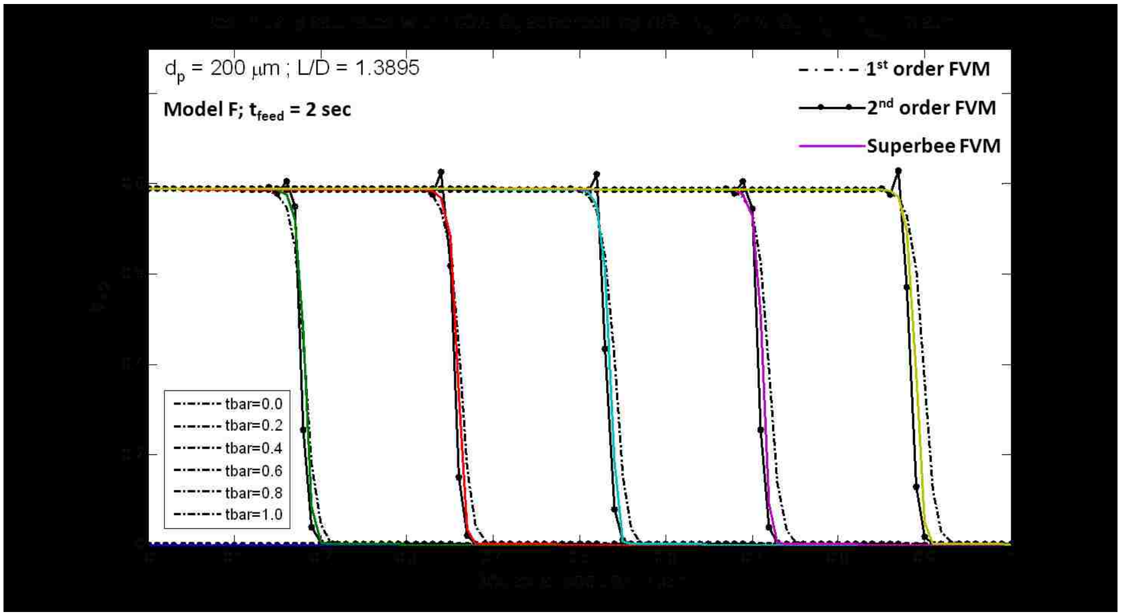


Figure 6.6: Adsorption steep fronts for model F (t_{feed} = 2 sec, d_p = 200 μm), using 1st order upwind, 2nd order upwind, and Superbee finite volume methods, with CPU times = 3, 4, 8 min, respectively.

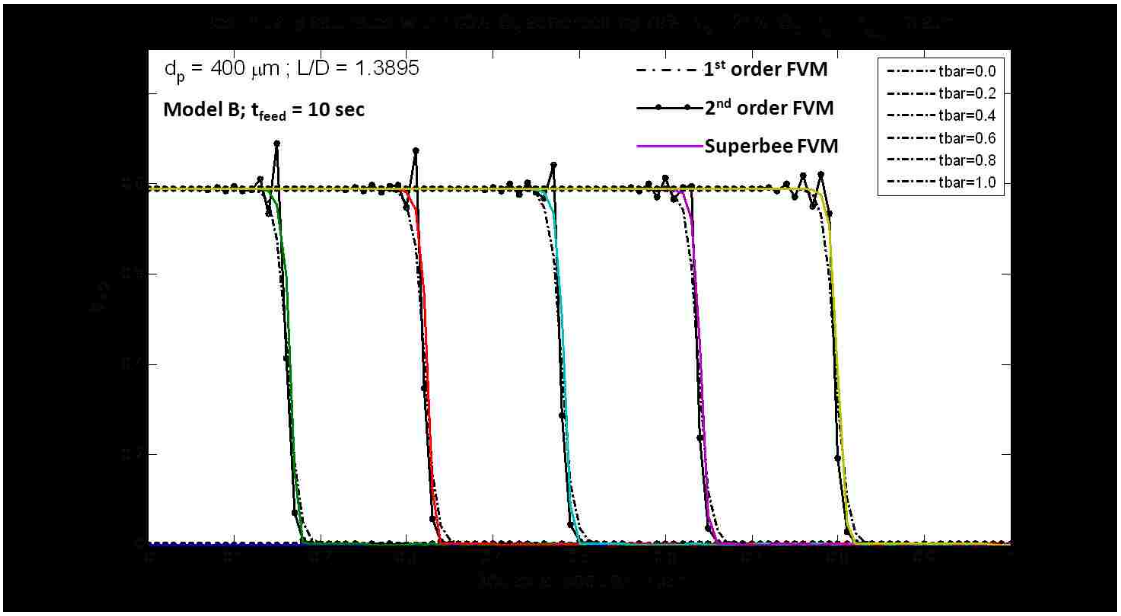


Figure 6.7: Adsorption steep fronts for model B (t_{feed} = 10 sec, d_p = 400 μm), using 1st order upwind, 2nd order upwind, and Superbee finite volume methods, with CPU times = 4, 4, 4 min, respectively.

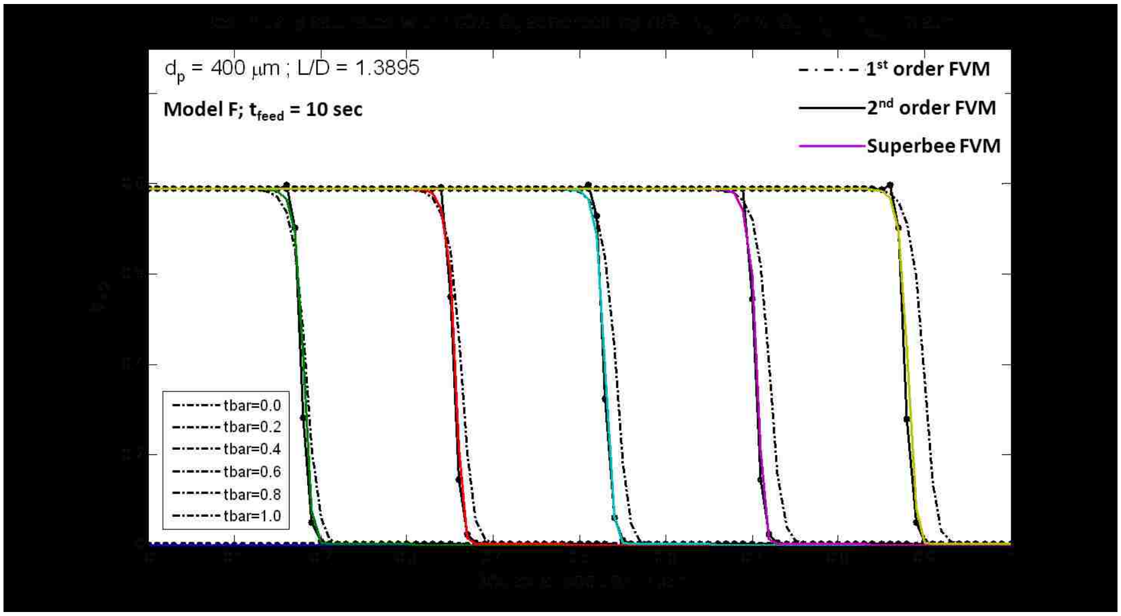


Figure 6.8: Adsorption steep fronts for model F (t_{feed} = 10 sec, d_p = 400 μm), using 1st order upwind, 2nd order upwind, and Superbee finite volume methods, with CPU times = 75, 86, 105 min, respectively.

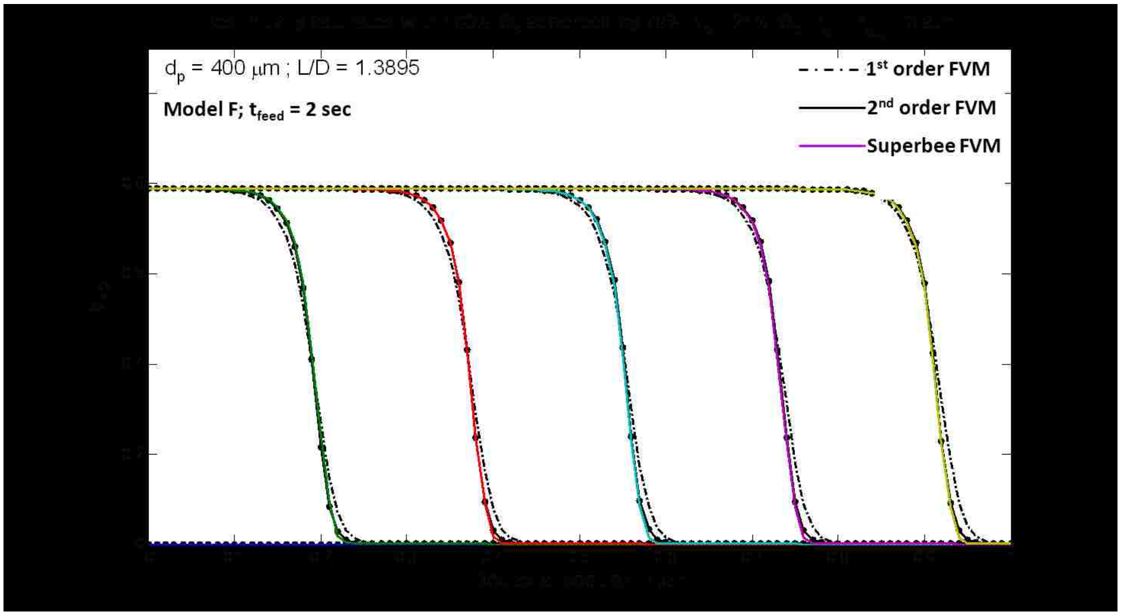


Figure 6.9: Adsorption steep fronts for model F (t_{feed} = 2 sec, d_p = 400 μm), using 1st order upwind, 2nd order upwind, and Superbee finite volume methods, with CPU times = 14, 16, 18 min, respectively.

Comparison among these three approximation schemes is again simulated using model F (detailed model) for displacing 100% O_2 saturated bed by 79% N_2 + 21% O_2 feed at T = 298 K, P_o = P_{out} = 4 atm, d_p = 400 μm, Q_o = 5.248 mmole/cm^2/s, t_{feed} = 2 s, and grid density n = 151. Figures 6.10, 6.11 and 6.12 show that solution convergence is improved using higher order upwind difference. Nonlinear flux conserved approximation with Superbee flux limiter gives the best solution approximation in dealing with steep fronts. The solution differences are less obvious in these three cases because the detailed model is used, which gives rise to dispersive steep fronts in rapid adsorption, and the grid density has been increased, which results to better solution convergence but at excessively long computing time. When the same model with equal grid density (n = 151) is used to simulate purge desorption step under nonlinear finite volume approximation with Superbee flux limiter, 7 minutes of CPU time is required, compared to 105 minutes required for adsorption step.

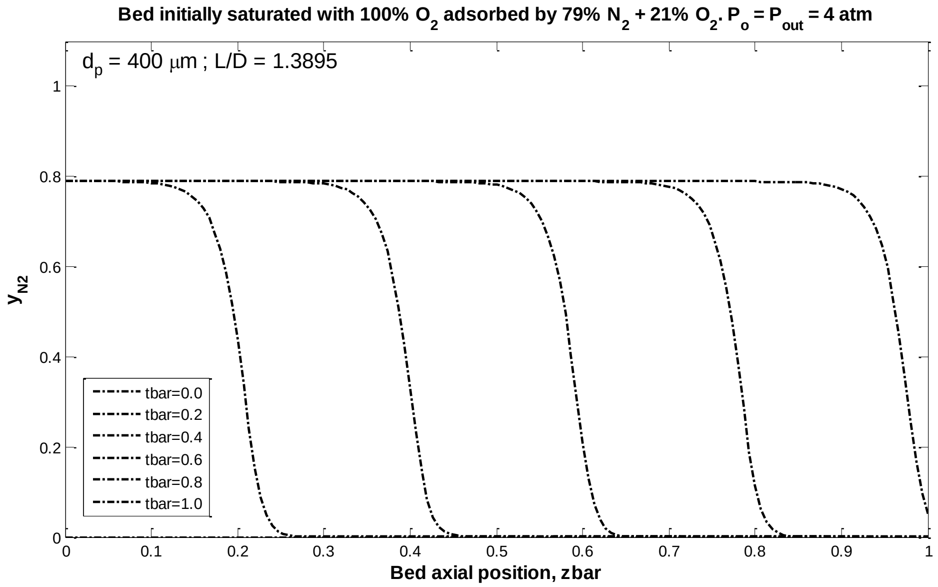


Figure 6.10: Adsorption steep fronts for model F (t_{feed} = 2 sec, d_p = 400 µm), using 1st order upwind finite volume method with n = 151. CPU time = 69 min.

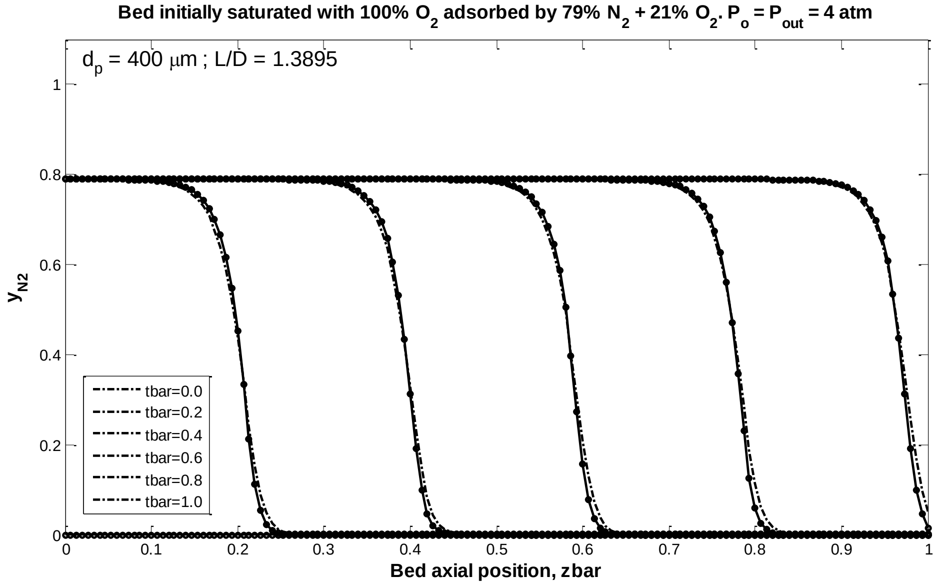


Figure 6.11: Adsorption steep fronts for model F (t_{feed} = 2 sec, d_p = 400 µm), using 2nd order upwind finite volume method with n = 151. CPU time = 78 min.

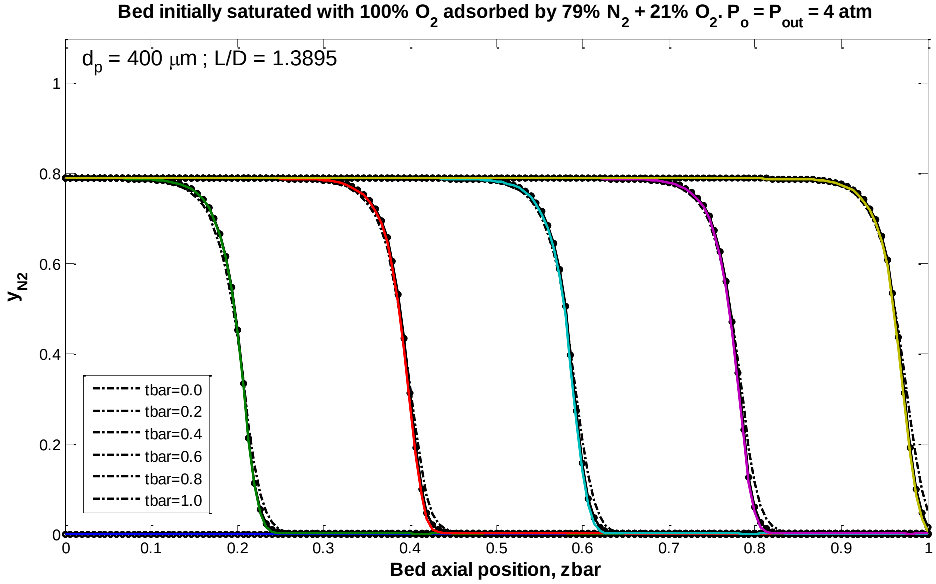


Figure 6.12: Adsorption steep fronts for model F (t_{feed} = 2 sec, d_p = 400 μm), using Superbee finite volume method with n = 151. CPU time = 105 min.

6.5 References

Alhumaizi, K.: Flux Limiting Solution Techniques for Simulation of Reaction-Diffusion-Convection System. *Communications in Nonlinear Science and Numerical Simulation*, **12**, 953-965 (2007).

Dullemond, C.P.: Numerical Fluid Dynamics Chapter 4: Advection Algorithms II. Flux Conservation, Subgrid Models and Flux Limiters. Max-Planck-Institut fuer Astronomie (2009).
http://www.mpia.de/homes/dullemon/lectures/fluiddynamics/index.html

Godunov, S.K.: Different Methods for Shock Waves. *Ph.D. Dissertation*, Moscow State University (1954).

Godunov, S.K.: A Difference Scheme for Numerical Solution of Discontinuous Solution of Hydrodynamic Equations. *Math. Sbornik*, **47**, 271-306 (1959). Translated US Joint Publ. Res. Service, JPRS 7226 (1969).

Griffiths, G.W., Schiesser, W.E.: Traveling Wave Analysis of Partial Differential Equations: Numerical and Analytical Methods with Matlab and Maple. Chapter 2, Academic Press (2011).

Lin, L.: Numerical Simulation of Pressure Swing Adsorption Process. *Master Thesis*, Simon Fraser University (1997).

Saucez, P., Schiesser, W.E., Wouwer, A.V.: Upwinding in the Method of Lines. *Mathematics and Computers in Simulation*, **56**, 171-185 (2001).

Schiesser, W.E., Griffiths, G.W.: A Compendium of Partial Differential Equation Models: Method of Lines Analysis with Matlab. Cambridge University Press (2009).

Toro, E.F.: Riemann Solvers and Numerical Methods for Fluid Dynamics. 2nd Ed. Springer (1999).

Wouwer, A.V., Saucez, P., Schiesser, W.E.: Adaptive Method of Lines, Chapter 1, Chapman and Hall/CRC (2001).

Chapter 7

Non-isobaric, Non-isothermal, and Non-equilibrium Model Simulation of Oxygen Back-Purge Step in Medical Oxygen Concentrator by Rapid Pressure Swing Adsorption

Purge desorption step in rapid pressure swing adsorption (RPSA) system for a medical oxygen concentrator (MOC) is mathematically modeled and numerically simulated. The details of this model and its solution method are given in Chapters 5 and 6. Simulation is performed in the order of increased model complexities using different particle sizes, column geometries, and purge desorption times. Simulation results show that pressure drop, non-isothermal operation, and finite adsorption kinetics are the primary contributions to purge inefficiency. Purge desorption using larger particle is inhibited by slow kinetics. Kinetic is improved substantially by using smaller particles at the cost of increased pressure drop and particle agglomeration. Pressure drop can be alleviated by using column of reduced length to diameter ratio, L/D, or pancake adsorber, but that introduces undesirable effects of lowered fraction of column volume cleaned at the purge entrance region and gas mal-distribution. These resistances add to purge gas requirement, hence lower O_2 recovery and increase BSF. In addition, ultra-rapid purge step may not be applicable because amount of purge gas required is high and adsorbent bed is not properly cleaned. An optimum particle size, which considers fraction of column volume cleaned up and minimum amount of purge gas required, is established. This work is sent to Adsorption Journal for peer review and publication.

7.1 Parametric Simulation of Desorption-by-Purge

Schematic diagram for a simulation case is illustrated in Figure 7.1. A packed bed with length L, diameter D, and cross sectional area A, that contains 160g LiX zeolite adsorbent at particle size d_p, and initially saturated by 79% N_2 + 21% O_2 at 1 atm and 298 K is purged with 100% O_2 at 298 K and 1+ atm (to overcome column pressure drop). The effluent gas is at 1 atm. A total of 306 mmole O_2 is used to purge this adsorbent column, this amount of purge gas gives rise to 99.98% clean bed at the end of purge step under ideal condition. N_2 desorbed from the column is accumulated at the purge exit, total purge durations of 0.2 seconds, 2 seconds, and 10 seconds are simulated to compare the efficiencies between normal and rapid purge.

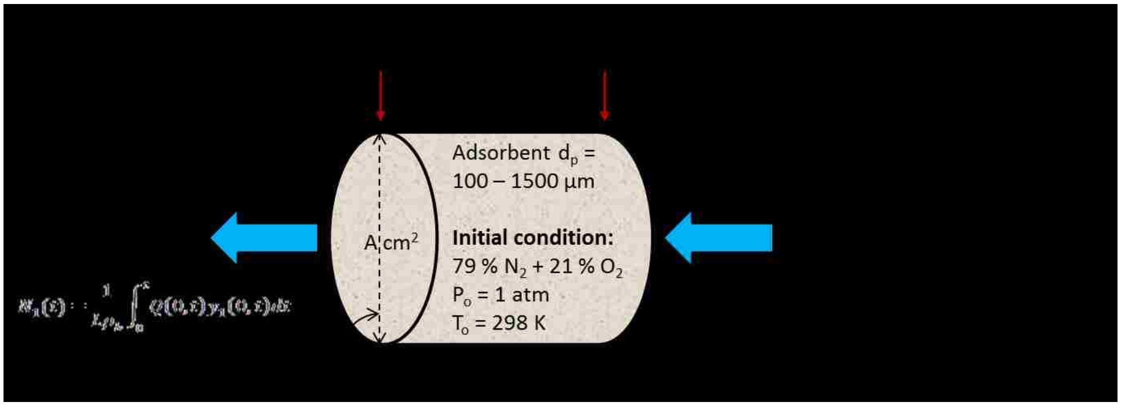

Adsorbent d_p = 100 – 1500 μm
Initial condition:
79 % N_2 + 21 % O_2
P_o = 1 atm
T_o = 298 K
A cm^2
$N_1(t) = \frac{1}{L\rho_b}\int_0^t Q(0,t)y_1(0,t)dt$

Figure 7.1: Schematic diagram of purge desorption step in a RPSA MOC system.

7.1.1 Input Parameters for Model Simulation

Listed below are input parameters for model simulation. They include adsorption column geometries, adsorbent bed properties, gas phase properties, mass transfer properties, adsorption equilibria and heat properties. Among them, some are users' inputs, some information is intuitive, the rest of them are obtained from the literatures.

Adsorption column geometries:

Two adsorber designs having different length (L) to diameter (D) ratios but the same total amount of adsorbent [$W = \pi/4\ D^2 L\rho_b$ ~ 160 gms] packed in them were used in the simulations.

- Conventional type: diameter D = 6.0325 cm; length L = 8.382 cm; cross sectional area A = 28.5815 cm^2; L/D = 1.3895
- Pancake type: diameter D = 12 cm; length L = 2.12 cm; cross sectional area A = 133.0973 cm^2; L/D = 0.1767

Adsorbent bed properties:

- Particle diameter, d_p = 100 – 1500 µm
- Bulk density, ρ_b = 0.668 g/cm^3 (160 g LiX adsorbent contained in column)
- Solid heat capacity, C_s= 0.28 cal/g/K (Rege and Yang 1997)
- Internal void, ε_p = 0.35
- External void, ε = 0.40

- Helium void, $\varepsilon = \varepsilon_p \ 1 - \varepsilon \ + \varepsilon$ = 0.61
- Particle tortuosity factor, τ_p = 3 (Alpay, Kenney and Scott 1994)

Gas phase properties:

- Gas constant, R = 8.314472 x 10^4 g.cm^2/K/mmol/s^2
- Dynamic viscosity of air, μ = 18.2385 x 10^{-5} g/cm/sec (Rama Rao et al. 2010)
- Thermal conductivity, k_g = 6.44 x 10^{-5} cal/cm/sec/K
- Thermal heat capacity, C_g = 0.00687 cal/mmol/K (Rege and Yang 1997)
- Isosteric heat of adsorption of N_2, q_1 = 5.60 cal/mmol (Rege and Yang 1997)
- Isosteric heat of adsorption of O_2, q_2 = 3.16 cal/mmol (Rege and Yang 1997)
- Molecular weight of N_2, Mg_1 = 0.028 g/mmol
- Molecular weight of O_2, Mg_2 = 0.032 g/mmol

Mass transfer properties:

Effective Knudsen diffusivities, $D_{K,i}^{e}$, and effective molecular diffusivity, $D_{m,1-2}^{e}$, on Zeochem LiLSX sorbent, obtained from Todd and Webley (2006), are interpolated to T = 300K and tabulated in Table 7.1.

Table 7.1: Effective diffusion coefficients for N_2 and O_2 in Zeochem LiLSX over the range of operating conditions investigated in model simulation. Interpolated values from Todd and Webley (2006).

Component	**Effective Knudsen diffusivity, $D^{e}_{K,i}$ (cm^2/s)**	**Effective molecular diffusivity of O_2-N_2, $D^{e}_{m,1-2}$, (cm^2/s)**	
	(at 300 K and 1 – 4 atm)	(at 300 K and 1 atm)	(at 300 K and 4 atm)
N_2	0.03277	0.0348	0.008707
O_2	0.0307		

Table 7.2: Isotherms of O_2 and N_2 on LiX from Rege and Yang (1997).

Sorbent	**Sorbate**	**k_1 (mmol/g/atm)**	**k_2 (K)**	**k_3 (atm^{-1})**	**k_4 (K)**
LiX	O_2	1.11 x 10^{-3}	1593.0	1.03 x 10^{-4}	2061.9
LiX	N_2	1.25 x 10^{-3}	2168.6	2.07 x 10^{-4}	2455.5

Table 7.3: Isosteric heats of adsorption and heat capacities of O_2 and N_2 on LiX from Rege and Yang (1997).

Sorbent	**Sorbate**	**$-\Delta H$ (kcal/mol)**	**C_{pg} (cal/mol/K)**
LiX	O_2	3.16	8.27
LiX	N_2	5.60	6.50

Table 7.4: Empirical model of adsorption equilibria without thermodynamic consistency from Rege and Yang (1997).

T_s (K)	**$mb_1 = K_{N2}$ (mmol/g/atm)**	**$mb_2 = K_{O2}$ (mmol/g/atm)**	**$b_1 = B_{N2}$ (atm^{-1})**	**$b_2 = B_{O2}$ (atm^{-1})**	**$m = mb_1/b_1$ (mmol/g)**	**$m = mb_2/b_2$ (mmol/g)**
310	1.365	0.189	0.570	0.080	2.37	2.39
298	1.809	0.233	0.784	0.104	2.23	2.31
285	2.521	0.297	1.142	0.143	2.08	2.21
280	2.887	0.328	1.332	0.163	2.02	2.17

Adsorption equilibria:

The following empirical N_2 and O_2 adsorption isotherms on LiX zeolite at different temperatures were obtained from Rege and Yang (1997). The isotherm parameters, isosteric heats of adsorption, and heat capacities are given in Tables 7.2 and 7.3. These values and equation (7.1) are the inputs to the mathematical model.

$$n_i^{\infty} = \frac{K_i P y_i}{1+ \quad B_j P y_j}; K_i = k_1 exp \ \frac{k_2}{T_s} \ ; B_i = k_3 exp \ \frac{k_4}{T_s} \qquad (7.1)$$

Though fitting experimental data, this empirical model does not possess thermodynamic consistency, as demonstrated by the varying maximum capacity, m, with solid phase temperature, T_s, in Table 7.4.

However, when treated strictly as binary Langmuir isotherms as given by equation (7.2), the Henry's Law selectivity of adsorption of N_2 over O_2 at 298 K is 7.53 and the corresponding isosteric heats of adsorption of N_2 and O_2 are, respectively, 4.88 and 4.10 kcal/mole.

$$n_i^{\infty} = \frac{m B_i P y_i}{1+ \quad B_j P y_j} = \frac{K_i P y_i}{1+ \quad B_j P y_j}; K_i = K_i^o exp \ \frac{Q_{Ki}}{RT_s} \ ; B_i = B_i^o exp \ \frac{Q_{Bi}}{RT_s} \qquad (7.2)$$

7.2 The Importance of Product Back Purge Step in RPSA Process

The back purge step is critical for a PSA process design because (a) it provides an efficient means of overall column regeneration (desorption of N_2) and (b) it creates a section of the column at the O_2 product end which is equilibrated with the O_2 enriched product gas at the start of an adsorption step. Goal (b) is critical for maintaining the product O_2 purity during the adsorption step. Two key performance variables for a PSA O_2 generator are (i) bed size factor (BSF, total amount of zeolite adsorbent in the PSA system per unit amount of contained O_2 in product gas per unit time) and (ii) net O_2 recovery by the process (R, amount of O_2 in product gas per unit amount of O_2 in feed air per cycle).

The experimental study of a simulated, four-step Skarstom-like RPSA cycle for MOC application using a commercial sample of LiX zeolite and dry and CO_2-free compressed air feed elaborated in Chapter 4, showed that there can be an optimum cycle time for the RPSA process where the BSF is lowest (smallest adsorber size) and the O_2 recovery is moderate (Chai et al. 2011). Figure 7.2 provides an example of such optimum RPSA process performance from that study using adsorption pressure P_A = 3 atm, desorption pressure P_D = ~ 1 atm and an adsorbent particle size d_p = ~ 350 μm. The optimum cycle time t_c = ~ 2 – 4 seconds, the optimum BSF = ~ 50 lbs/TPD_c, and the O_2 recovery = ~ 20%. Consequently, the optimum duration of a back purge step of a RPSA process for MOC needs to be < 1 – 3 seconds, and the purge gas flow rate can be high.

The following idealized model calculation estimates the sensitivity of the specific amount of back purge gas (moles/kg of sorbent) used in a generic Skarstrom-like PSA cycle in determining BSF and R.

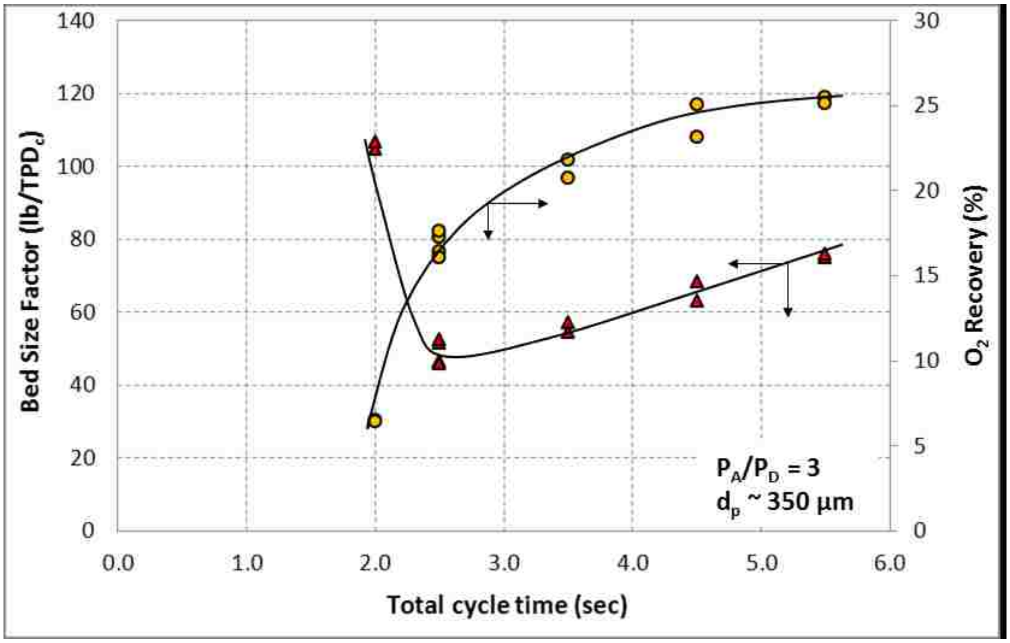


Figure 7.2: Experimental RPSA process performance data for a simulated MOC using LiX zeolite.

7.3 Sensitivity of Back Purge Gas Quantity to PSA Process Performance

The amount of back purge gas used in a generic Skarstrom-like PSA cycle determines process performances, O_2 recovery (R) and bed size factor (BSF). Given below are three schemes of overall and component mass balances performed on an idealized PSA process producing pure O_2 from air consisting of 79% N_2 + 21% O_2. Using N_2 = component 1, O_2 = component 2, model assumptions include (a) isothermal operation, (b) local thermodynamic equilibrium (mass and heat) between the gas and solid phases in the column at all times during the cycle, and (c) plug flow of gas with no column pressure drop or axial dispersion. All three schemes illustrate the sensitivity of α = Purge/Feed ratio or purge gas amount onto BSF and R.

7.3.1 Scheme A

Figure 7.3 shows the block diagram of an ideal PSA cycle with various influent and effluent flows and quantities E, F, D, PU, NP, PR (moles/cycle) for producing 100% O_2 from 79% N_2 + 21% O_2.

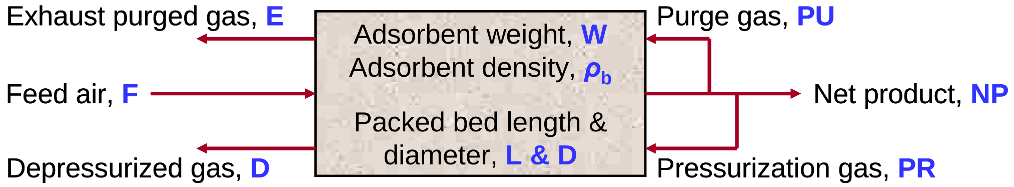


Figure 7.3: Block diagram of an ideal O_2 PSA system for scheme A.

Amount of component 1 (mole) in adsorbent bed is given by:

$$Wn_1 \;\; P, T_0, y_1 \;\; = Wn_1^{\infty} \;\; P, T_0, y_1 \;\; + \frac{W}{\rho_b} \frac{\varepsilon y_1 P}{R_g T_0} \tag{7.3}$$

where n_1 is the specific amount of component 1 in the adsorbent bed (adsorbed onto solid + in the void space), y_1 is the gas phase mole fraction of component 1, P and T_0 are the system pressure and temperature, n_1^{∞} is the equilibrium specific amount of component 1 adsorbed onto solid, W and ρ_b are the adsorbent weight and bulk density, ε is the total or helium void fraction, and $R_g$is the gas constant.

During and after the pressurization step:

Packed bed which has been completely purged clean with 100% O_2 purge gas at desorption pressure P_D is pressurized with 100% O_2 product gas to adsorption pressure P_A. Amount of product gas used for this pressurization step:

$$PR = W\ n_2\ P_A, T_0, y_2 = 1\ \ - n_2\ P_D, T_0, y_2 = 1 \tag{7.4}$$

Specific amount of O_2 in packed bed at this state:

$$n_2\ P_A, T_0, y_2 = 1 \tag{7.5}$$

During and after the adsorption step:

Feed air composed of 79% N_2 + 21% O_2 is fed into packed bed at P_A. 100% O_2 product gas is produced thus all N_2 from F is left in the packed bed. In the ideal condition, packed bed is saturated with 79% N_2 + 21% O_2 in gas phase at the end of adsorption step. Amount of feed air used for this step:

$$F = W\ n_1\ P_A, T_0, y_1 = y_{1\_sat\_ads}\ \ /y_{1\_air} \tag{7.6}$$

Specific amount of N_2 in packed bed at this state:

$$n_1\ P_A, T_0, y_1 = y_{1\_sat\_ads} \tag{7.7}$$

Specific amount of O_2 in packed bed at this state:

$$n_2\ P_A, T_0, y_2 = y_{2\_sat\_ads} \tag{7.8}$$

During and after the depressurization and purge step:

All N_2 gas is expunged from packed bed, leaving no N_2 residual in packed bed for pressurization step that follows. Total amount of N_2 gas depressurized and purged out:

$$-Dy_1^D - Ey_1^E = -D(1 - y_2^D) - E(1 - y_2^E) = -Wn_1\ P_A, T_0, y_1 = y_{1\_sat\_ads}\ \ = \\ -y_{1\_air}F \tag{7.9}$$

Amount of O_2 gas depressurized:

$$-Dy_2^D = W\ n_2\ P_D, T_0, y_2^D\ \ - n_2\ P_A, T_0, y_2 = y_{2\_sat\_ads} \tag{7.10}$$

Amount of O_2 gas purged in and out:

$$PU - Ey_2^E = W\ \ n_2\ \ P_D, T_0, y_2 = 1\ \ - n_2\ \ P_D, T_0, y_2^D \tag{7.11}$$

Adding three equations above to get:

$$PU - E - D = -y_{1\_air}F + W\ \ n_2\ \ P_D, T_0, y_2 = 1\ \ - n_2\ \ P_A, T_0, y_2 = y_{2\_sat\_ads} \tag{7.12}$$

Specific amount of O_2 in packed bed at this state:

$$n_2\ \ P_D, T_0, y_2 = 1 \tag{7.13}$$

Total mass balance on the block diagram at cyclic steady-state (no accumulation):

$$F = D + E + NP \tag{7.14}$$

Amount of purge gas used:

Eliminate D and E from equation (7.12) by equation (7.14) to get the amount of purge gas used:

$$PU = (1 - y_{1\_air})F - NP + W\ \ n_2\ \ P_D, T_0, y_2 = 1\ \ - n_2\ \ P_A, T_0, y_2 = y_{2\_sat\_ads} \tag{7.15}$$

Net O_2 product:

$$NP = O_2 recovery\ \times\ 0.21F \tag{7.16}$$

With $F$and Wn_2 known, NP as a function of O_2 recovery, and PU calculated from equation (7.15), all process variables are solved.

7.3.2 Scheme B

Figure 7.4 is a block diagram showing the influent and effluent flows with quantities F and EP (moles/cycle) during an ideal adsorption step with presure P_A and isothermal T_0 for producing 100% O_2 from 79% N_2 + 21% O_2.

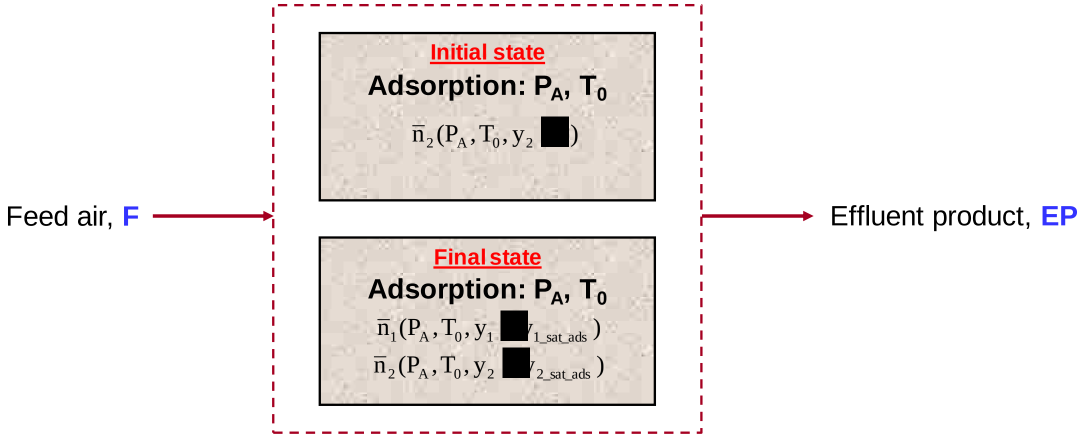

Initial state
Adsorption: P_A, T_0
$\bar{n}_2(P_A, T_0, y_2$ [redacted]$)$
Final state
Adsorption: P_A, T_0
$\bar{n}_1(P_A, T_0, y_1$ [redacted]$_{1\_sat\_ads})$
$\bar{n}_2(P_A, T_0, y_2$ [redacted]$_{2\_sat\_ads})$
Feed air, F
Effluent product, EP

Figure 7.4: Block diagram of an ideal adsorption step of O_2 PSA system for scheme B.

During and after the pressurization step:

Packed bed which has been completely purged clean with 100% O_2 purge gas at P_D is pressurized with 100% O_2 product gas to P_A. Amount of product gas used for this pressurization step:

$$PR = W \;\; n_2 \;\; P_A, T_0, y_2 = 1 \;\; - n_2 \;\; P_D, T_0, y_2 = 1 \qquad (7.17)$$

Specific amount of O_2 in packed bed at this state:

$$n_2 \;\; P_A, T_0, y_2 = 1 \qquad (7.18)$$

During and after the adsorption step:

Feed air composed of 79% N_2 + 21% O_2 is fed into packed bed at P_A. 100% O_2 product gas is produced thus all N_2 from F is left in the packed bed. In ideal condition, packed bed is saturated with 79% N_2 + 21% O_2 in gas phase at the end of adsorption step. Amount of feed air used for this step:

$$F = W\ n_1\ \ P_A, T_0, y_1 = y_{1\_sat\_ads}\ \ \ /y_{1\_air} \qquad (7.19)$$

Specific amount of N_2 in packed bed at this state:

$$n_1\ \ P_A, T_0, y_1 = y_{1\_sat\_ads} \qquad (7.20)$$

Specific amount of O_2 in packed bed at this state:

$$n_2\ \ P_A, T_0, y_2 = y_{2\_sat\_ads} \qquad (7.21)$$

The overall balance for adsorption step:

Final state – Initial state = $F - EP$

$$W\ n_1\ \ P_A, T_0, y_1 = y_{1\_sat\_ads}\ \ + n_2\ \ P_A, T_0, y_2 = y_{2\_sat\_ads}\ \ - n_2\ \ P_A, T_0, y_2 = 1\ \ \ = F - EP \qquad (7.22)$$

Since F and all the states n_1 and n_2 are known, effluent of product, EP, can be calculated.

Purge/feed ratio (PU/F) and O_2 recovery:

From net product, $NP = EP - PR - PU$, and purge/feed ratio, $\alpha = PU/F$, we get $NP = EP - PR - \alpha F$. With EP, PR, F calculated from the equations above, NP as a function of α is known.

O_2 recovery, R (%):

$$O_2 recovery = NP/0.21F \tag{7.23}$$

Bed size factor, BSF (lb/TPD_c):

$$BSF = \frac{W(lb)}{\frac{100\%\, O_2\, NP(mole)}{cycle}} \times \frac{mole\ O_2}{32g\ O_2} \times \frac{454g\ O_2}{lb\ O_2} \times \frac{2000lb\ O_2}{Ton\ O_2} \times \frac{t_c\,(\mathrm{sec})}{cycle} \times \frac{Day}{60\times60\times24\ sec} \tag{7.24}$$

The resulting process variables calculated from Scheme A and Scheme B allow BSF and R to be plotted against purge/feed ratio (PU/F) using different total cycle times, t_c, as given by Figures 7.5 and 7.6. Note that Scheme A and Scheme B produce exactly the same results. This confirms the solutions from either scheme.

Figure 7.5 shows that R is not affected by t_c because the process is assumed ideal, in which mass and heat transfer kinetics are instantaneous. On the other hand, Figure 7.6 shows that BSF is a function of t_c, lower t_c yields lower BSF. Moreover, BSF at low values of α approches zero because, again, the process is assumed ideal, and those low values of α may not be really practical. Nonetheless, Figures 7.5 and 7.6 show that as α increases, R reduces significantly and BSF increases exponentially. This demonstrates the sensitivity of O_2 purge amount onto BSF and R. Scheme C in Section 7.3.3 provides further explanation on sensitivity of back purge gas quantity to BSF and R.

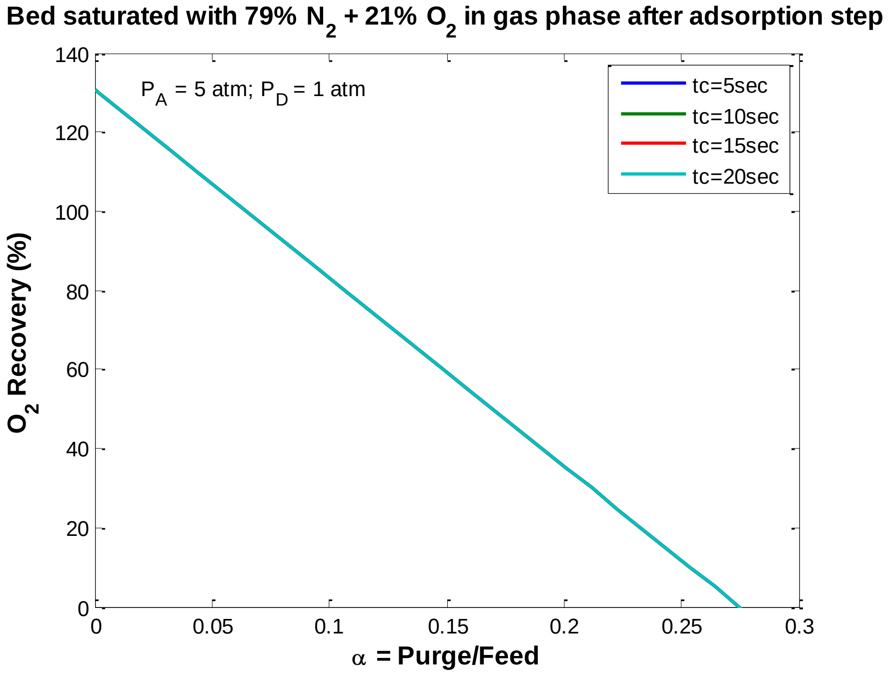


Figure 7.5: O_2 recovery (R) vs purge/feed ratio (α) of an ideal O_2 PSA system using Schemes A and B.

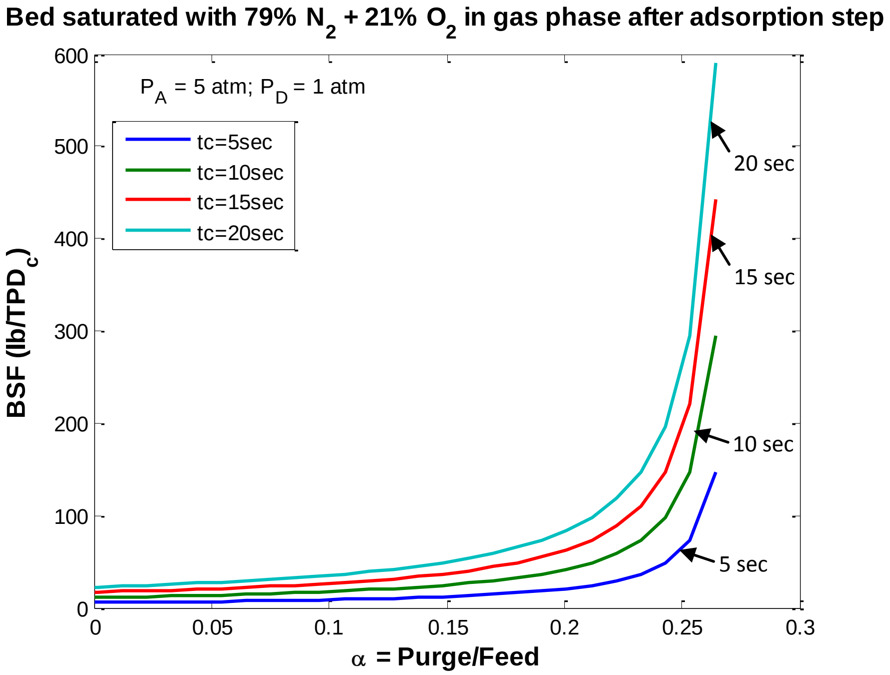


Figure 7.6: Bed size factor (BSF) vs purge/feed ratio (α) of an ideal O_2 PSA system using Schemes A and B.

7.3.3 Scheme C

The ideal cyclic process steps are: (A) counter-current column back purge at ambient pressure using pure O_2, (B) counter-current column pressurization from ambient to adsorption pressure level of P_A with pure O_2, (C) co-current adsorption from compressed air at pressure P_A to produce a pure O_2 effluent gas at pressure P_A, and (D) counter-current column depressurization from pressure P_A to ambient pressure. A part of the effluent gas from step C is withdrawn as the net pure O_2 product by the process and the balance is used in steps A and B. The effluent gases from steps A and D are wasted.

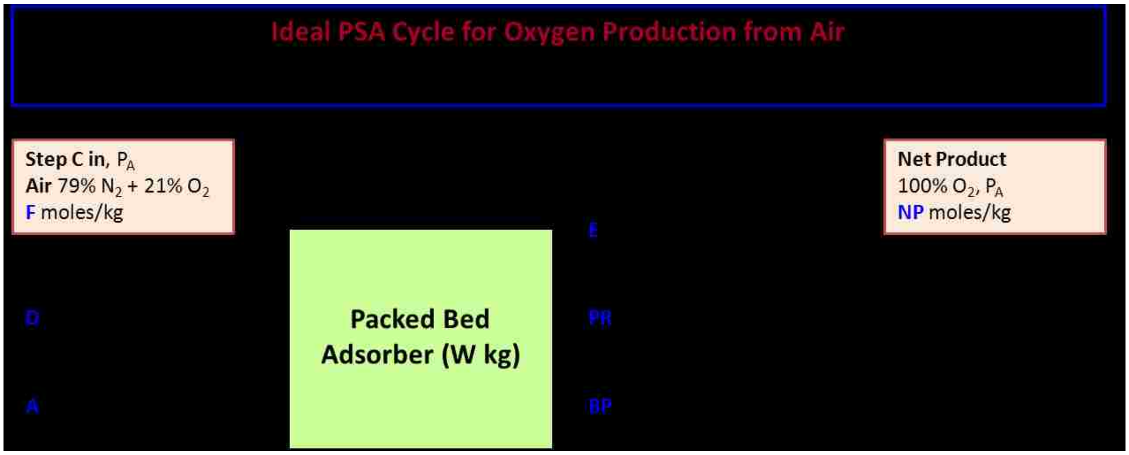

Ideal PSA Cycle for Oxygen Production from Air
Step C in, P_A
Air 79% N_2 + 21% O_2
F moles/kg
Net Product
100% O_2, P_A
NP moles/kg
E
D
Packed Bed Adsorber (W kg)
PR
A
BP

Figure 7.7: Block diagram of an ideal O_2 PSA system for scheme C.

Figure 7.7 is a block diagram of the model showing various influent and effluent flows and quantities (moles/kg of total adsorbent in the PSA system/cycle) for the steps. The key performance variables for the process are:

$$BSF = 1/(NP.\omega) \tag{7.25}$$

$$R = NP/0.21F \tag{7.26}$$

where ω (= $1/t_c$) is the cycle frequency (number of cycles per unit time) and t_c is the total cycle time.

It can be shown by component mass balances (N_2 = 1; O_2 =2) for various steps that:

$$F = N_1^C/0.79 \tag{7.27}$$

$$PR = \; N_2^B - N_2^A \tag{7.28}$$

$$E = PR + BP + NP = 0.21F + \; N_2^B - N_2^C \tag{7.29}$$

$$NP = \; \alpha - BP \;\; ; \alpha = \; E - PR \tag{7.30}$$

where $N_i^j \;\; = n_i^j + \varepsilon\rho_g^j y_i^j/\rho_b$ is the total (adsorbed + void gas) specific amount (moles/kg) of component i in the column at the end of step j, n_i^j is the specific equilibrium amount (moles/kg) of component i adsorbed at the end of step j when the gas phase density (moles/cm^3) is $\rho_g^j \;\; = P^j/R_gT$ and the gas phase mole fraction of component i is y_i^j. P^j (atm) is the column pressure at the end of step j, ε is the helium void fraction of the column, T is the system temperature, R_g is the gas constant, and ρ_b is the bed density. Variables F, E, PR and α can be estimated using equations (7.27) – (7.30) and the adsorption isotherms of pure N_2 and O_2 and their mixtures at T.

It follows from equations (7.25), (7.26), and (7.30) that:

$$\frac{\Delta BSF}{BSF} \; = \; \frac{\Delta R}{R} \; = \beta \; \frac{\Delta BP}{BP} \tag{7.31}$$

$$\beta = \; \frac{\alpha}{0.21F(R)} - 1 \tag{7.32}$$

Equations (7.31) and (7.32) indicate that the changes in the absolute values of the parameters (BSF) and (R) due to a change in the value of (BP) will be magnified by a factor of β (> 1). Thus, the amount of back purge used in a PSA process is a sensitive variable for determining the overall performance of a PSA process.

Using the published adsorption isotherms of pure N_2 and O_2 and their mixtures on a sample of ~ 100% Li exchanged X zeolite, which could be described by an empirical Langmuir-like model (Rege and Yang 1997), in order to derive the following relationship between the parameters β and R for the above-described PSA cycle by assuming that the adsorption pressure (P_A) is 5 atm and the system temperature (T) is 298 K:

$$\beta = \frac{1.31}{R} - 1 \tag{7.33}$$

According to equation (7.33), the value of the parameter β is 4.24 if the base case O_2 recovery is 25% (R = 0.25), which is typical for an O_2 RPSA system (Chai et al. 2011). Consequently, it follows from equations (7.31) and (7.33) that a ~ 10% change (or estimation error) in the quantity of back purge (BP) from a base case value can produce a ~ 42.4% change (or estimation error) in BSF or R from those of the base case values. This clearly demonstrates that the quantity of back purge in a PSA process is a very sensitive design variable in establishing the overall process performance, albeit for an idealized O_2 PSA process.

7.4 Model Complexity

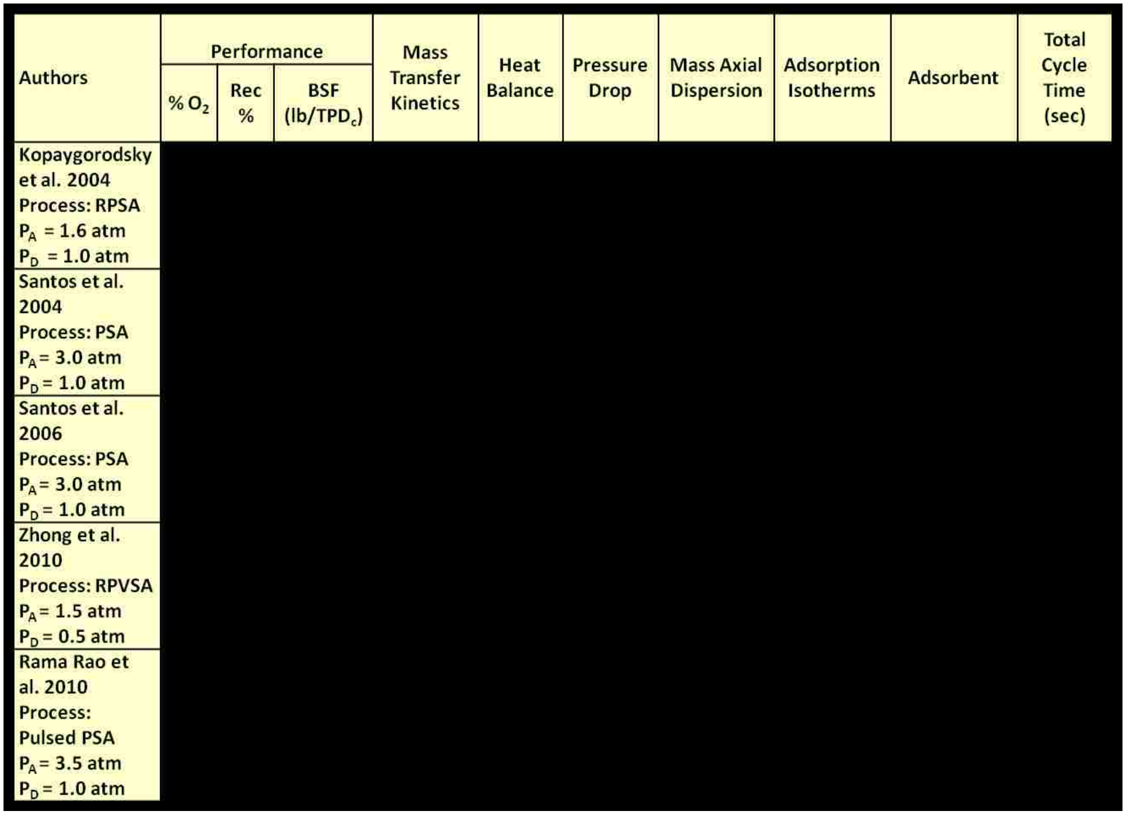
Model simulation of PSA system in the literature is generally simplified by assuming one or more of the conditions below:

- no pressure drop (isobaric)
- no temperature change (isothermal)
- instantaneous adsorption of species i from gas to solid phase (local adsorption equilibrium)
- instantaneous heat transfer between gas and solid phases (local thermal equilibrium)
- perfect gas mixing (absence of axial dispersions)

For example, many model studies of isobaric and isothermal (or adiabatic) desorption of a single or binary adsorbates (dilute or bulk) from a packed adsorbent column by purge with a less strongly adsorbed gas (pure or binary) under the conditions of local thermodynamic equilibrium (mass and heat) have been published (Wicke 1939a,b; Rhee and Amundson 1970; Rhee et al. 1972; Basmadjian et al. 1975a,b; Jacob and Tondeur 1983; Sircar and Kumar 1985; Sircar and Golden 1995). A rigorous analytical solution of an isothermal and isobaric desorption by purge model for a bulk binary Langmuirian adsorbate system has also been published (Sircar and Golden 1995). It has been observed that the assumption of local equilibrium is generally adequate to describe isothermal and isobaric desorption-by-purge when a low to moderate purge gas velocity is used.

More recently, several numerical models of cyclic PSA O_2 processes for MOC application, where desorption of N_2 from a LiX zeolite adsorbent column by purge with an O_2 enriched gas was an integral part of the process, were published. A reference list of these process simulation studies and the corresponding process performances is reproduced in Table 7.5 from an earlier publication (Chai et al. 2011). Table 7.5 shows that all of these simulations have been carried out under isothermal process conditions except that of Zhong et al. (2010), with or without inclusion of column pressure drop in the model, and for one case, by assuming an unrealistic linear adsorption isotherm model with no adsorbate mass transfer resistance.

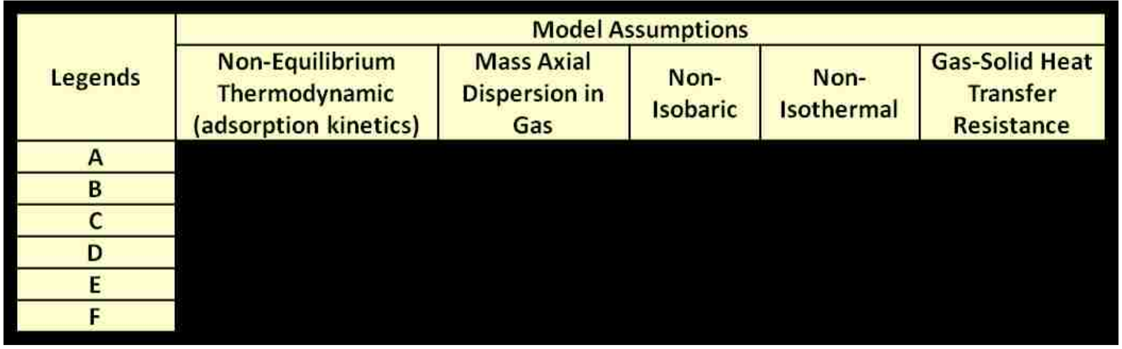
Table 7.5: Model simulation studies of various PSA O_2 concentrators.



Authors	Performance			Mass Transfer Kinetics	Heat Balance	Pressure Drop	Mass Axial Dispersion	Adsorption Isotherms	Adsorbent	Total Cycle Time (sec)
	% O_2	Rec %	BSF (lb/TPD_c)							
Kopaygorodsky et al. 2004 Process: RPSA P_A = 1.6 atm P_D = 1.0 atm	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
Santos et al. 2004 Process: PSA P_A = 3.0 atm P_D = 1.0 atm	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
Santos et al. 2006 Process: PSA P_A = 3.0 atm P_D = 1.0 atm	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
Zhong et al. 2010 Process: RPVSA P_A = 1.5 atm P_D = 0.5 atm	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
Rama Rao et al. 2010 Process: Pulsed PSA P_A = 3.5 atm P_D = 1.0 atm	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]

The purpose of this work is to numerically investigate the effects of various resistances in mass, heat and momentum transfer processes present inside a real packed bed adsorber undergoing a rapid ‘desorption-by-purge’ process having a duration of only a few seconds or less. The effects of the transport resistances are magnified due to the shortness of the process step time. Analysis of the effects of (a) ad(de)sorption mass transfer resistance, (b) gas phase mass and heat axial dispersions, (c) column pressure drop, (d) non-isothermal desorption, and (e) gas-solid heat transfer resistance is carried out vis a vis an idealized (most efficient hypothetically) desorption-by-purge process which is isothermal, isobaric, and where local thermodynamic equilibrium prevails. This idealized case would require the least specific amount of purge gas for any specified desorption duty, thereby providing a benchmark for other realistic desorption conditions. The cases with their legends are defined in Table 7.6.

Table 7.6: Legends for model assumptions used in various simulation cases.



Legends	Model Assumptions				
	Non-Equilibrium Thermodynamic (adsorption kinetics)	Mass Axial Dispersion in Gas	Non-Isobaric	Non-Isothermal	Gas-Solid Heat Transfer Resistance
A					
B					
C					
D					
E					
F					

a – with or without thermal axial dispersion in gas

7.5 Compare Numerical and Analytical Solutions

A rigorous analytical solution of an isothermal, isobaric, and local equilibrium desorption-by-purge model for a bulk binary Langmuirian adsorbate system (Sircar and Kumar 1985; Sircar and Golden 1995) is used to check the accuracy of numerical solutions for ideal purge desorption using model A. Listed below are the parameters and the analytical equations applied directly from Sircar and Golden (1995). For this solution, N_2=1, O_2=2, N_2 initially in the bed, $y^s = 0.79$, and N_2 in purge gas, $y^0 = 0$.

$$n_i^\infty = \frac{mb_iPy_i}{1+\ \ b_iPy_i} \qquad (7.34)$$

$$\frac{Q(y)}{Q^0} = \ \frac{(Z-\beta_1)(Z_0-\beta_2)}{(Z-\beta_2)(Z_0-\beta_1)}^{\ C} \qquad (7.35)$$

$$Z = (1 + b_2P)(1 + \lambda y) \qquad (7.36)$$

$$\lambda = \frac{b_1-b_2\ \ P}{(1+b_2P)} \qquad (7.37)$$

$$S = b_1/b_2 \qquad (7.38)$$

$$\beta_1 = \frac{1+\ \ \overline{1-4\alpha\psi}}{2\alpha} \qquad (7.39)$$

$$\beta_2 = \frac{1-\ \ \overline{1-4\alpha\psi}}{2\alpha} \qquad (7.40)$$

$$C = \frac{1}{\overline{1-4\alpha\psi}} \qquad (7.41)$$

$$\psi = (1 + b_1P)(1 + b_2P) \qquad (7.42)$$

$$\alpha = \varepsilon\rho_g/(m\rho_b) \qquad (7.43)$$

$$n_1 = my^s\ \ \alpha + \frac{S\lambda}{(S-1)(1+\lambda y^s)} \qquad (7.44)$$

$$N_1\ \ y\ \ = n_1 - \frac{m(1+\lambda)}{\lambda}\ \ 1 - \frac{1+2\lambda y}{(1+\lambda y)^2} \qquad (7.45)$$

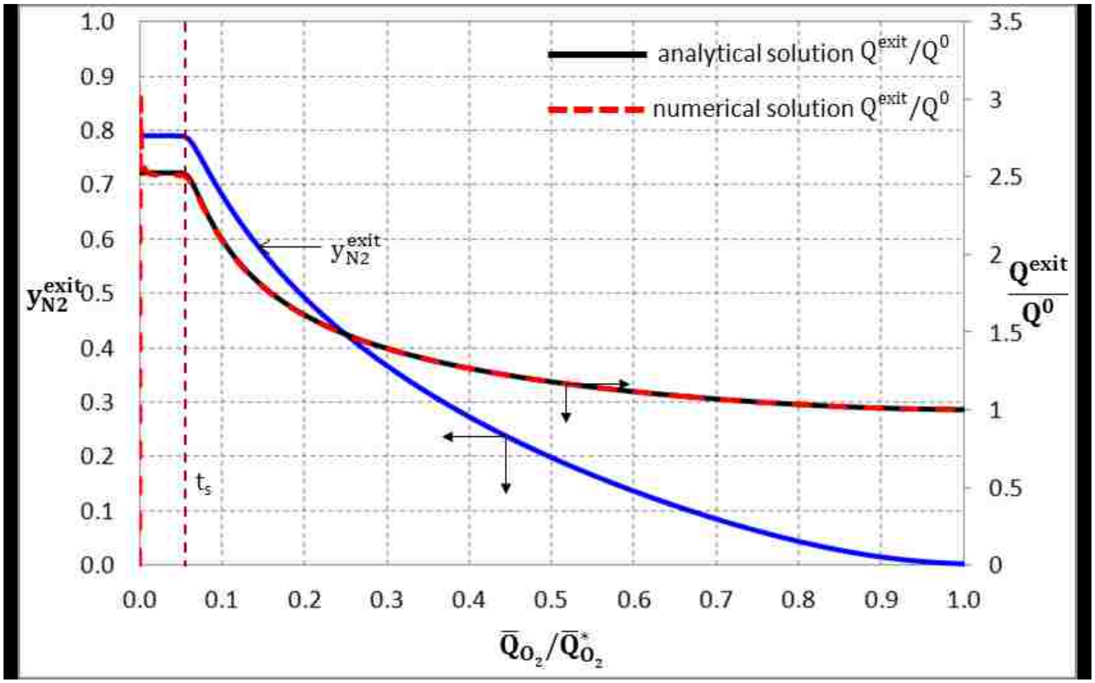


Figure 7.8: Analytical and numerical solutions of Q_{exit}/Q^0 of ideal purge desorption using numerical y_{N2}^{exit} profile as input to the analytical equations.

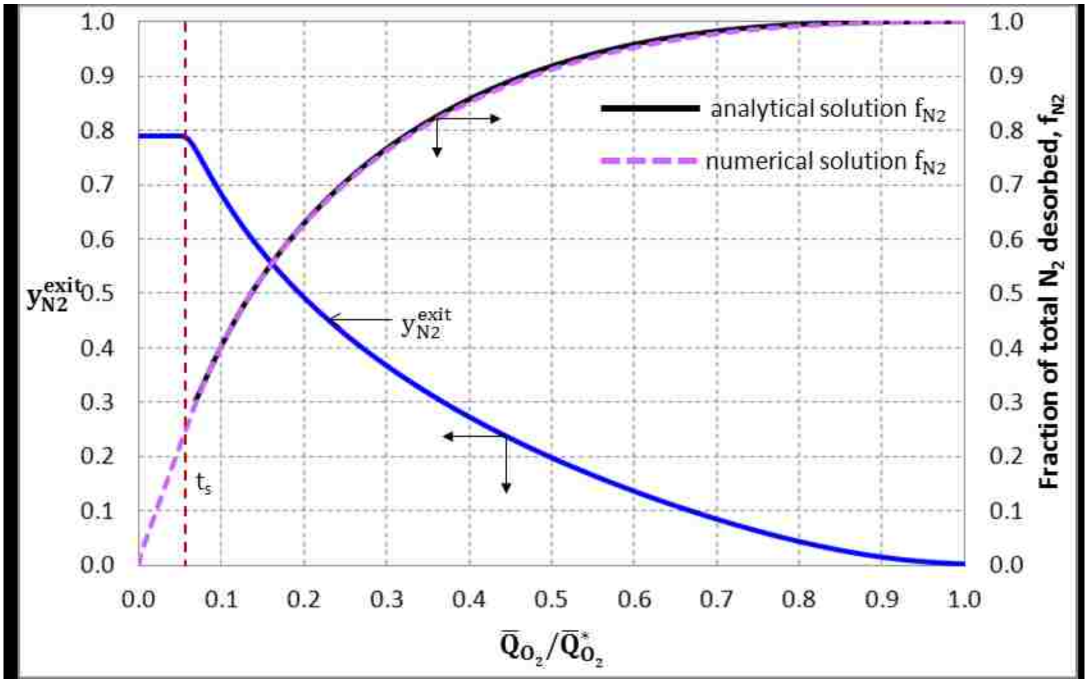


Figure 7.9: Analytical and numerical solutions of f_{N2} of ideal purge desorption using numerical y_{N2}^{exit} profile as input to the analytical equations.

Figures 7.8 and 7.9 compare the resulting analytical and numerical solutions of, respectively, fraction of gas mass flux at the column exit, Q^{exit}/Q_o , and fraction of total N_2 (adsorbed + void gas) initially present in the bed desorbed, f_{N_2}, as a function of fraction of total O_2 purge gas used, $Q_{O_2}/Q^*_{O_2}$, using numerical profile of gas phase N_2 mole fraction at the column exit, $y^{exit}_{N_2}$, as input to analytical equations.

It is observed from Figures 7.8 and 7.9 that there is slight discrepancy between the analytical and numerical solutions of Q^{exit}/Q^0 and f_{N_2} because the numerical solutions were computed using the empirical isotherms from Rege and Yang (1997), which do not meet thermodynamic consistency despite their fittings to the experimental isotherm data, while binary Langmuirian isotherms are required for the analytical solutions by Sircar and Golden (1995). Parameters $m = 2.277$, $b_1 = 0.8$, $b_2 = 0.1$ are calculated from the empirical isotherms of Rege and Yang (1997) at T = 298 K for analytical solutions. Nonetheless, these matching results confirm the numerical methods of ideal model, which is extended to other non-ideal models.

7.6 Definitions of Performance Parameters

Dynamic and overall mass balances below are used to define the performance parameters involved in purge desorption process.

Dynamic mass balance:

The total specific amount (adsorbed + void gas) of component i in the column at any z and t is given by:

$$n_i \;\; z,t \;\; = n_i \;\; z,t \;\; + \varepsilon y_i(z,t)\rho_g \;\; z,t \;\; /\rho_b \qquad \text{(mmoles/g)} \qquad (7.46)$$

Overall mass balances:

The total initial specific amount of component i in the column at the start of the purge process is given by:

$$N_i^o = n_i^o + \varepsilon y_i^o \rho_g^o/\rho_b \qquad \text{(mmoles/g)} \qquad (7.47)$$

where superscript zero defines the uniform condition present in the column at the start of the purge process.

The total specific amount of N_2 (component 1) desorbed from the column during the purge process at time t is given by:

$$N_1(t) = \frac{1}{L\rho_b} \; {}_0^t Q \;\; 0,t \;\; y_1 \;\; 0,t \;\; dt \qquad \text{(mmoles/g)} \qquad (7.48)$$

The total residual specific amount of N_2 in the column at time t is given by:

$$N_1^R(t) = N_1^o - N_1 \;\; t \;\; = \frac{1}{L} \; {}_0^L n_1(z,t)dz \qquad \text{(mmoles/g)} \qquad (7.49)$$

The total specific amount of O_2 (component 2) purge gas used at time t is given by:

$$Q_2(t) = \frac{1}{L\rho_b} \; {}_0^t Q \;\; L,t \;\; y_2 \;\; L,t \;\; dt \qquad \text{(mmoles/g)} \qquad (7.50)$$

The key variables describing the performance of the desorption process:

- Fraction of total N_2 initially present in the column (adsorbed + void gas) that is desorbed at time t: $f_1 = N_1(t)/N_1^o$ (7.51)
- Fraction of column volume at the purge inlet end which is free of N_2 (y_{N_2}< 0.0001): F (7.52)

Efficient desorption of N_2 by O_2 purge is represented by larger values of f_1 and F and smaller amount of Q_2 at the end of the purge process.

7.7 Numerical Simulations and Results

The mathematical model presented in Chapter 5 is used for simulation of desorption-by-purge step. Convective and diffusive terms were spatially discretized using biased upwind and center differences, respectively. Moreover, flux conserving formulation and Superbee flux limiter were employed. The model was solved by numerical method of lines (MOL) using Matlab's ODE solver for stiff system. Details of numerical methods are described in Chapter 6.

151 spatial nodes and 1001 – 2001 time nodes were used for all the simulation cases. Stability criteria were met by choosing the appropriate number of time nodes, and solution convergence was achieved. The validity of the numerical solution was checked against an analytical solution, as was presented in Section 7.5.

Using a workstation of Quad 2.93 GHz Intel Core i3 530 @3575 MB, the computation (CPU) time for simulating a single purge case varied between 1 – 37 minutes; cases that included gas-solid heat transfer resistance in the model requires the longest CPU times. CPU time increased drastically to 5.5 hours when spatial nodes were increased from 151 to 251.

7.7.1 Effect of Adsorbent Particle Size

Both adsorbate mass transfer (k_i) and gas-solid heat transfer (h) coefficients increase with decreasing particle size (d_p). Therefore, the effects of lowering d_p (increasing transfer coefficients) on the desorption process for all cases B – F described in Table 7.6 are studied.

Figures 7.10, 7.11 and 7.12 show the simulation results for d_p = 1500, 400 and 200 µm in adsorber L/D = 1.39, respectively. The variable f_1 is plotted against $Q_{O_2}/Q_{O_2}^*$ where $Q_{O_2}^*$ is the specific amount of O_2 purge gas needed to desorb 99.98% of N_2 from the column $f_1^* = 0.9998$ for the idealized case A. A full purge time, t_{purge}, of 2 seconds was used for all simulations except for the largest particle case where t_{purge} = 10 seconds was also used in order to evaluate the relative effect of mass transfer kinetics on faster cycle time. The corresponding mass fluxes of O_2 purge gas at the feed end, $Q_{O_2,feed}$, were, respectively, 5.35 and 1.07 mmoles/cm^2/s. All figures show the ideal desorption characteristics as a benchmark of the most efficient desorption-by-purge case.

Curves B_{10} and B_2 in Figure 7.10, with total desorption times of 10 and 2 seconds, respectively, show the detrimental effects of finite mass transfer kinetics on isothermal and isobaric desorption by purge using a relatively large adsorbent particle size (d_p = 1500 µm), which is typically used in a conventional O_2 PSA generator

(Chai et al. 2011). Faster cycle time significantly increases the amount of O_2 purge gas needed for the same degree of N_2 desorption. Mass axial dispersion in gas phase (curve C) does not have much effect on the process. Column pressure drop (curve D) also does not appreciably add to the desorption inefficiency for the large particle case. Non-isothermal (adiabatic) operation substantially reduces desorption efficiency by purge (curve E). Finally, it was found that the inclusion of a finite gas-solid heat transfer resistance as well as gas phase heat axial dispersion in desorption-by-purge model does not have much impact on overall desorption efficiency (curve F). The reason for this interesting behavior will be discussed later.

Figure 7.11 shows that the detrimental effect of mass transfer kinetics on a rapid isothermal and isobaric desorption process using a relatively large adsorbent particle size can be substantially reduced by lowering d_p to 400 μm (curve B). At the same time, the desorption inefficiency of a non-isobaric process increases significantly due to increased column pressure drop (curve D). Adiabatic operation further lowers the desorption efficiency (curve E). The gas phase mass axial dispersion, and the gas-solid heat transfer resistance do not impact the desorption process.

Figure 7.12 demonstrates that the negative effect of finite mass transfer resistance on isobaric and isothermal desorption can be essentially eliminated by using an even smaller particle size of 200 μm, but the efficiency of isothermal, non-isobaric desorption substantially suffers due to higher column pressure drop. As expected, non-isothermal desorption further reduces the overall desorption efficiency.

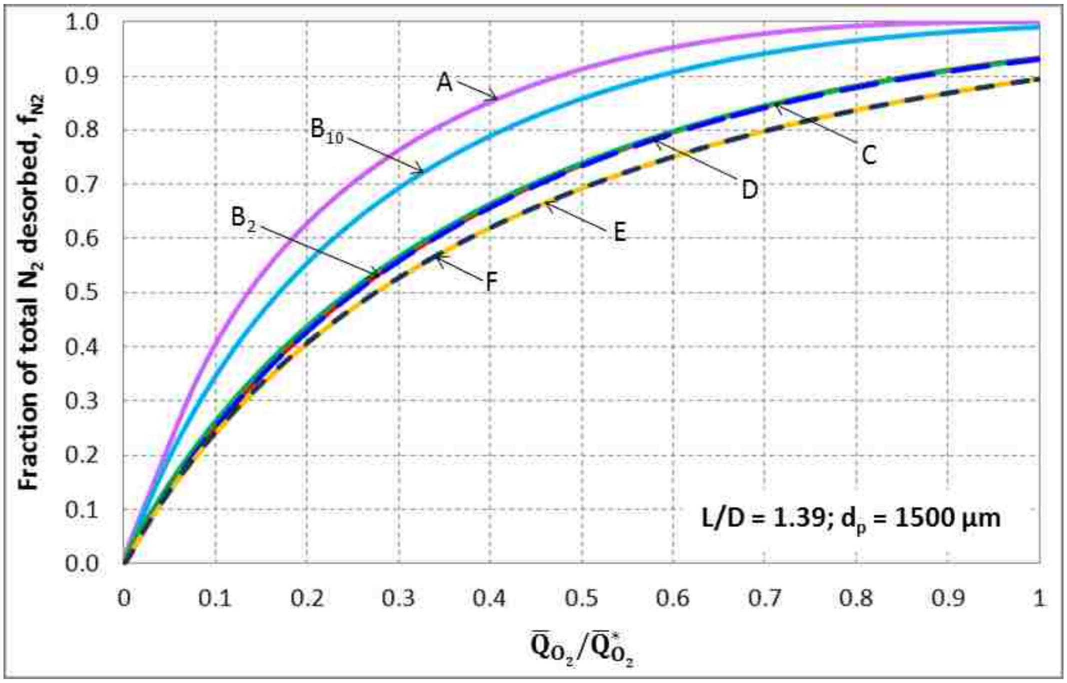


Figure 7.10: Fraction of total N_2 desorbed vs amount of O_2 purge gas used. L/D = 1.39, d_p= 1500 µm, k_{N_2}= 5.3 s^{-1}, k_{O_2}= 5.1 s^{-1}, D_L ~ 24 cm^2/s, D_g ~ 0.22 cm^2/s, Nu ~ 20.4, Re ~ 144, ha ~ 0.21 cal/cm^3/s/K, $Q_{O_2,feed}$ = 5.35, 1.07 mmoles/cm^2/s, respectively, for t_{purge} = 2, 10 s. Legends: Table 7.6.

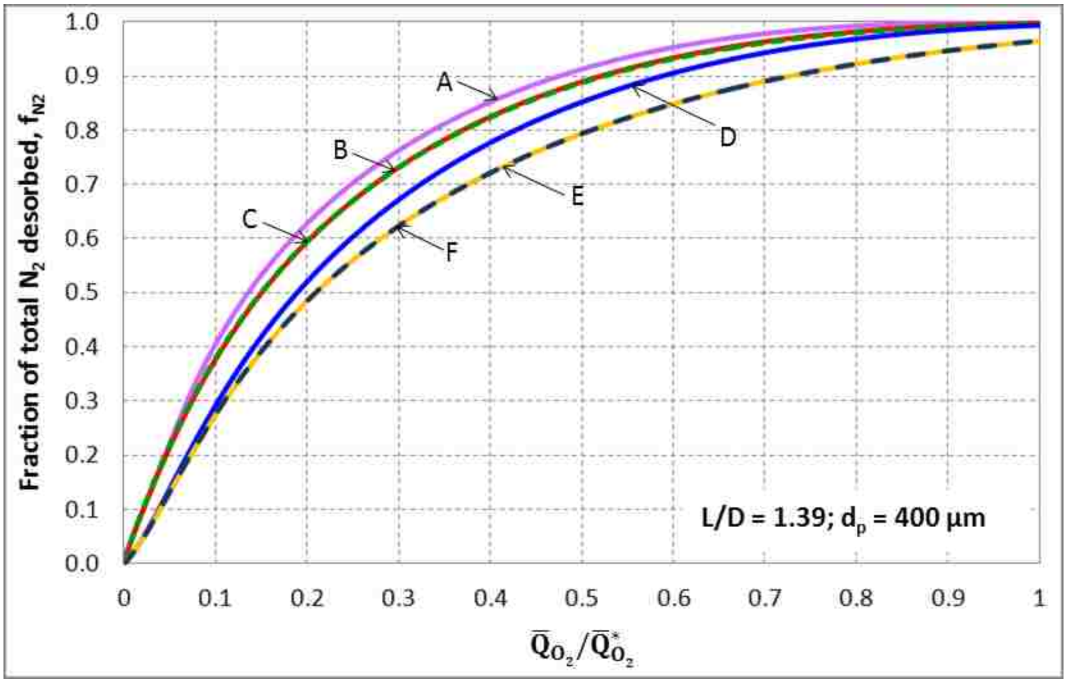


Figure 7.11: Fraction of total N_2 desorbed vs amount of O_2 purge gas used. L/D = 1.39, d_p= 400 µm, k_{N_2}= 73.8 s^{-1}, k_{O_2}= 71.4 s^{-1}, D_L ~ 5.9 cm^2/s, D_g ~ 0.20 cm^2/s, Nu ~ 10.3, Re ~ 38, ha ~ 1.49 cal/cm^3/s/K, $Q_{O_2,feed}$ = 5.35 mmoles/cm^2/s, t_{purge} = 2 s. Legends: Table 7.6.

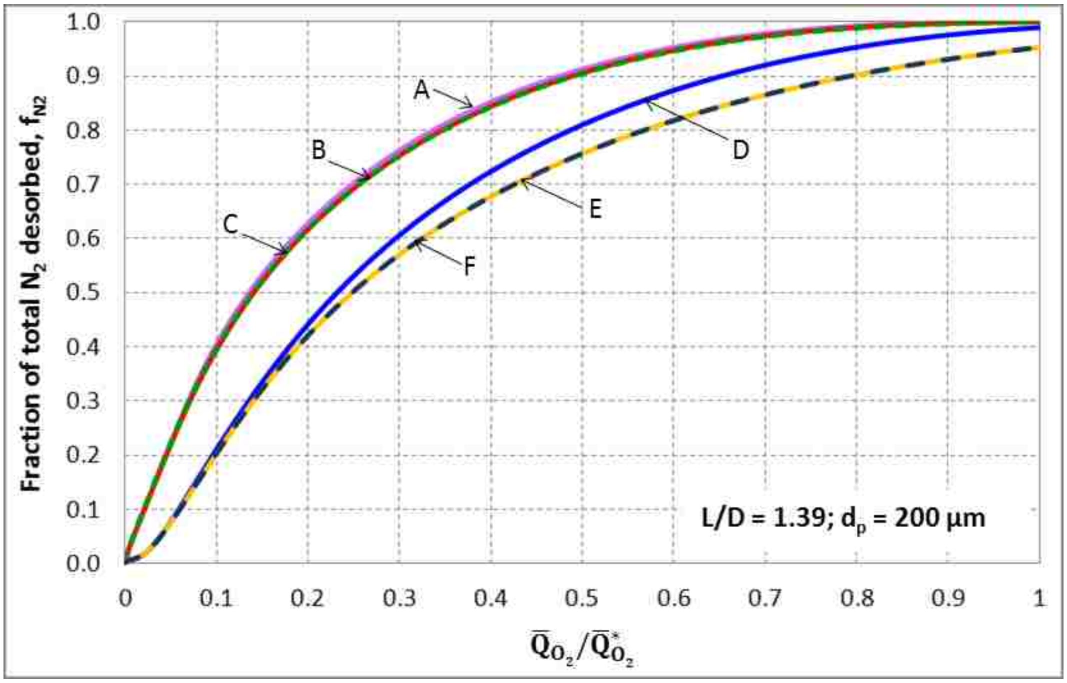


Figure 7.12: Fraction of total N_2 desorbed vs amount of O_2 purge gas used. L/D= 1.39, d_p= 200 μm, k_{N_2}= 295 s^{-1}, k_{O_2}= 285 s^{-1}, D_L ~ 2.6 cm^2/s, D_g ~ 0.18 cm^2/s, Nu ~ 7.5, Re ~ 19, ha ~ 4.3 cal/cm^3/s/K, $Q_{O_2,feed}$ = 5.35 mmoles/cm^2/s, t_{purge} = 2 s. Legends: Table 7.6.

Figures 7.13 and 7.14 plot the local total loading of N_2 (adsorbed + void) inside the column $n_{N_2}\ z,t$ relative to the initial total N_2 loading $N_{N_2}^{o}$ as functions of dimensionless column distance z/L at the end of the purge desorption process when the fractions of total N_2 desorbed from the column were, respectively, 85% (f_{N_2}= 0.85) and 90% (f_{N_2}= 0.90). It should be noted that the durations of purge processes vary for these duties depending on particle sizes as well as model complexities. They show the locations of the trailing edges ($y_{N_2} < 0.0001$) of the proportionate patterned desorption fronts at the end of the purge process for different adsorbent particle sizes and desorption times. Thus, these figures give the fractions (F) of the N_2-free adsorber volumes at the purge inlet end. The figures also provide the corresponding loading profiles for the idealized purge case A.

It may also be seen from Figure 7.13 that a very small percentage of the column is free of N_2 (F ~ 1%) when the adsorbent particle size is 1500 μm and the mass flux of O_2 purge gas, $Q_{O_2,feed}$, is high. Decreasing $Q_{O_2,feed}$ enlarges F to ~ 11%. Reducing the adsorbent particle size for a rapid purge process increases F; the model estimated values of F are, respectively, ~ 21% and ~ 27% for 400 and 200 μm particle sizes. The corresponding F value for the ideal purge case is ~ 30%.

Figure 7.14 shows that a particle size of 1500 μm cannot achieve f_{N_2} = 0.90 in a rapid purge desorption situation using $Q_{O_2,feed}$ = 5.35 mmoles/cm^2/s with $t_{purge} \leq$ 2 s. On the other hand, smaller particle sizes of 400 and 200 μm can be used to achieve f_{N_2} = 0.90 and obtain, respectively, ~ 28% and ~ 35% N_2-free column fractions under identical conditions. The corresponding F value for the ideal purge case is ~ 37%. Another interesting observation from Figures 7.13 and 7.14 is that the loading profiles for 400 and 200 µm particles are not very different in most of the column length for the fast desorption cases.

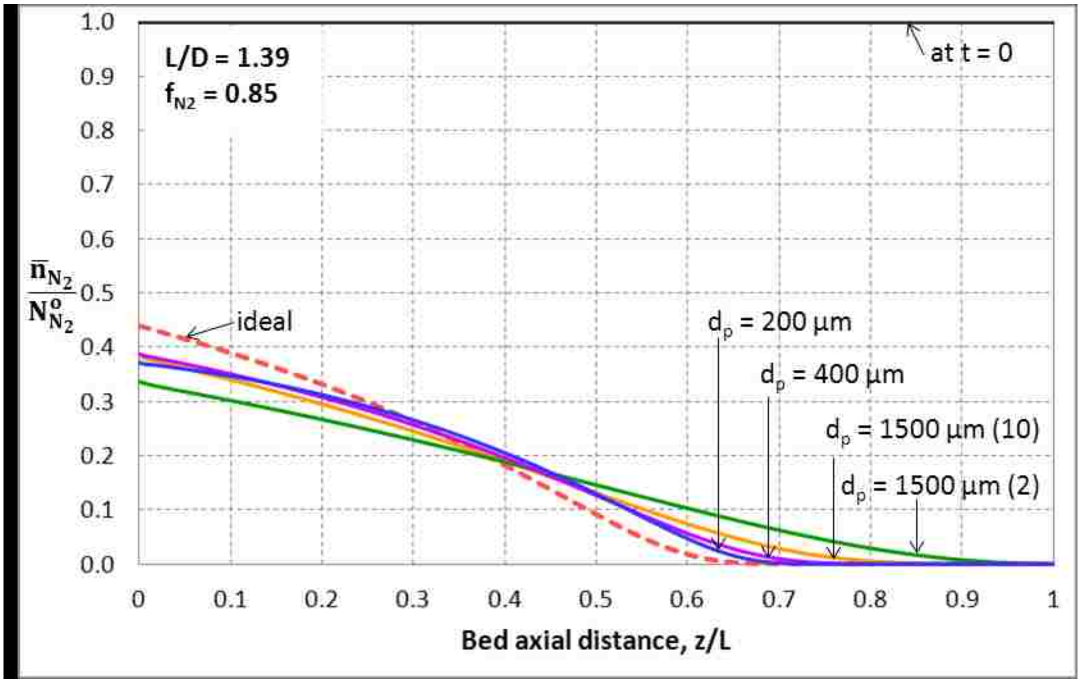


Figure 7.13: Fraction of total N_2 loadings vs bed axial distance for f_{N_2}= 0.85. L/D=1.39, $Q_{O_2,feed}$ = 5.35, 1.07 (10) mmoles/cm^2/s, purge durations for achieving f_{N_2}= 0.85: 6 s, 1.68 s, 1.21 s, 1.33 s, and 0.80 s, respectively, for d_p= 1500 (10), 1500 (2), 400, 200 µm, and ideal case A. Dashed line: ideal case; Solid lines: case F (Table 7.6).

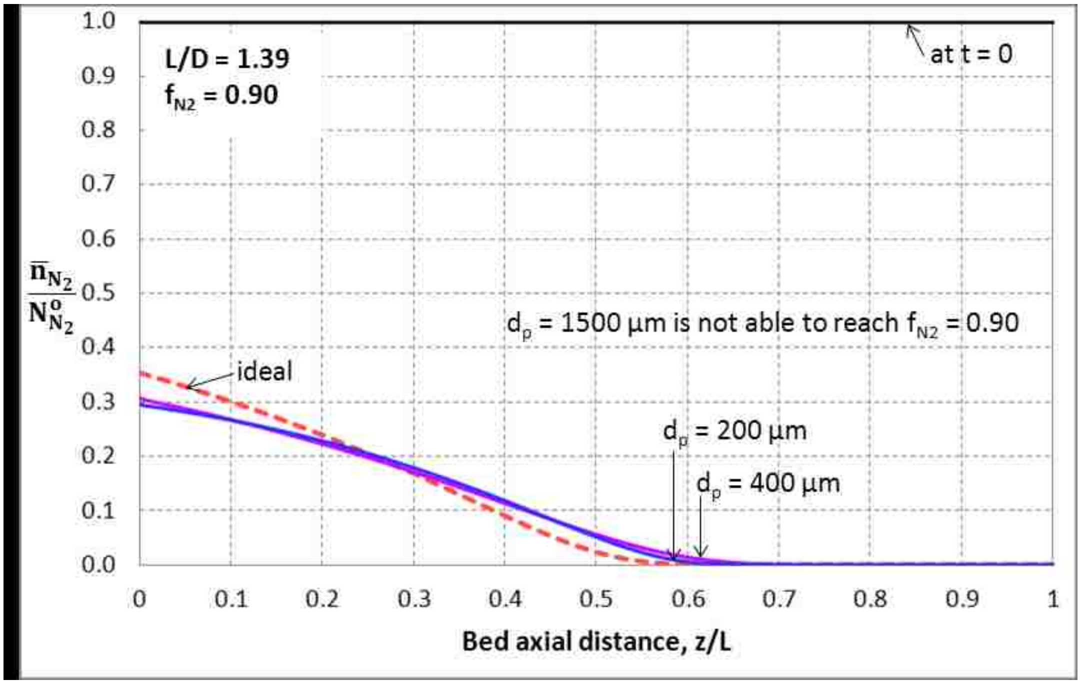


Figure 7.14: Fraction of total N_2 loadings vs bed axial distance for f_{N_2}= 0.90. L/D=1.39, $Q_{O_2,feed}$ = 5.35 mmoles/cm^2/s, purge durations for achieving f_{N_2}= 0.90: 1.46 s, 1.59 s, and 0.95 s, respectively, for d_p= 400, 200 μm, and ideal case. Dashed line: ideal case A; Solid lines: case F (Table 7.6).

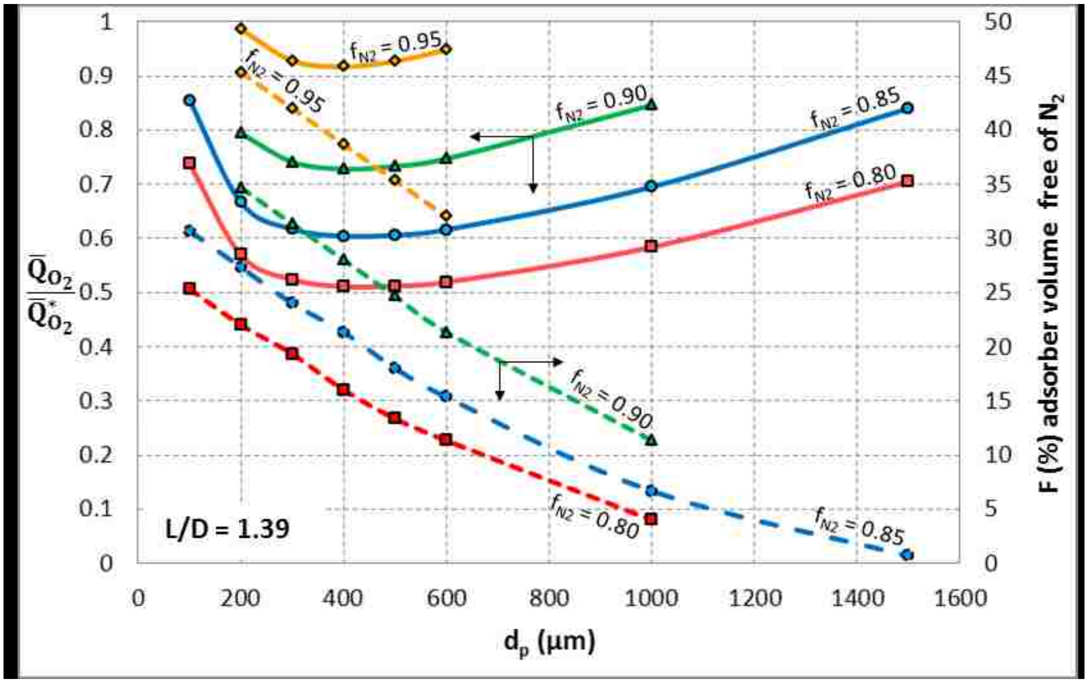


Figure 7.15: Amount of O_2 purge required to desorb different fractions of total N_2 from a column (solid lines) and the corresponding fractional adsorbent volume free of N_2 at the purge inlet end (dashed lines) using different d_p. L/D = 1.39, $Q_{O_2,feed}$ = 5.35 mmoles/cm^2/s, $t_{purge} \leq 2$ s, case F (Table 7.6).

Figure 7.15 is a comprehensive summary of the simulated desorption performance data. It plots the dimensionless quantity of O_2 purge gas $Q_{O_2}/Q^*_{O_2}$ used in a rapid desorption process of $Q_{O_2,feed}$ = 5.35 mmoles/cm^2/s with $t_{purge} \leq 2$ s as a function of adsorbent particle size d_p for different extents of total N_2 desorption $f_{N_2} = 0.8, 0.85, 0.9, 0.95$ as well as the corresponding fractions of column volume free of N_2 at the purge inlet end F. It should be mentioned that larger particles cannot produce a positive value of F when f_{N_2} is low, and high f_{N_2} may not be achievable by very large or small particles.

A very important observation is that the purge gas requirement goes through a shallow minimum value for all values of f_{N_2}. The adsorbent particle size range where the minimum occurs is ~ 300 – 500 μm. The purge gas quantity for all values of f_{N_2} increases when d_p < 300 μm and when d_p > 500 μm, primarily due to detrimental effects of increased pressure drop and increased mass transfer resistance, respectively, as described earlier by Figures 7.10 – 7.12. It may also be seen from Figure 7.15 that F monotonically increases with decreasing d_p for all values of f_{N_2}. However, smaller d_p demands larger amount of O_2 purge.

An important finding of this analysis is that an optimum adsorbent particle size exists which will provide the best performance (lower BSF and higher O_2 recovery) by a MOC design if the RPSA cycle involves an explicit 'back purge with product gas' step.

A previous experimental and model study of a RPSA process for production of O_2 enriched air using a two-step cycle (rapid column pressurization with air and rapid countercurrent depressurization of the column with continuous withdrawal of the product gas) has shown that the product O_2 purity can be maximized by using an optimum particle size (Alpay et al. 1994). This RPSA cycle does not contain an explicit and distinct 'product back purge step' but it implicitly uses a non-isobaric back purge with enriched O_2 gas during the depressurization step.

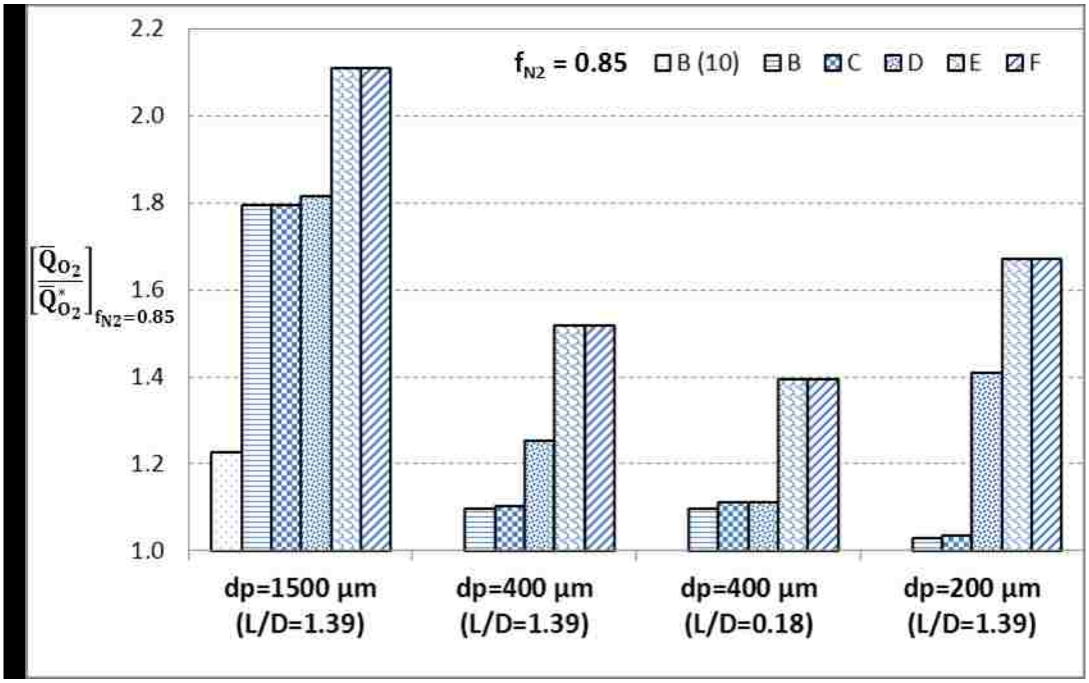


Figure 7.16: Histograms showing the relative contributions of increase in O_2 purge gas amount due to different non-idealities (Table 7.6) for three different adsorbent particle sizes, f_{N_2} = 0.85, L/D = 1.39, 0.18. Base line = 1 (ideal case A).

Figure 7.16 shows histograms describing progressive increases in O_2 purge gas quantities, or consequently, increases in inefficiency of the desorption process, above that for the idealized desorption case A (represented by base line = 1) due to inclusion of various non-idealities (B – F) listed in Table 7.6 using adsorbent particle sizes of 1500, 400 and 200 µm. The total fraction of N_2 desorbed from the adsorber for each case was 85%, and the L/D ratio of the column was 1.39. Mass flux of O_2 purge gas introduced at the feed end, $Q_{O_2,feed}$, were, 5.35 and 1.07 mmoles/cm^2/s for $t_{purge} \leq 2$ s and $t_{purge} \leq 10$ s. The following conclusions can be drawn from the figure:

a) Slow adsorption kinetics, column pressure drop, and non-isothermal desorption are the main causes of increased purge gas quantity for a specific desorption duty.

b) Larger purge gas amount is required for a rapid desorption process than a conventional desorption process for the same extent of N_2 desorption.

c) The gas phase mass axial dispersion does not affect the desorption efficiency.

d) A finite heat transfer resistance between the gas and the solid does not have much effect on the overall desorption efficiency.

e) The adverse effect of adsorption kinetics for a rapid desorption process can be significantly reduced by using smaller particle sizes, but at the cost of increased pressure drop, which eventually turns the desorption process inefficient below a certain particle size.

7.7.2 Pancake Adsorber

PSA processes are often designed using a low L/D ratio which increases the column cross-sectional area and decreases the gas mass flux, and hence, column pressure drops. The desorption performance of a pancake (L/D = 0.18) adsorber containing the same amount of adsorbent as in the simulation cases above is simulated. This reduced the gas mass flux by a factor of ~ 4. The adsorbent particle size was 400 μm, $Q_{O_2,feed}$ = 1.35 mmoles/cm^2/s, and $t_{purge} \leq 2$ s. Figure 7.17 shows the resulting plots of f_{N_2} vs $Q_{O_2}/Q_{O_2}^*$ for cases A – F (cases defined in Table 7.6), and Figure 7.18 compares the overall desorption profiles between conventional and pancake type adsorbers. A histogram of relative contributions of different non-idealities on the purge gas amount for the pancake adorber is also described in Figure 7.16 which indicates that the detrimental effect of column pressure drop, and hence, the relative overall inefficiency of the purge process is decreased by using a pancake adsorber design compared to a common adsorber. On the other hand, Figure 7.16 shows that the detrimental effects of non-isothermality and mass transfer resistance using particle size of 400 μm are still apparent. The use of small particle size with a pancake adsorber design may lower purge gas amount but hydrodynamic problems like gas mal-distribution, particle agglomeration, etc., can become serious problems (Porter et al. 1993; Moulijn and Van Swaaij 1976).

Figure 7.19 shows the plots of dimensionless total N_2 loadings vs distance in the column at the end of the purge desorption process when 85% of N_2 was desorbed from the column. The figure compares these profiles for the ideal desorption case A and those for the common (L/D = 1.39) and pancake (L/D = 0.18) adsorber designs using case F. It may be seen that use of a lower L/D reduces the fraction of N_2-free adsorber volume (F) at the purge inlet end. The values of parameter F were, respectively, 0.13 and 0.21 for L/D values of 0.18 and 1.39. The corresponding value of F for the ideal purge case was 0.3. This can be considered to be a negative characteristic of a pancake adsorber.

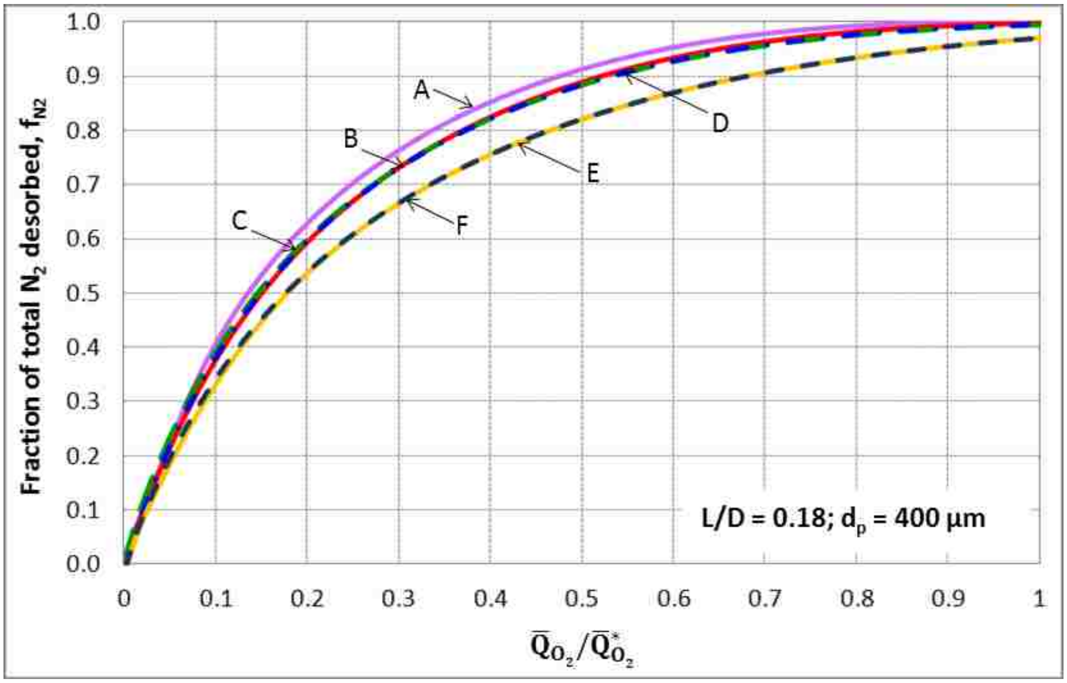


Figure 7.17: Fraction of total N_2 desorbed vs amount of O_2 purge gas used. L/D = 0.18, d_p= 400 µm, k_{N_2}= 73.8 s^{-1}, k_{O_2}= 71.4 s^{-1}, D_L ~ 1.6 cm^2/s, D_g ~ 0.22 cm^2/s, Nu ~ 5.62, Re ~ 9.6, ha ~ 0.81 cal/cm^3/s/K, $Q_{O_2,feed}$ = 1.35 mmoles/cm^2/s, t_{purge} = 2 s. Legends: Table 7.6.

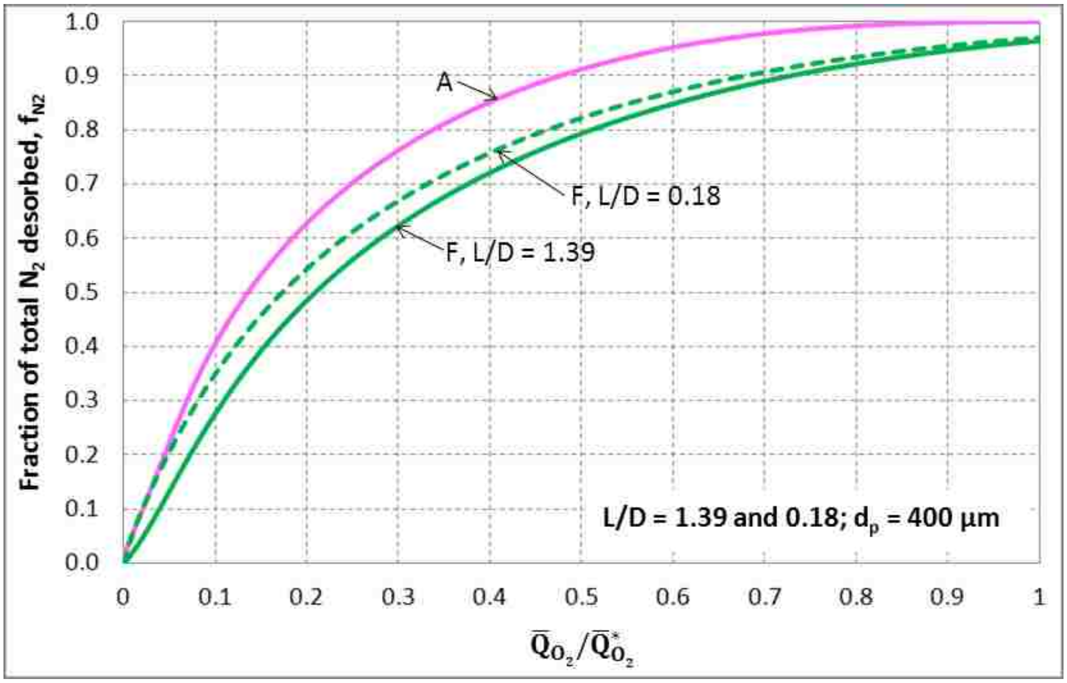


Figure 7.18: Effects of column L/D on the efficiency of desorption of total N_2 by O_2 purge. d_p = 400µm, $Q_{O_2,feed}$ = 1.35 (L/D = 0.18), 5.35 (L/D = 1.39) mmoles/cm^2/s, t_{purge} = 2 s. Legends: cases A and F (Table 7.6).

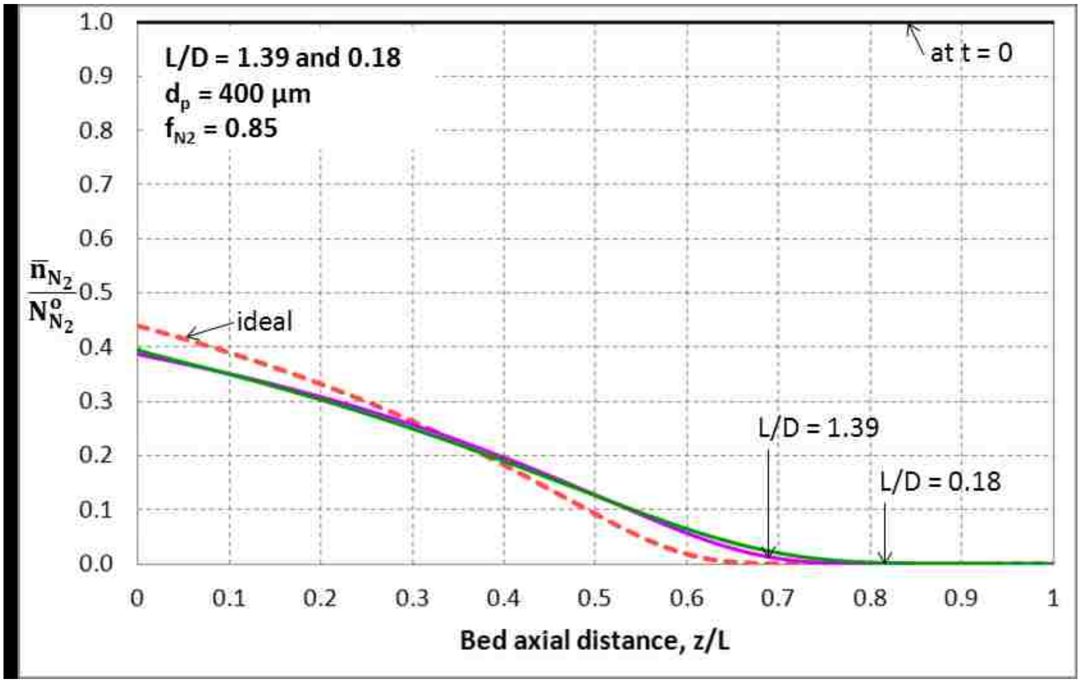


Figure 7.19: Effects of column L/D on total N_2 loadings in column for f_{N_2}= 0.85. d_p = 400 µm, $Q_{O_2,feed}$ = 1.35 (L/D = 0.18), 5.35 (L/D = 1.39) mmoles/cm^2/s, purge durations for achieving f_{N_2}= 0.85: 1.11 s, 1.21 s, and 0.80 s, respectively, for pancake adsorber, common adsorber, and ideal case. Dashed line: ideal case; Solid lines: case F (Table 7.6).

7.7.3 Non-Isothermal Column Behavior for Adiabatic Desorption-by-Purge

Figure 7.20 shows the simulated gas (dashed lines) and adsorbent (solid lines) temperature profiles as functions of dimensionless distance inside an adiabatic adsorber at the end of a rapid purge step for the case where the fraction of total N_2 desorbed from the column during the process was 0.85, the column L/D was 1.39, the adsorbent particle size was 400 µm, and the required duration of purge for achieving f_{N_2}= 0.85 was ~ 1.21 seconds. The values of gas-solid heat transfer coefficients (ha) used in the simulation were 1.49 (curve 2) and 0.53 (curve 3) cal/cm^3/s/K depending on the correlation used. The figure also shows the temperature profiles for the case where instantaneous thermal equilibrium ($ha = \infty$) between the gas and the solid prevailed (curve 1). The profiles of gas phase mole fractions of N_2 in the column at the end of the purge process are also shown in the figure.

It may be seen from Figure 7.20 that a sharp dip forms in both gas and solid phase temperature profiles near the purge gas inlet end of the column which reaches a minimum temperature difference θ of ~ -14.8 K, followed by a plateau region, and then both temperatures slowly increase towards the purge gas exit end. This is caused by self-cooling of the adsorbent and gas in order to supply the endothermic heat of desorption of N_2 from the sensible heat of the gas and the solid during the purge process, since the purge gas inlet temperature is same as that of the initial column temperature. The entire column gets colder than the starting condition during the rapid purge process. An equilibrium theory for adiabatic desorption discusses the formation of a temperature plateau region (Sircar and Kumar 1985).

Three interesting observations from Figure 7.20 are that (a) the gas temperature is slightly higher [more so when ha is smaller for obvious reasons] than the solid temperature at the purge gas inlet end until they reach their minimum plateau values, and then they become approximately equal for most of the remaining column length ($0 < z/L < 0.85$), (b) the existence of a minimum in the gas phase temperature profile means that heat flows by thermal axial dispersion in the gas phase towards the location of the temperature minimum from both ends of the column, and (c) the column temperature profiles for cases where ha has a finite value crisscross the corresponding profile for local thermal equilibrium case.

Figure 7.21 shows the column temperature profiles for the same desorption system employing a pancake adsorber (L/D = 0.18). The temperature profiles for the pancake adsorber are similar to those for the common adsorber when the gas-solid heat transfer coefficient is relatively large (ha = 0.81 ca/cm^3/s/K) except that the size of the plateau region is smaller. However, the differences between the gas and solid temperatures are relatively larger throughout the entire length of the column and a well-defined temperature plateau is not even formed when ha is sufficiently low (ha = 0.065 cal/cm^3/s/K) for the pancake adsorber. In addition, the temperature minimum occurs at a location much deeper inside the column from the purge inlet end for the pancake adsorber. These complex temperature profiles, which are governed by the mass and heat transfer kinetics and the column pressure drop, determine the local adsorbate loadings during desorption, and hence the overall desorption characteristics of the column in a rapid, non-isothermal process. They can only be studied by a detailed non-isothermal, non-equilibrium, and non-isobaric desorption processes model. Experimental measurement of the profiles shown by Figures 7.20 and 7.21 may be tedious and not even be practically feasible.

The simulations showed that the integrated overall N_2 desorption characteristics depicted by Figures 7.10 – 7.12 and Figures 7.16 – 7.17 were not affected appreciably by the presence of a finite gas-solid heat transfer resistance despite the differences in the gas and solid temperature profiles near the purge inlet section of the column when ha was low. This was because the gas and solid temperature profiles inside the column were not much different for most of the column length at the end of desorption process irrespective of the value of ha for all cases. This point is elaborated in the next section.

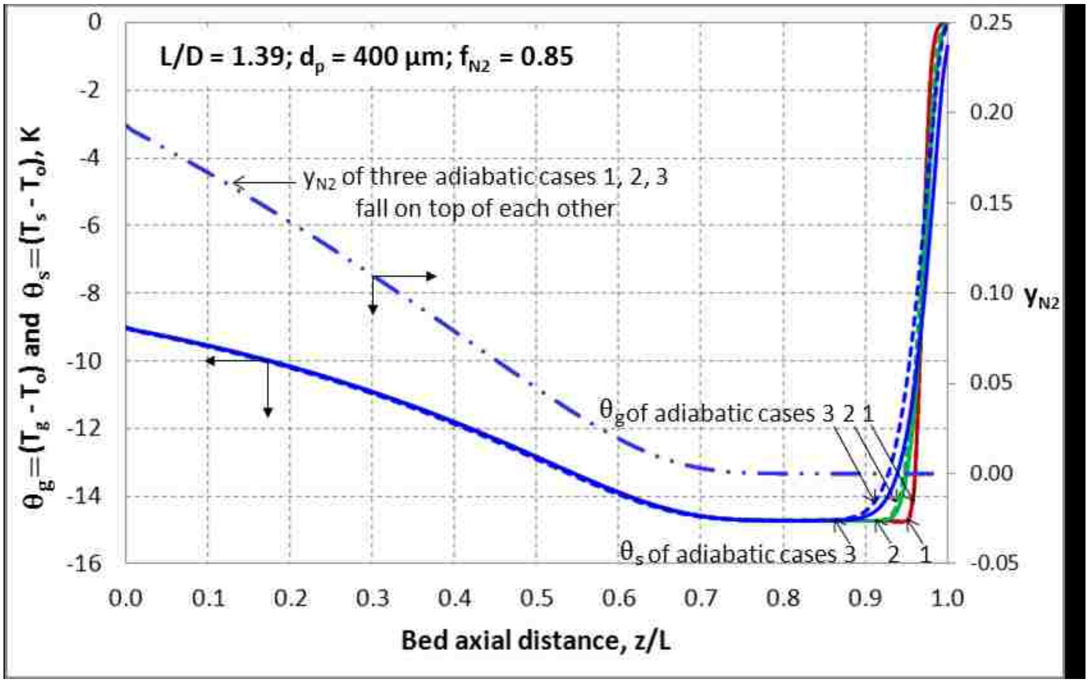


Figure 7.20: Solid and Gas temperature and gas phase N_2 mole fraction profiles in column at the end of purge extent f_{N_2} = 0.85. L/D = 1.39, d_p = 400 µm, $Q_{O_2,feed}$ = 5.35 mmoles/cm^2/s, all required ~ 1.21 s of purge duration for achieving f_{N_2} = 0.85, gas-solid heat transfer coefficients (ha = cal/cm^3/s/K): ∞ (curve 1, equilibrium), 1.49 (curve 2, Wakao), 0.52 (curve 3, Kunii Suzuki).

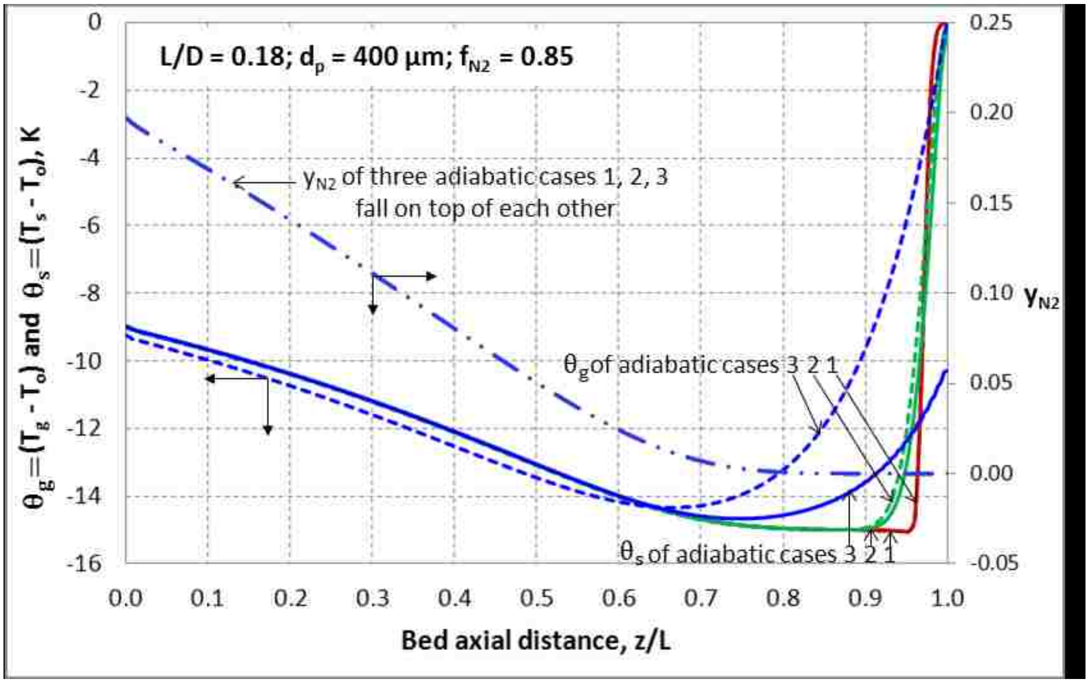


Figure 7.21: Solid and Gas temperature and gas phase N_2 mole fraction profiles in column at the end of purge extent f_{N_2} = 0.85. L/D = 0.18, d_p = 400 µm, $Q_{O_2,feed}$ = 1.35 mmoles/cm^2/s, all required ~ 1.11 s of purge duration for achieving f_{N_2} = 0.85, gas-solid heat transfer coefficients (ha = cal/cm^3/s/K): ∞ (curve 1, equilibrium), 0.81 (curve 2, Wakao), 0.065 (curve 3, Kunii Suzuki).

7.7.4 Parametric Study of the Effect of a Finite Gas-Solid Heat Transfer Coefficient

The effects of extreme values of gas-solid heat transfer coefficients (ha = 0, finite, and ∞ cal/cm^3/s/K) on the desorption process in system

(a) L/D = 1.39, d_p = 1500 μm, $t_{purge} \leq 2$ s

(b) L/D = 0.18, d_p = 400 μm, $t_{purge} \leq 0.2$ s

are parametrically simulated. Simulation using conditions (a) is mainly for investigating the effect of gas-solid heat transfer resistance caused by large particle in common adsorber; on the other hand, simulation using conditions (b) allow us to see the effect of local thermal non-equilibrium caused by ultra-rapid purge step in pancake adsorber. The simulation results for (a) and (b) are given, respectively, by Figures 7.22 – 7.24 and Figures 7.25 – 7.27.

Figure 7.22 shows the gas and solid temperature profiles in the column at the end of the purge process of f_{N_2}= 0.85 using different values of ha, with and without thermal axial dispersion. It bears out a general observation of this study that the magnitude of a finite gas-solid heat transfer coefficient affects the gas and solid temperature profiles in the column only in the purge gas inlet section. The effect is small on the temperature profiles in the rest of the column. In fact, the gas and solid temperature profiles are practically independent of the value of ha in most of the column length at the end of the purge process. Furthermore, the total N_2 loading and N_2 gas phase mole fraction profiles in the column for all of the adiabatic cases of Figure 7.22 practically overlap on each other as shown by Figure 7.23. As a consequence of this behavior, Figure 7.24 shows that the overall integrated N_2 desorption profiles corresponding to different temperature profiles of Figure 7.22 also overlap indicating that ha has practically no effect on these profiles for the present system.

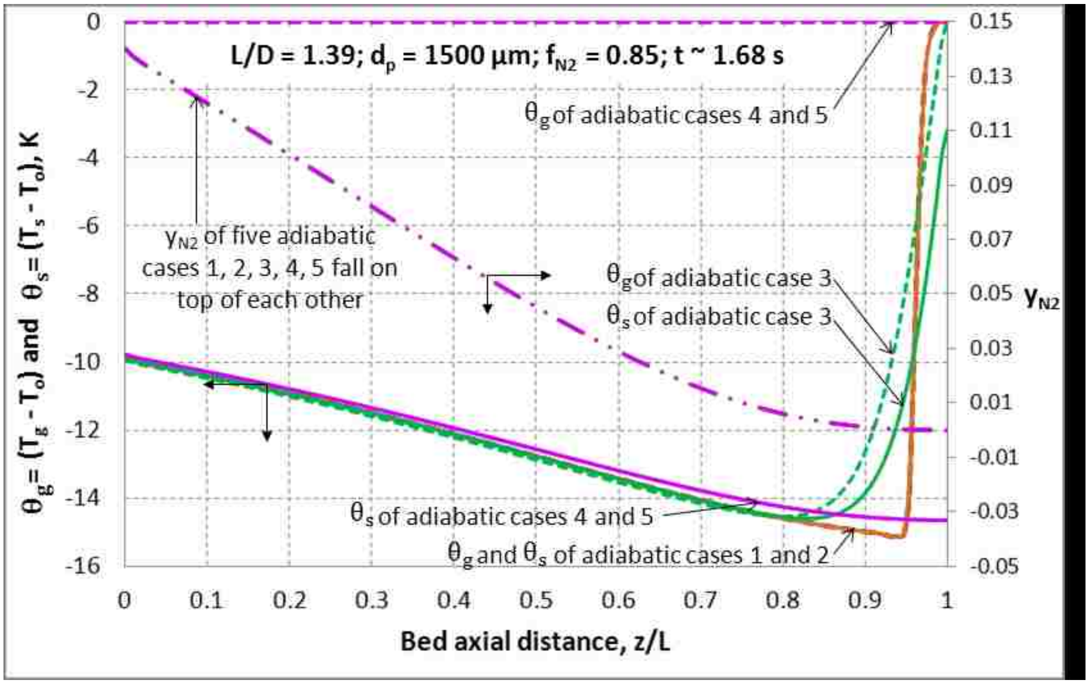


Figure 7.22: Solid and Gas temperature and gas phase N_2 mole fraction profiles in column at the end of purge extent f_{N_2} = 0.85. L/D = 1.39, d_p = 1500 µm, $Q_{O_2,feed}$ = 5.35 mmoles/cm^2/s, all required ~ 1.68 s of purge duration for achieving f_{N_2} = 0.85, gas-solid heat transfer coefficients (ha = cal/cm^3/s/K): ∞ (curves 1 and 2, with and without thermal gas axial dispersion), 0.21 (curve 3, Wakao with thermal gas axial dispersion), 0 (curves 4 and 5, with and without thermal gas axial dispersion).

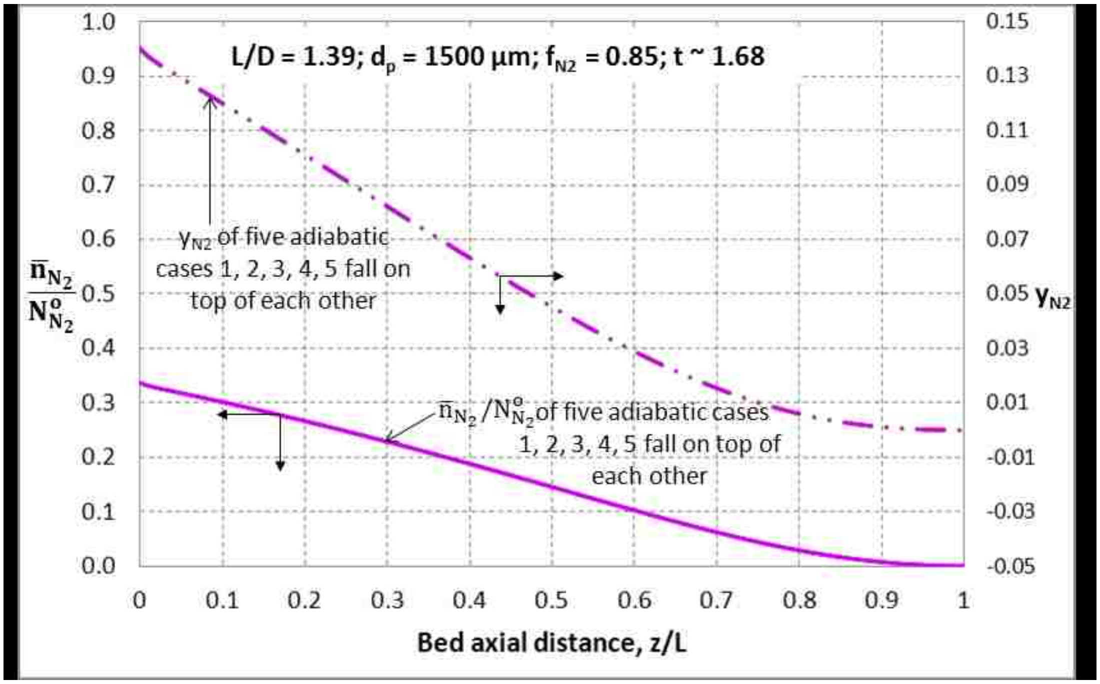


Figure 7.23: Overlapping of total N_2 loadings (solid lines) and N_2 gas phase mole fraction (dashed lines) profiles in column at the end of purge extent f_{N_2} = 0.85. L/D = 1.39, d_p = 1500 µm, $Q_{O_2,feed}$ = 5.35 mmoles/cm^2/s, required ~ 1.68 s of purge duration for achieving f_{N_2} = 0.85, for all cases of Figure 7.22.

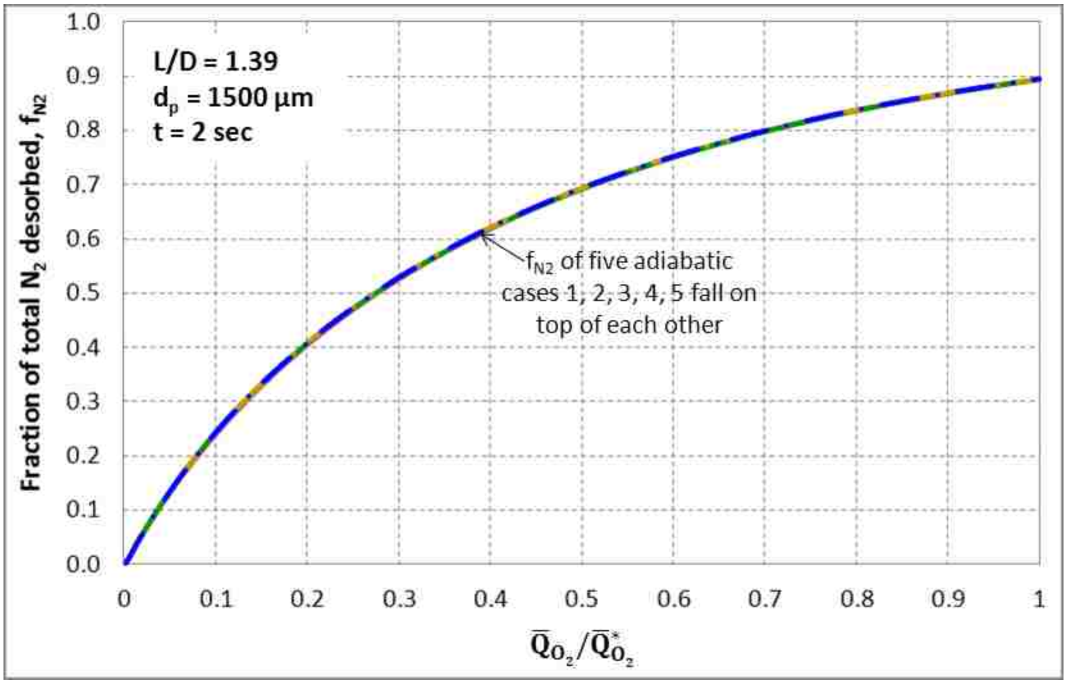


Figure 7.24: Fraction of total N_2 desorbed vs purge O_2 quantity for all cases of Figure 7.22.

Figures 7.25 – 7.27 show similar phenomenon as those in Figures 7.22 – 7.24. With temperature profiles vary only at the purge inlet region but remain essentially same for the rest of the column length as shown in Figure 7.25, gas-solid heat transfer coefficients ha have no effect on N_2 gas phase mole fraction and total N_2 loading profiles as shown in Figure 7.26. The result is overlapping of the overall integrated N_2 desorption profiles in Figure 7.27. It is also observed from Figures 7.22 – 7.24 and Figures 7.25 – 7.27 that inclusion or exclusion of thermal axial dispersion is not critical in the present purge desorption process.

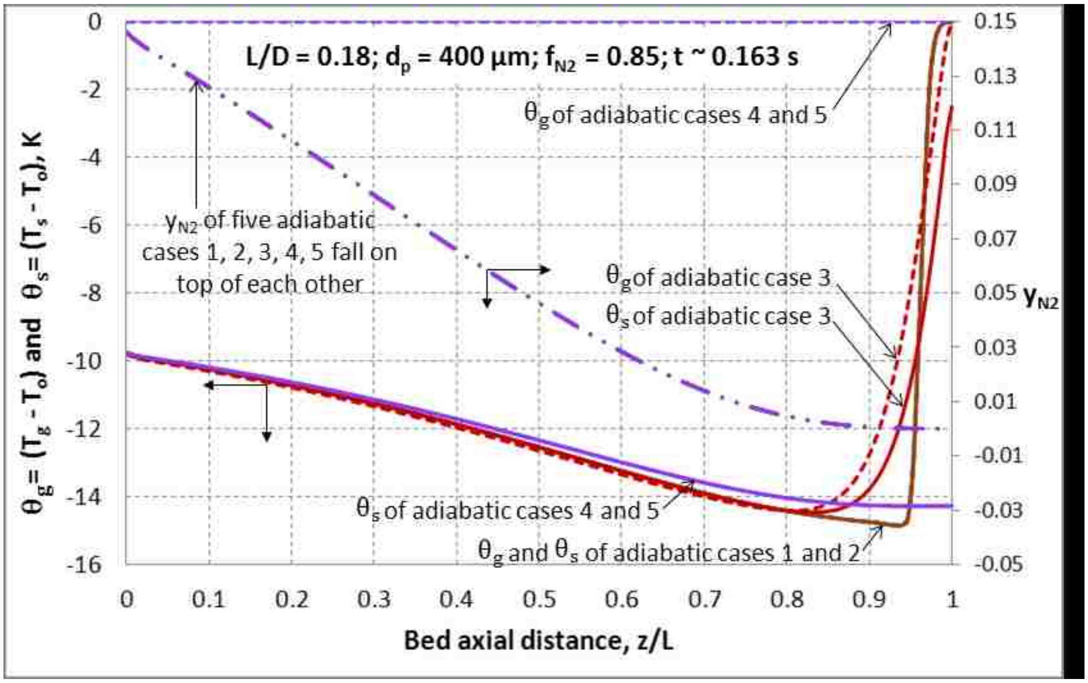


Figure 7.25: Solid and Gas temperature and gas phase N_2 mole fraction profiles in column at the end of purge extent f_{N_2} = 0.85. L/D = 0.18, d_p = 400 µm, $Q_{O_2,feed}$ = 13.53 mmoles/cm^2/s, all required ~ 0.163 s of purge duration for achieving f_{N_2} = 0.85, gas-solid heat transfer coefficients (ha = cal/cm^3/s/K): ∞ (curves 1 and 2, with and without thermal gas axial dispersion), 2.40 (curve 3, Wakao with thermal gas axial dispersion), 0 (curves 4 and 5, with and without thermal gas axial dispersion).

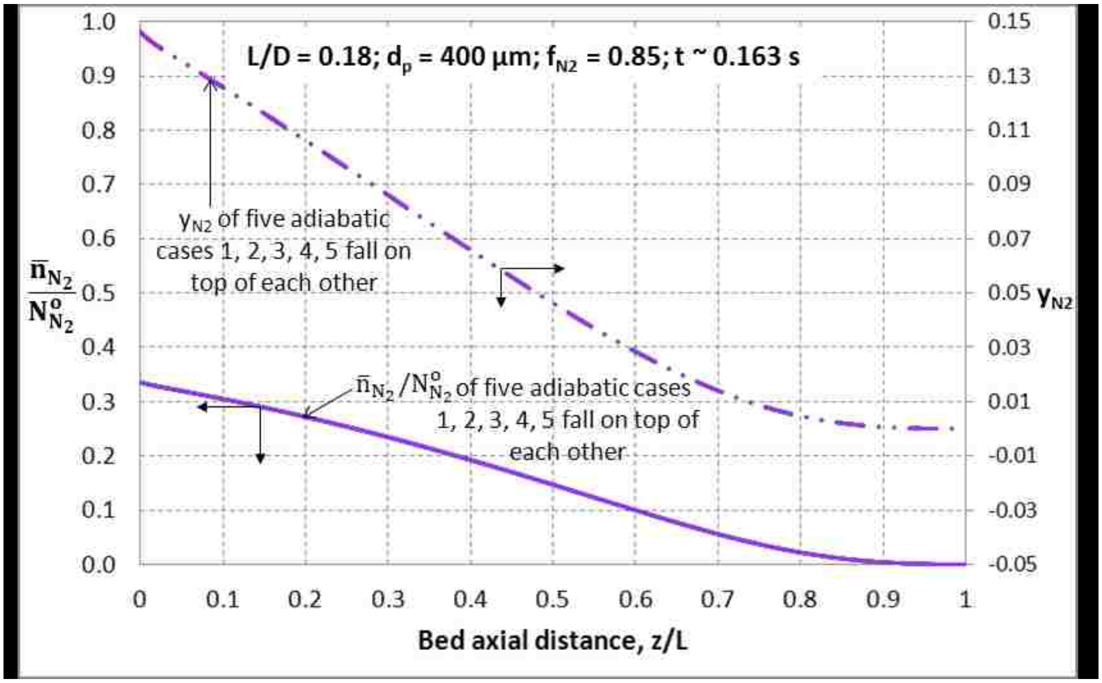


Figure 7.26: Overlapping of total N_2 loadings (solid lines) and N_2 gas phase mole fraction (dashed lines) profiles in column at the end of purge extent f_{N_2} = 0.85. L/D = 0.18, d_p = 400 µm, $Q_{O_2,feed}$ = 13.53 mmoles/cm^2/s, required ~ 0.163 s of purge duration for achieving f_{N_2} = 0.85, for all cases of Figure 7.25.

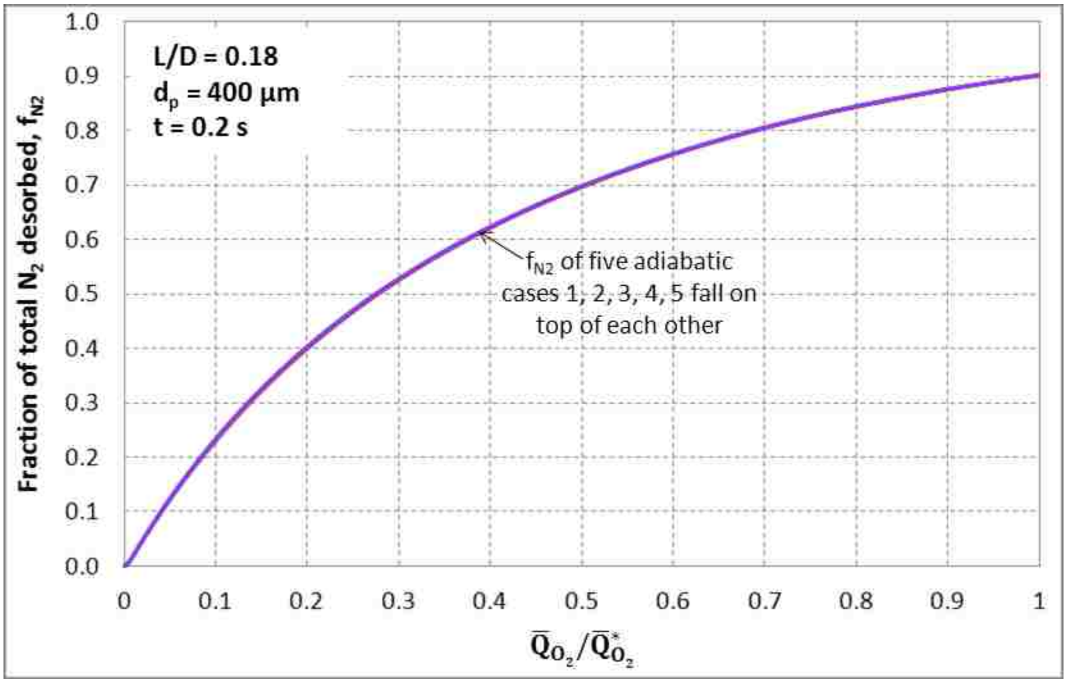


Figure 7.27: Fraction of total N_2 desorbed vs purge O_2 quantity for all cases of Figure 7.25.

It should, however, be emphasized here that the non-isothermal, non-isobaric, non-equilibrium desorption characteristics will generally be system specific, depending on adsorption equilibria and heats, mass and heat transfer kinetics, as well as the duration of the purge process.

7.7.5 Pancake Adsorber: Small Particle Size and Short Step Time

It is generally known that pancake adsorber is used for reducing gas mass flux hence pressure drop across column when small adsorbent particle size is used in conjunction with very short step times of RPSA process. In this section, the adequacy of using ultra-rapid purge step and small particle size is evaluated in pancake adsorber.

Figures 7.28 – 7.30 compare the simulation results of using particle sizes d_p = 100 μm and 400 μm, and purge times $t_{purge} \leq 0.2$ s and ≤ 2 s, in pancake adsorber. Figure 7.28 shows that larger particle size creates temperature difference between gas and solid phases due to gas-solid heat transfer resistances. The temperature difference is magnified in ultra-rapid purge step because there is not enough contact time for gas-solid heat transfer. Smaller particle size yields to bigger temperature plateau and near thermal local equilibrium condition. Smaller temperature plateau is reached for shorter purge time. These temperature profiles and the corresponding pressures and gas phase mole fractions affect the N_2 loadings in the column as shown in Figure 7.29.

Figure 7.29 presents the total N_2 loadings and N_2 gas phase mole fraction profiles in pancake adsorber. Smaller d_p and longer t_{purge} allow better cleaning at the purge entrance region. F values for 100 μm (0.167 s), 100 μm (1.09 s), and 400 μm

(0.163 s), 400 µm (1.11 s), are 23.33%, 26.5%, 3.33%, and 13.33%, respectively. The times in parentheses are the purge durations required to reach f_{N_2} = 0.85.

Furthermore, Figure 7.30 shows that smaller d_p (= 100 µm) and longer t_{purge} (= 2 s) is the preferable combination for saving purge gas amount and reaching better desorption efficiency as demonstrated in Figure 7.29. The combination of smaller d_p (= 100 µm) and ultra-rapid purge time t_{purge} (= 0.2 s), as is generally employed in pancake adsorber (Rama Rao et al. 2010), demands the highest amount of purge gas for the same extent of desorption, which eventually leads to decreased O_2 recovery and increased bed size factor.

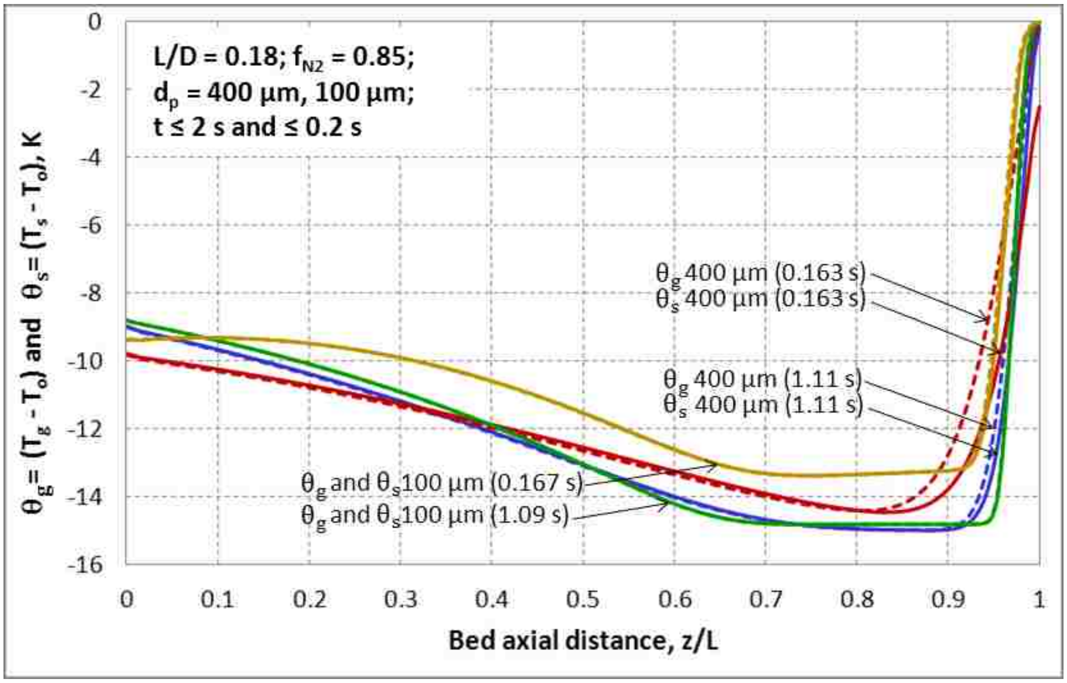


Figure 7.28: Solid (solid lines) and Gas (dashed lines) temperature profiles in column at the end of purge extent f_{N_2} = 0.85. L/D = 0.18, d_p = 400 µm and 100 µm, $Q_{O_2,feed}$ = 13.53 and 1.353 mmoles/cm^2/s, respectively, for t_{purge} ≤ 0.2 s and ≤ 2 s, all using case F (Table 7.6).

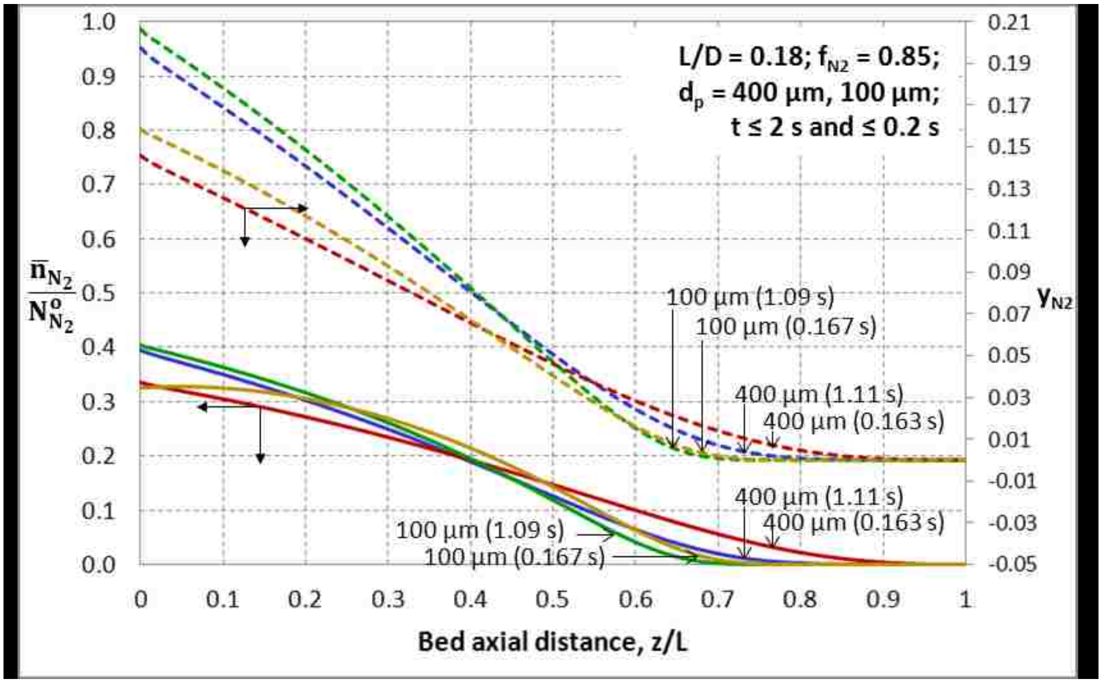


Figure 7.29: Total N_2 loadings (solid lines) and N_2 gas phase mole fraction (dashed lines) profiles in column at the end of purge extent f_{N_2} = 0.85. L/D = 0.18, d_p = 400 µm and 100 µm, $Q_{O_2,feed}$ = 13.53 and 1.353 mmoles/cm^2/s, respectively, for $t_{purge} \leq$ 0.2 s and $\leq$ 2 s, all using case F (Table 7.6).

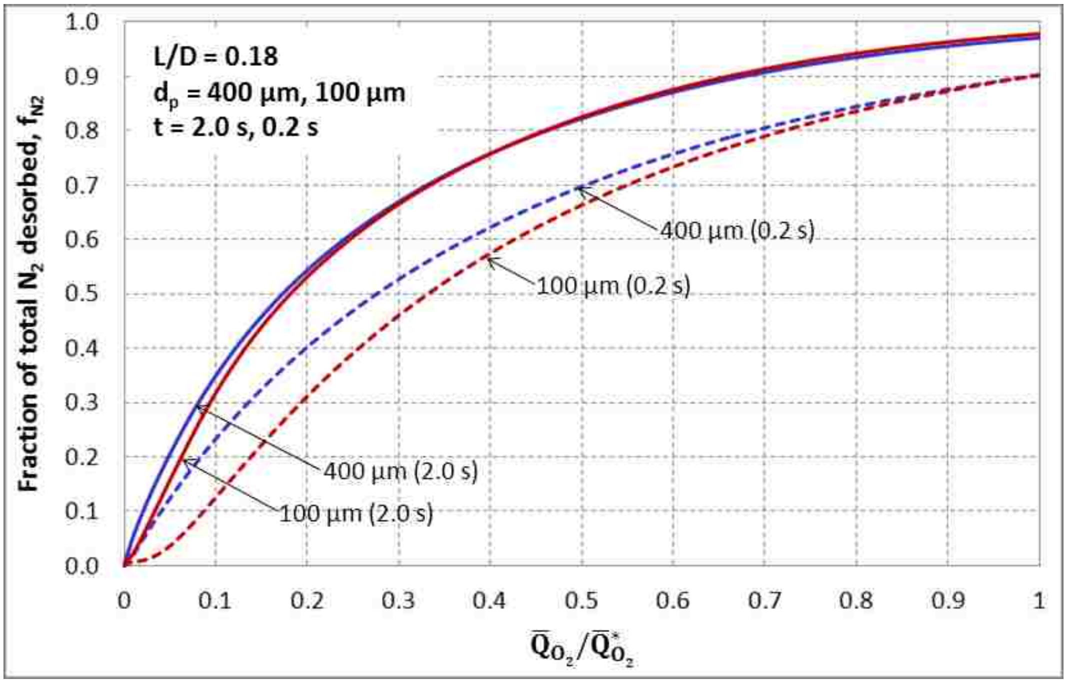


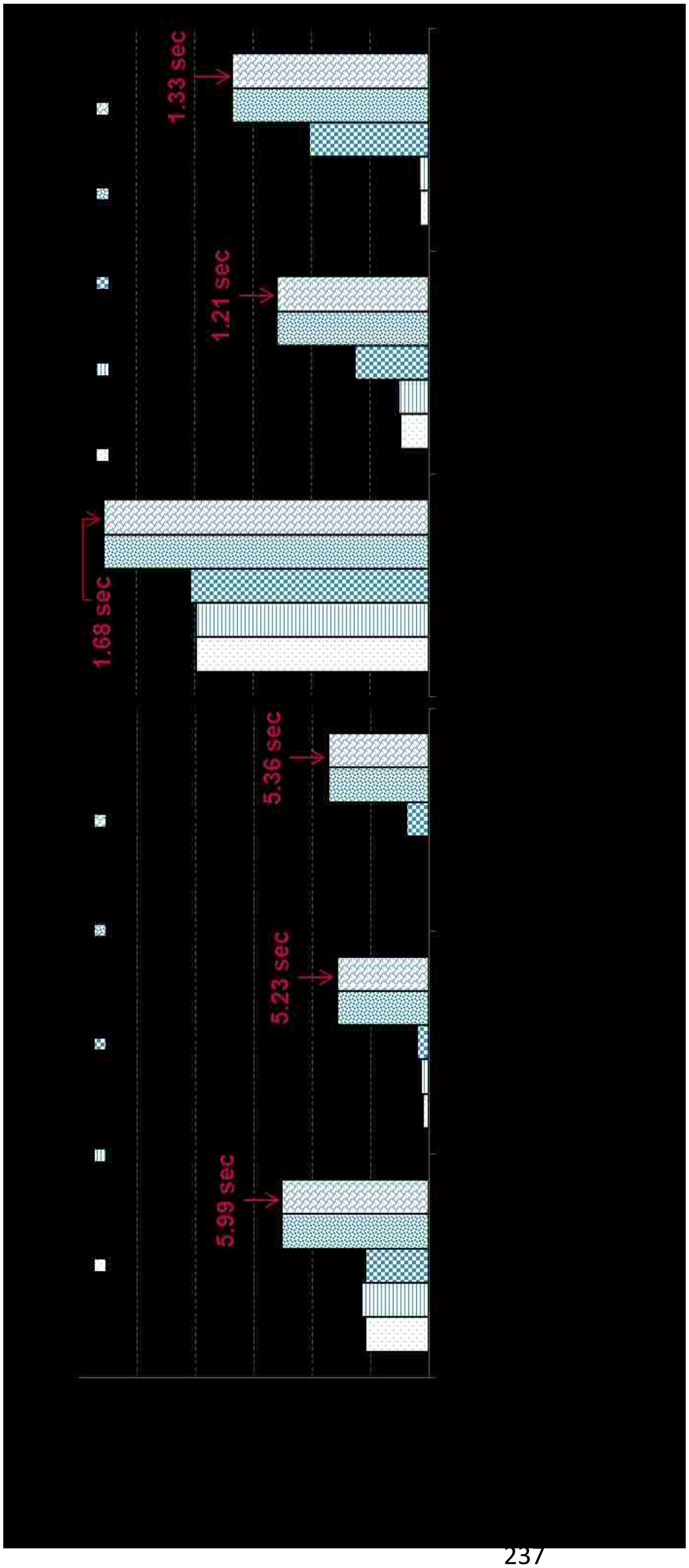
Figure 7.30: Fraction of total N_2 desorbed vs purge O_2 quantity. L/D = 0.18, d_p = 400 µm and 100 µm, $Q_{O_2,feed}$ = 13.53 and 1.353 mmoles/cm^2/s, respectively, for t_{purge} = 0.2 s and 2 s, all using case F (Table 7.6).

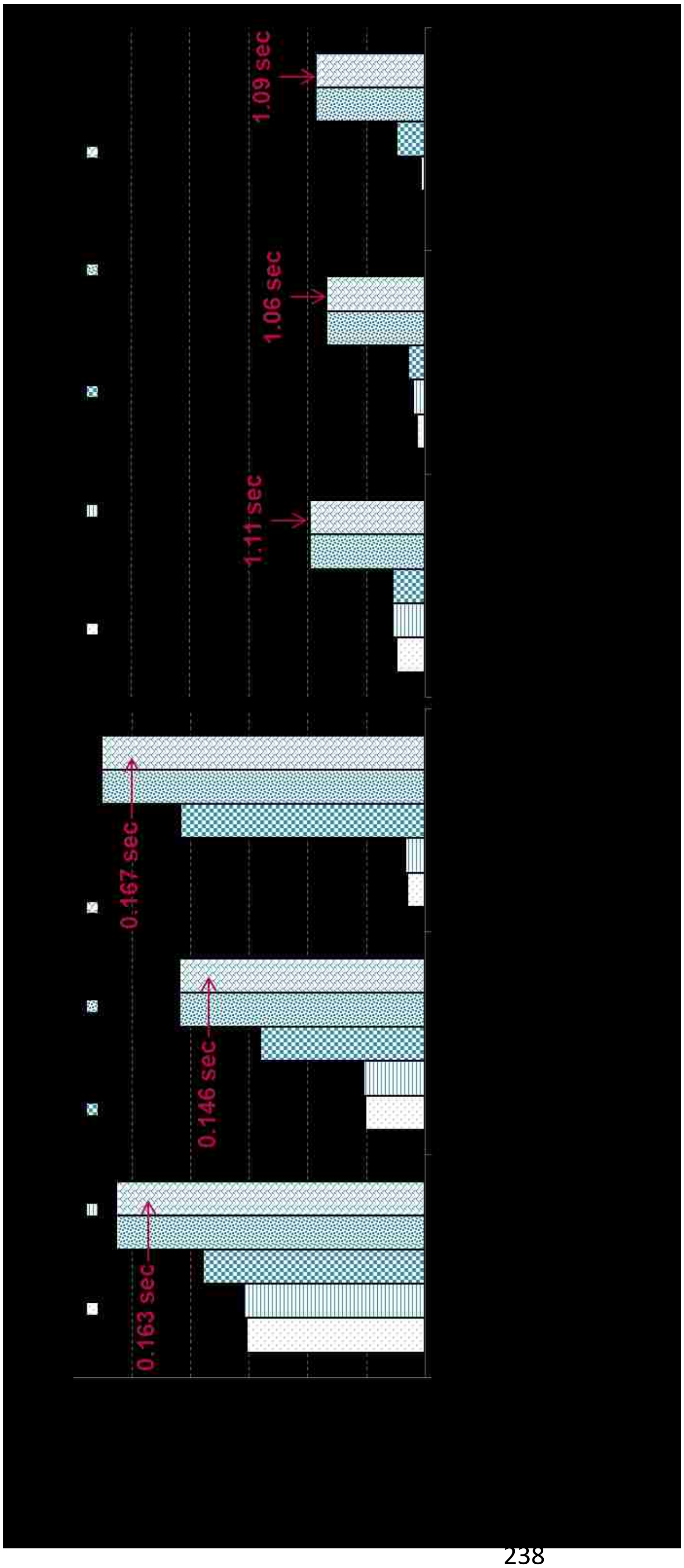
7.7.6 Overall Evaluation of the Efficiency for Desorption-by-Purge Step

In Section 7.6, f_{N_2}, F, and $Q_{O_2}$are identified as the key variables that describe the efficiency of the desorption process. In this section, these variables are evaluated in conventional (L/D = 1.39) and pancake (L/D = 0.18) adsorbers at different particle sizes and purge times in order the frame the optimum desorption-by-purge conditions for these adsorbers.

Figure 7.31 presents the histograms describing progressive increases in O_2 purge gas quantities above that for the idealized desorption case A (represented by base line = 1) due to inclusion of various non-idealities (B – F) listed in Table 7.6, for achieving extent of N_2 desorbed, f_{N_2} = 0.85, in conventional adsorber (L/D = 1.39) using constant O_2 mass flux at the feed end of 1.0706 mmol/cm^2/sec and 5.3534 mmol/cm^2/sec, respectively, for t_{purge} < 10 sec and t_{purge} < 2 sec. It shows that for both long and rapid purge, 400 µm is the optimum particle size that requires minimum amount of O_2 purge gas for the same extent of purge duty. Note that within the same adsorber geometry and using the same particle size of 400 µm, rapid purge step (t_{purge} < 2 sec) requires more purge gas (51.8% above ideal purge) compared to that (31.4% above ideal purge) of longer purge step (t_{purge} < 10 sec) because each resistance is magfinied during rapid purge. Nonetheless, t_{purge} < 2 sec in L/D = 1.39 adsorber is still feasible because rapid cycling makes up more cycles for making extra product gas and compensating the purge amount lost.

Figure 7.32 presents the similar histograms in pancake adsorber (L/D = 0.18) using constant O_2 mass flux at the feed end of 11.4953 mmol/cm^2/sec and 1.1495 mmol/cm^2/sec, respectively, for t_{purge} < 0.2 sec and t_{purge} < 2 sec. It shows that for both ultra-rapid and rapid purge, 200 µm is the optimum particle size that requires minimum amount of O_2 purge gas for the same extent of purge duty. Ultra-rapid purge (t_{purge} < 0.2 sec) and rapid purge (t_{purge} < 2 sec) require 83.7% and 33.4% more purge gas than an ideal purge for this adsorber packed with 200 µm. In view of the very high product loss and increased power consumption for high-frequency-cycling, ultra-rapid purge in pancake adsorber is not economically appealing.


1.33 sec
1.21 sec
1.68 sec
5.36 sec
5.23 sec
5.99 sec


0.163 sec
0.146 sec
0.167 sec
1.11 sec
1.06 sec
1.09 sec

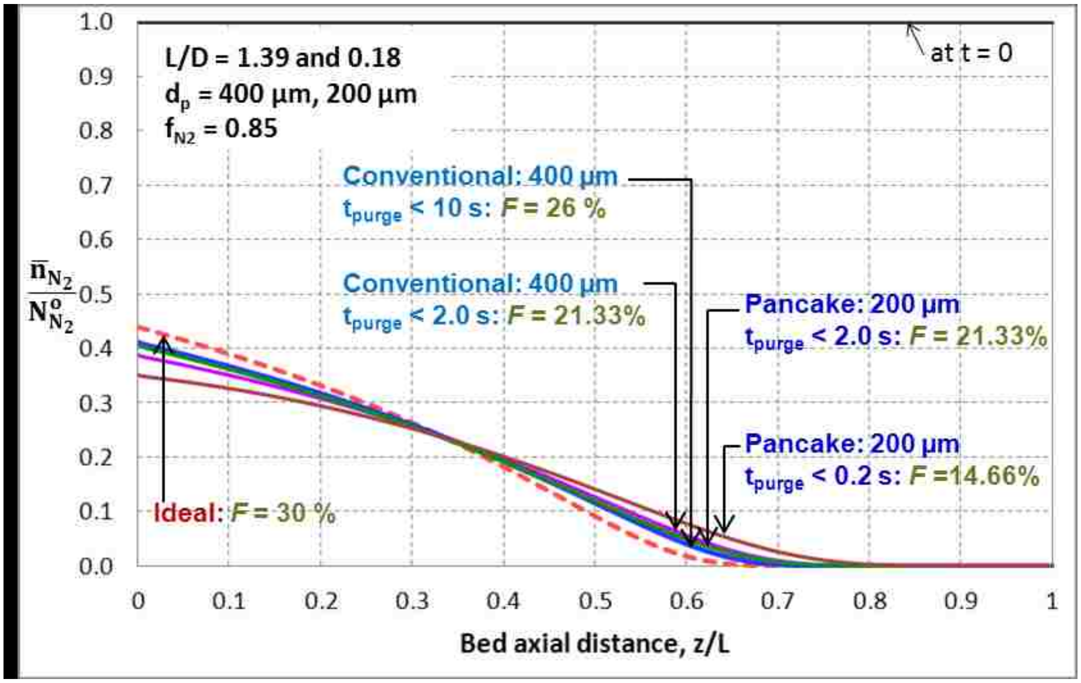


Figure 7.33: Fraction of total N_2 loadings vs bed axial distance for f_{N_2}= 0.85. L/D = 1.39 and 0.18, d_p= 400 μm and 200 μm, $Q_{O_2,feed}$ and purge durations corresponding to those in Figures 7.31 and 7.32. Dashed line: ideal case A; Solid lines: case F (Table 7.6).

Figure 7.33 presents the total N_2 loading profiles and fraction of column volume free of N_2 at the purge inlet end, F, in conventional and pancake adsorbers and their respective optimum particle size (400 μm and 200 μm) at long (t_{purge} < 10 sec), rapid (t_{purge} < 2 sec), and ultra-rapid (t_{purge} < 0.2 sec) purge times. It is seen that rapid purge time can produce satisfactory high F without requiring excessively high O_2 purge gas amount (as depicted by Figures 7.31 and 7.32). Ultra-rapid purge in pancake adsorber, on the other hand, produces low F and demands excessively high O_2 purge gas amount. Pancake adsorber also does not fit with the use of very small particle size.

From these analysis, it is shown that selection of purge time, particle size, and adsorber geometry must consider f_{N_2}, F, and Q_{O_2}simultaneously.

7.8 Conclusions of Numerical Studies on Desorption-by-Purge in RPSA MOC

Rapid pressure swing adsorption processes are frequently employed for production of ~ 90% O_2 enriched gas from ambient air in medical oxygen concentrators for individual use. Many of these processes use a product gas back purge step as a part of the N_2 desorption scheme for cyclic regeneration of the N_2 selective adsorbent. A simple sensitivity analysis of an idealized RPSA process shows that any error in the estimation of the purge gas quantity can substantially magnify the errors in estimation of overall performance of the process such as bed size factor (BSF) and O_2 recovery (R).

A detailed mathematical simulation of a rapid, adiabatic, non-isothermal, non-isobaric, and non-equilibrium desorption process for N_2 removal from a LiX zeolite column, which is initially equilibrated with air, by pure O_2 purge is presented. Coupled partial differential equations describing mass balances, gas and solid phase heat balances, and pressure drop in the column were simultaneously solved numerically in conjunction with published models for describing binary adsorption equilibria for the system and conventional linear driving force model for adsorption kinetics. Ergun equation described the transient column pressure drop. Effects of adsorbate mass transfer coefficient, column pressure drop, column non-isothermality, gas-solid heat transfer coefficient, mass and heat axial dispersions in gas phase, desorption time, adsorbent particle size, and column geometry (length to diameter, L/D, ratio) on desorption-by-purge efficiency were systematically evaluated.

The minimum amount of O_2 purge gas for a given desorption duty (fraction of total N_2 that is initially present in the column removed by the purge process) is needed by an idealized case where the process is isothermal, isobaric, and controlled by local thermodynamic equilibrium. A realistic desorption-by-purge process requires a significantly larger purge gas quantity for the same duty. The analysis shows that adsorption kinetics (especially for a rapid desorption process), column pressure drop, and non-isothermal column operation are major contributors to desorption inefficiency. The influence of gas phase mass and thermal axial dispersions is minimal. Lowering adsorbent particle size increases adsorption kinetics but at the cost of increased pressure drop. Consequently there is an optimum particle size which can minimize the purge gas requirement for a specific desorption duty. A pancake shaped column (L/D < 0.2) can be effective in lowering the effect of column pressure drop in conjunction with smaller adsorbent particles, but chances of gas mal-distribution increase. The negative impact of non-isothermal operation cannot generally be avoided, and needs to be accounted for in a realistic process design.

The gas and solid temperature profiles in the column at the end of a rapid desorption process can be complex and non-intuitive. They dip to a minimum value near the purge gas inlet end of the column to form a plateau region, and then slowly rise towards the exit end. The entire column cools down to supply the heat of desorption if the purge gas is not heated above the initial column temperature. Existence of a finite gas-solid heat transfer coefficient creates different gas and solid temperature profiles inside the column which are substantially different from those of a column operating under local thermal equilibrium condition only in a section near the purge gas inlet end. These profiles are very similar in the balance of the column length irrespective of the value of the heat transfer coefficient. Consequently, the influence of finite heat transfer coefficient on an integrated overall desorption efficiency is small for the present system.

It is also shown that ultra-rapid purge step ($t_{purge} \leq 0.2$ s) in conjunction with the use of very small particle size (≤ 100 µm) for pancake adsorber is detrimental because the amount of purge gas required is high and the adsorbent bed is not properly cleaned.

A very important conclusion of this study is that a simplified isothermal, or isobaric, or local equilibrium process design model of rapid desorption can lead to an unrealistically optimistic MOC process performance.

7.9 References

Alpay, E., Kenney, C.N., Scott, D.M.: Adsorbent Particle Size Effects in the Separation of Air by Rapid Pressure Swing Adsorption. *Chem. Eng. Sci.,* **49**, 3059 (1994).

Basmadjian, D., Ha, K.D., Pan, C.Y.: Nonisothermal Desorption by Gas Purge of Single Solutes in Fixed-Bed Adsorbers. I. Equilibrium Theory. *Ind. Eng. Chem. Process Des. Dev.,* **14**, 328 (1975a).

Basmadjian, D., Ha, K.D., Proulx, D.P.: Nonisothermal Desorption by Gas Purge of Single Solutes from Fixed-Bed Adsorbers. II. Experimental Verification of Equilibrium Theory. *Ind. Eng. Chem. Process Des. Dev.,* **14**, 340 (1975b).

Chai, S.W., Kothare, M.V., Sircar, S.: Rapid Pressure Swing Adsorption for Reduction of Bed Size Factor of a Medical Oxygen Concentrator. *Ind. Eng. Chem. Res.* (2011).

Ergun, S.: Fluid Flow Through Packed Columns. *Chem. Eng. Prog.,* **48**, 89 (1952).

Jacob, P., Tondeur, D.: Non-Isothermal Gas Adsorption in Fixed Beds II. Non-Linear Equilibrium Theory and 'Guillotine' Effect. *Chem. Eng. J.,* **26**, 41 (1983).

Kopaygorodsky, E.M., Guliants, V.V., Krantz, W.B.: Predictive Dynamic Model of Single-Stage Ultra-Rapid Pressure Swing Adsorption. *AIChE J.,* **50**, 953 (2004).

Kunii, D., Suzuki, M.: Particle-to-Fluid Heat and Mass Transfer in Packed Beds of Fine Particles. *Int. J. Heat Mass Transfer,* **10**, 845 (1967).

Moulijn, J.A., Van Swaaij, W.P.M.: The Correlation of Axial Dispersion Data for Beds of Small Particles. *Chem. Eng. Sci.,* **31**, 845 (1976).

Porter, K.E., Ali, Q.H., Hassan, A.O., Aryan, A.F.: Gas Distribution in Shallow Packed Beds. *Ind. Eng. Chem. Res.,* **32**, 2408 (1993).

Rama Rao, V., Farooq, S., Krantz, W.B.: Design of a Two-Step Pulsed Pressure Swing Adsorption Based Oxygen Concentrator. *AIChE J,* **56**, 354 (2010).

Rege, S.U., Yang, R.T.: Limits for Air Separation by Adsorption with LiX Zeolite. *Ind. Eng. Chem. Res.,* **36**, 5358 (1997).

Rhee, H.K., Amundson, N.R.: An Analysis of an Adiabatic Adsorption Column: Part I Theoretical Development. *Chem. Eng. J.,* **1**, 241 (1970).

Rhee, H.K, Heerdt, E.D., Amundson, N.R.: An Analysis of an Adiabatic Adsorption Column: Part III Adiabatic Adsorption of Two Solutes. *Chem. Eng. J.,* **3**, 22 (1972).

Santos, J.C., Portugal, A.F., Magalhaes, F.D., Mendes A.: Simulation and Optimization of Small Oxygen Pressure Swing Adsorption Units. *Ind. Eng. Chem. Res.*, **43**, 8328 (2004).

Santos, J.C., Portugal, A.F., Magalhaes, F.D., Mendes A.: Optimization of Medical PSA Units for Oxygen Production. *Ind. Eng. Chem. Res.*, **45**, 1085 (2006).

Sircar, S., Kumar, R.: Equilibrium Theory for Adiabatic Desorption of Bulk Binary Gas Mixtures by Purge. *Ind. Eng. Chem. Process Des. Dev.,* **24**, 358 (1985).

Sircar, S., Golden, T.C.: Isothermal and Isobaric Desorption of Carbon Dioxide by Purge. *Ind. Eng. Chem. Res.,* **34**, 2881 (1995).

Todd, R.S., Webley, P.A.: Mass-Transfer Models for Rapid Pressure Swing Adsorption Simulation. *AIChE J.,* **52**, 3126 (2006).

Wakao, N., Kaguei, S., Funazkri, T.: Effect of Fluid Dispersion Coefficients on Particle-to-Fluid Heat Transfer Coefficients in Packed Beds. *Chem. Eng. Sci.,* **34,** 325 (1979).

Wicke, E.: Empirische and Theoretische Untersuchungen der Sorptionsgeschwindigkeit von Gasen an por sen Stoffen I. *Kolloid, Z.,* **86**, 167 (1939a).

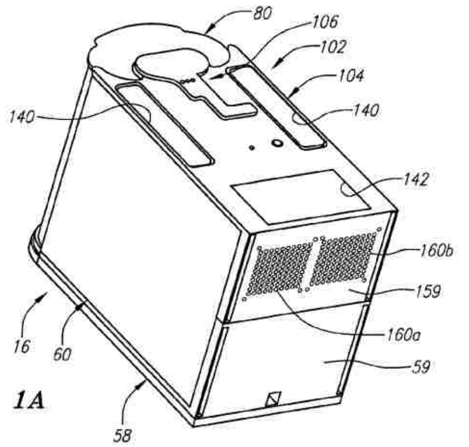
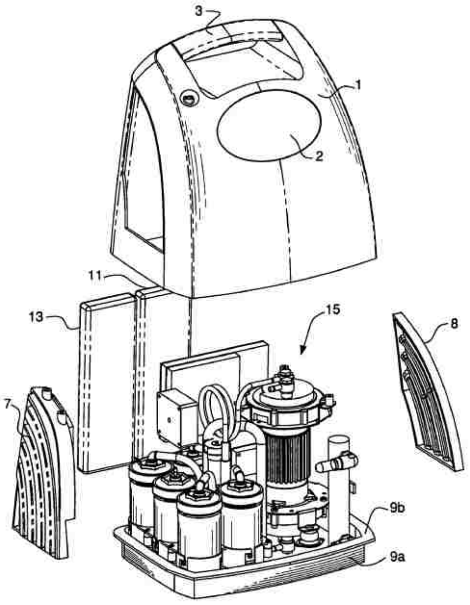
Wicke, E.: Empirische and Theoretische Untersuchungen der Sorptionsgeschwindigkeit von Gasen an por sen Stoffen II. *Kolloid, Z.,* **86**, 295 (1939b).

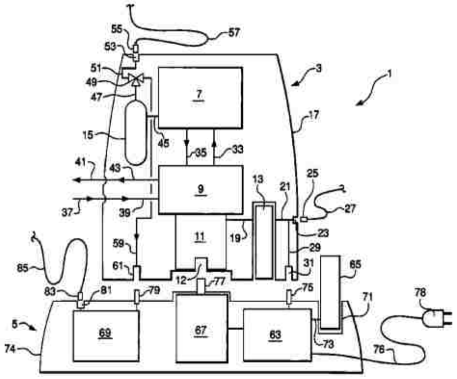
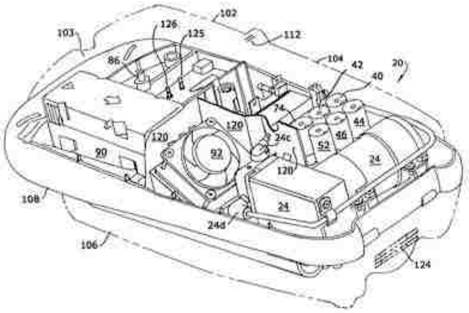
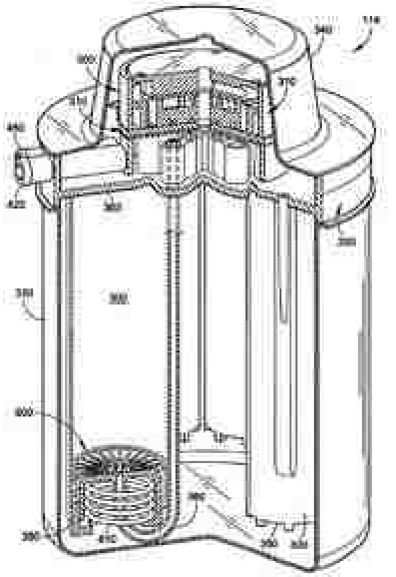
Zhong, G.M, Rankin, P.J., Ackley, M.W.: High Frequency PSA Process for Gas Separation. U.S. Patent 7,828,878 (2010).

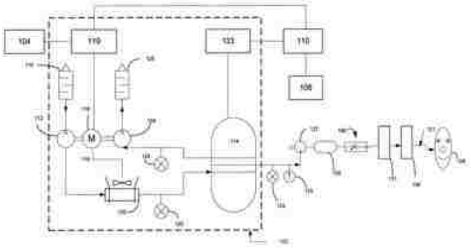
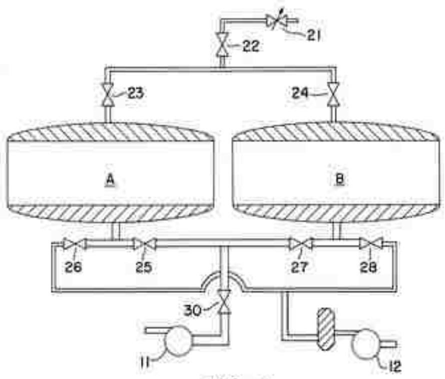
Appendix 1

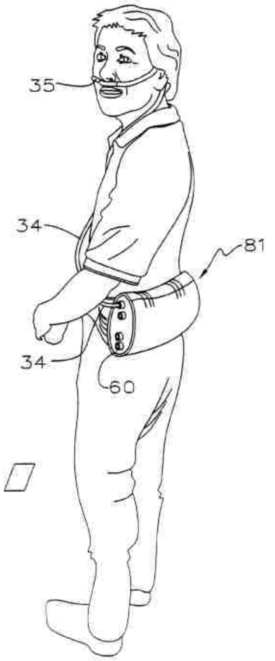
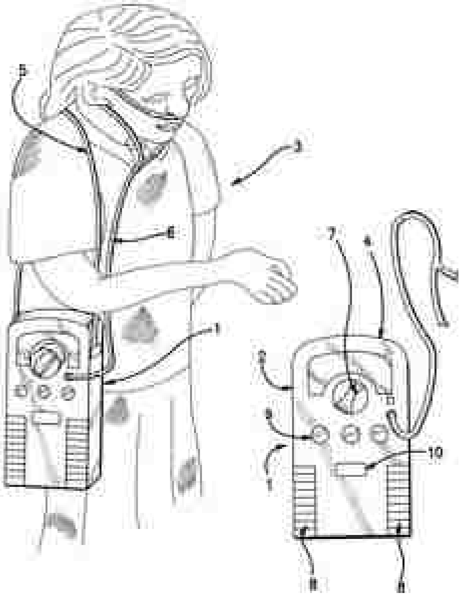
U. S. Patents of Portable Medical Oxygen Concentrators

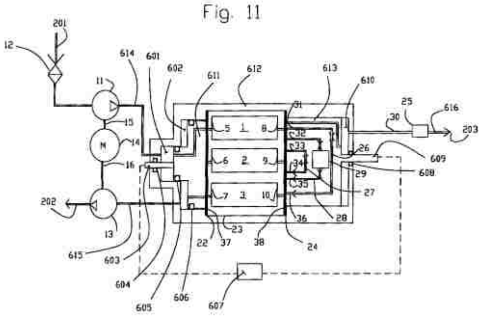
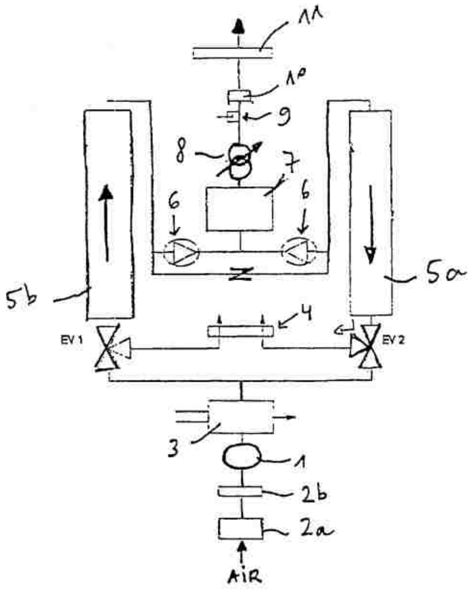
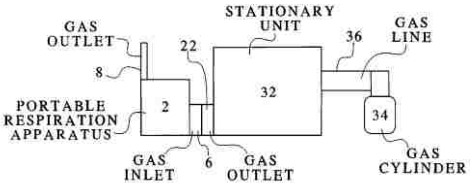
Table A.1: U.S. Patents of portable medical oxygen concentrators.

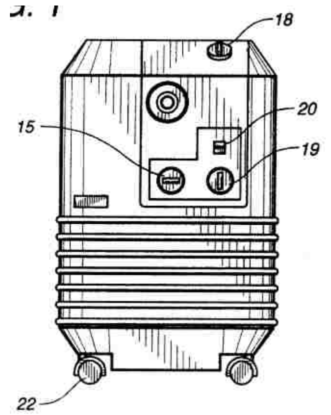
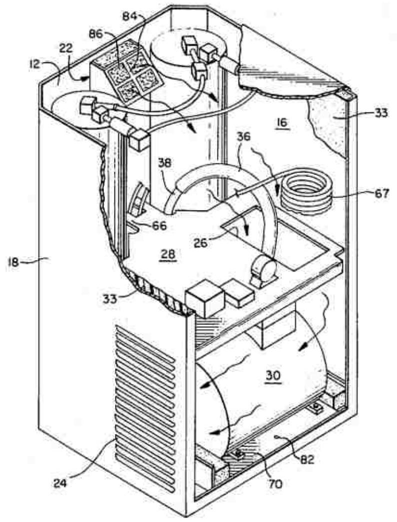
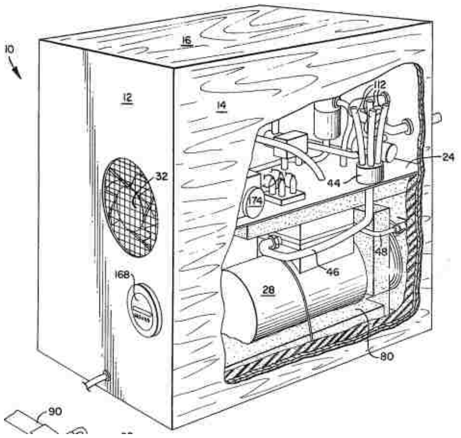
Patent No.	Inventors	Publication Date Filing Date	Assignee	Representative image
US 7,794,522 B2 Portable oxygen concentrator	Bliss et al. Bliss, Peter L. (Prior Lake, MN, US) Atlas, Charles R. (Coto de Caza, CA, US) Halperin, Scott Carson (Orange, CA, US)	09/14/2010 02/04/2008	Respironics, Inc. (Murrysville, PA, US)	 80 106 102 104 140 140 142 160b 159 160a 59 16 60 1A 58
US 7,510,601 B2 Portable medical oxygen concentrator	Whitley et al. Whitley, Roger Dean (Allentown, PA, US) Wagner, Glenn Paul (Fogelsville, PA, US) Labuda, Matthew James (Fogelsville, PA, US) Schiff, David R. (Highland Park, NJ, US) Byar, Peter D. (Willingboro, NJ, US) Weiman, Andrew M. (Langhorne, PA, US) Galewyrick, Seth (Philadelphia, PA, US)	03/31/2009 12/20/2005	Air Products and Chemicals, Inc. (Allentown, PA, US)	

Patent No.	Inventors	Publication Date Filing Date	Assignee	Representative image
US 7,273,051 B2 Dual mode medical oxygen concentrator	Whitley et al. Whitley, Roger Dean (Allentown, PA, US) Wagner, Glenn Paul (Fogelsville, PA, US) Labuda, Matthew James (Fogelsville, PA, US)	09/25/2007 01/22/2004	Air Products and Chemicals, Inc. (Allentown, PA, US)	 1 3 5 7 9 11 12 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 74 75 76 77 78 79 81 83 85
US 6,764,534 B2 Portable oxygen concentrator	McCombs et al. Mccombs, Norman R. (Tonawanda, NY) Casey, Robert E. (Buffalo, NY) Chimiak, Michael A. (Williamsville, NY) Klimaszewski, Andrezj (Jacksonville, FL)	07/20/2004 01/30/2003	AirSep Corporation (Buffalo, NY)	 102 103 126 125 112 86 104 42 20 40 74 44 46 120 24c 52 90 92 24 108 24d 106 124
US 6,691,702 B2 Portable oxygen concentration system and method of using the same	Appel et al. Appel, William Scot (San Diego, CA) Winter, David Phillip (Encinitas, CA) Sward, Brian Kenneth (San Diego, CA) Sugano, Masato (Tokyo, JP) Salter, Edmund (Oceanside, CA) Bixby, James A. (La Jolla, CA)	02/17/2004 04/29/2002	SeQual Technologies, Inc. (San Diego, CA) Teijin Limited (Osaka, JP)	

Patent No.	Inventors	Publication Date Filing Date	Assignee	Representative image
US 6,651,658 B1 Portable oxygen concentration system and method of using the same	Hill et al. Hill, Theodore B. (San Diego, CA) Radtke, Edward A. (El Cajon, CA) Schneider, Robert A. (Del Mar, CA) Bixby, James A. (La Jolla, CA)	11/25/2003 08/03/2000	SeQual Technologies, Inc. (San Diego, CA) Teijin Limited (Osaka, JP)	
US 6,551,384 B1 Medical oxygen concentrator	Ackley et al. Ackley, Mark William (East Aurora, NY) Zhong, Guoming (Getzville, NY)	04/22/2003 07/05/2001	Praxair Technology, Inc. (Danbury, CT)	 21 22 23 24 A B 26 25 27 28 30 11 12 FIG. 1

Patent No.	Inventors	Publication Date Filing Date	Assignee	Representative image
US 6,547,851 B2 Miniaturized wearable oxygen concentrator	Warren Warren, John Lee (Salmon Arm, CA)	04/15/2003 08/02/2001	Wearair Oxygen Inc. (Kelowna, CA)	
US 6,520,176 B1 Portable oxygen concentrator	Dubois et al. Dubois, Anne (Le Chesnay, FR) Bodelin, Pierre (Vanves, FR) Vigor, Xavier (Chicago, IL)	02/18/2003 06/30/2000	L'Air Liquide, Societe Anonyme a Directoire et Conseil de Surveillance pour l'Etude et l'Exploitation des Procedes Georges Claude (Paris, FR)	

Patent No.	Inventors	Publication Date Filing Date	Assignee	Representative image
US 6,514,319 B2 Life support oxygen concentrator	Keefer et al. Keefer, Bowie G. (Vancouver, CA) Mclean, Christopher (Vancouver, CA)	02/04/2003 12/08/2000	QuestAir Technologies Inc. (Burnaby, CA)	Fig. 11  201 12 614 601 602 611 612 613 610 25 616 30 203 11 15 M 14 16 5 1 8 6 2 9 7 3 10 31 32 33 34 35 26 609 608 29 27 28 36 202 13 615 603 604 605 606 607 22 37 23 38 24
US 6,314,957 B1 Portable home oxygen therapy medical equipment	Boissin et al. Boissin, Jean-claude (Saint-Ismier, FR) Hennebel, Vincent (Sevres, FR)	11/13/2001 04/16/1999	Air Liquide Sante (International) (Paris Cedex, FR)	 11 10 9 8 7 6 6 5b 5a 4 EV 1 EV 2 3 1 2b 2a AIR
US 6,095,138 Portable respiration apparatus, and system employing same	Hognelid et al. Hognelid, Kurt (Bromma, SE) Psaros, Georgios (Tullinge, SE)	08/01/2000 05/13/1998	Siemens Elema AB (Solna, SE)	 GAS OUTLET STATIONARY UNIT GAS LINE 22 36 8 32 PORTABLE RESPIRATION APPARATUS 2 34 GAS INLET 6 GAS OUTLET GAS CYLINDER

Patent No.	**Inventors**	**Publication Date** **Filing Date**	**Assignee**	**Representative image**
US 5,893,944 Portable PSA oxygen generator	Dong Dong, Jung Hyi (262-13, Namgaza-dong, Seodaemun-ku, Seoul, KR)	04/13/1999 11/07/1997		
US 4,378,982 Compact oxygen concentrator	McCombs Mccombs, Norman R. (Tonawanda, NY)	04/05/1983 08/28/1981	Greene & Kellogg, Inc. (Tonawanda, NY)	 84 86 22 12 33 16 36 38 67 66 26 28 18 33 30 24 70 82
US 4,302,224 Compact oxygen concentrator	McCombs et al. McCombs, Norman R. (Tonawanda, NY) Schlaechter, John (Kenmore, NY)	11/24/1981 10/12/1979	Greene & Kellogg, Inc. (Tonawanda, NY)	 16 10 12 14 112 24 32 44 174 48 46 168 28 80 90

Appendix 2

Expenses of Experimental Apparatus and Supplies

Table A.2: Expenses of experimental apparatus and supplies.

No.	Items Required	Estimated Cost (USD)
1.	Mass flow controllers and meters and scheduled calibration	12,000
2.	Servomex oxygen analyzer	6,000
3.	Solenoid valves, control valves, needle valves	1,000
4.	Programmable logic controller and data acquisition system	3,500
5.	Compressed gas cylinders	2,000
6.	Analytical balance	1,200
7.	Benchtop precision thermistor thermometers with optional analog output	1000
8.	Tubings and connectors	800
9.	Desktop for data acquisition and control system	1500
10.	Back pressure regulator	600
11.	Heating tapes and temperature controller	900
12.	Pressure gauges	500
13.	Electronic and mechanical appliances / tools such as power adapters, connectors, buttons, wires, cables, wrenches, silicon adhesives etc.	500
14.	Miscellaneous	2,500
Total estimated cost:		**34,000**

Appendix 3

Models A, B, C

- **Model A**, an ideal adsorption (`adsorptionORpurge = 111`) or desorption (`adsorptionORpurge = 222`) step that **excludes all resistances**, is modeled in this routine by setting (`k1 = 50000, k2 = 50000 or k1 = ∞, k2 = ∞`), (`MassAxial = 881`).
- **Model B**, an adsorption (`adsorptionORpurge = 111`) or desorption (`adsorptionORpurge = 222`) step that includes **gas-solid adsorption kinetic**, is modeled in this routine by setting (`%k1 = 50000, %k2 = 50000`), (`MassAxial = 881`).
- **Model C**, an adsorption (`adsorptionORpurge = 111`) or desorption (`adsorptionORpurge = 222`) step that includes **gas-solid adsorption kinetic, gas phase mass axial dispersion**, is modeled in this routine by setting (`%k1 = 50000, %k2 = 50000`), (`MassAxial = 882`).
- One setting that especially needs your attention is (`nout, n`). This combination must be set properly to achieve numerical stability and convergence.
- One out of three spatial differential methods can be selected (`dYbardZbarDifferentialMethod = 333, or 444, or 777`).

Main Routine

```
%...........................................................................
% File: psa_main_sequence_model_ABC
%...........................................................................
% Chai Siew Wah, Chemical Engineering, Lehigh University, May 2011
%...........................................................................
% MATHEMATICAL MODEL OF PRESSURE SWING ADSORPTION SYSTEM FOR ONE STEP :
% ADSORPTION OR PURGE
%...........................................................................
%
%                  :.......................................:
%   ADSORPTION --> :                                       : --> PRODUCT
%   ADSORPTION --> :                                       : --> PRODUCT
%   ADSORPTION --> :                                       : --> PRODUCT
%                  :.......................................:
%                 i=1                                     i=n
%
%                  :.......................................:
%     EXHAUST <--  :                                       : <-- PURGE
%     EXHAUST <--  :                                       : <-- PURGE
%     EXHAUST <--  :                                       : <-- PURGE
%                  :.......................................:
%                 i=1                                     i=n
%...........................................................................
```

```
% Assumptions
%..........................................................................
% (1) Ideal gas law
% (2) Empirical Langmuirian type binary isotherms - from Rege and Yang 1997
% (3) Isobaric
% (4) Isothermal
% (5) Model A - free of all reistances
% (6) Model B - adding Linear driving force model for mass transfer kinetic
% (7) Model C - adding Mass axial dispersion in gas phase
% (7) Absence of solid phase thermal axial conduction
% (8) Absence of radial distribution of mass and heat
% (9) Absence of gas mal-distribution or particle agglomeration
%..........................................................................
% 3 ordinary differential equations : y1, n1, n2
% 1 direct calculation for          : rhogbar (ideal gas law)
% 1 algebral equation for           : Q (from overall mass balance)
% 1 calculated for                  : us (= Q/rhog) - but reduce its use
% Solve for y1, n1, n2, P, rhog, Q, us
%..........................................................................
% Clear previous files
clc
clear all
format short
%..........................................................................
% Parameters shared with the ODE routine
%..........................................................................
global  Mg1 Mg2 adsorptionORpurge dYbardZbarDifferentialMethod rhog0 T0....
    ncall n dzbar dtbar miu epsilon epsilonbar k1 k2 q1 q2 Cs Cg rhob R....
    dp L tfeed y1feed y2feed Qin Qfeed P0 Pout Tfeed Pmax Pmin Tmax Tmin...
    Tref Q0 kg MassAxial HeatAxial HeateqmORnoneqm Dem4 Dem1
%..........................................................................
% Select process step
%..........................................................................
adsorptionORpurge = 222;       %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% adsorptionORpurge = 111;     % 111 for adsorption
% adsorptionORpurge = 222;     % 222 for purge
%..........................................................................
% Assign positive or negative sign to feed gas mass flux (Q) and average
% velocity (uz = us/epsilon) at feed and initial condition (which are
% within our control). Do not assign +ve / -ve to Qbar(i) and usbar(i),
% which are calculated by the program
%..........................................................................
if adsorptionORpurge == 111
    FE = +1;    % for superficial velocity at feed end i = 1
end
if adsorptionORpurge == 222
    FE = -1;    % for superficial velocity at feed end i = n
end
%..........................................................................
% Select differential method for dYbardZbar
%..........................................................................
dYbardZbarDifferentialMethod = 777;   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% dYbardZbarDifferentialMethod = 333  % 1st order (2 points) upwind differe
% dYbardZbarDifferentialMethod = 444; % 2nd order (3 points) upwind differe
% dYbardZbarDifferentialMethod = 777; % 2nd order TVD Superbee Flux Limiter
%..........................................................................
% Select Mass and Heat Axial Dispersions
%..........................................................................
MassAxial = 882;       %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% MassAxial = 881;     % Shut off mass axial dispersion
% MassAxial = 882;     % Turn on mass axial dispersion, need 2nd BC
%..........................................................................
% Select type of plots
%..........................................................................
choose = 4;  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% choose = 1 : 2D subplots of y1, y2, n1, n2, T, P
% choose = 2 : Individual 2D plots of y1, y2, n1, n2, T, P, rhog, Q, us
% choose = 3 : Surface plots of y1, y2, n1, n2, T, P, rhog, Q, us
% choose = 4 : The 2D plots that Dr. Sircar is interested in
%..........................................................................
% Define number of nodes along t and along z
%..........................................................................
nout = 1001;        % nodes along t
n = 151;            % nodes along z
%..........................................................................
% Define dimensionless z and dz
%..........................................................................
zbarl = 0.0;                    % zbar lower limit, zbar = 0/L
zbaru = 1.0;                    % zbar upper limit, zbar = L/L
dzbar = (zbaru-zbarl)/(n-1);    % zbar differential space
zbar  = linspace(zbarl,zbaru,n);
%..........................................................................
% Define dimensionless t and dt
% Independent variable, tbar, for ODE integration
%..........................................................................
tbarl = 0.0;                    % tbar lower limit, tbar = 0/tfeed
tbaru = 1.0;                    % tbar upper limit, tbar = tfeed/tfeed
dtbar = (tbaru-tbarl)/(nout-1); % tbar differential space
tbar  = linspace(tbarl,tbaru,nout); % horizontal tbar
tbarv = tbar';                      % vertical tbar
%..........................................................................
% Gas properties
%..........................................................................
R = 8.314472*10^4;      % gas constant, g.cm2/K/mmol/sec2
Cg = 0.00687;           % gas phase heat capacity, cal/mmol/K, Rege&Yang 97
%kg = 6.19e-5;          % gas phase (N2 at 1 atm 300K) thermal conductivity
                        % , cal/cm/sec/K
%Cg = 0.00721;          % gas phase (Air at 300K) heat capacity, cal/mmol/K
kg = 6.44e-5;           % gas phase (Air at 1 atm 295K) thermal
                        % conductivity, cal/cm/sec/K
miu = 18.2385e-5;       % air dynamic viscosity at 25C and 1 atm, g/cm/sec
Mg1 = 0.032;            % O2 molecular weight, g/mmol
Mg2 = 0.028;            % N2 molecular weight, g/mmol
%..........................................................................
% Packed bed dimensions
%..........................................................................
% Conventional adsorber
L = 8.382;              % bed length, cm (3.3 in = 8.382 cm)
D = 6.0325;             % bed diameter, cm (2 3/8 in)

% Pancake adsorber
%L = 2.12;              % bed length, cm (3.3 in = 8.382 cm)
%D = 12;                % bed diameter, cm (2 3/8 in)

A = pi*(D/2)^2;         % bed cross-sectional area, cm2
rhob = 0.668;           % bulk density, g/cm3, according to 160 g LiX
%rhob = 0.632;          % bulk density, g/cm3, at 1/3 lb (151.33 g) LiX
%..........................................................................
% Adsorbent properties
%..........................................................................
```

```
dp = 0.0200;            % particle diameter, cm   %%%%%%%%%%%%%%%%%%%%%%%%%
epsilon = 0.61;         % total column (helium) void fraction= ep x (1-e)+e
epsilonp = 0.35;        % internal/intraparticle void fraction
epsilonbar = 0.40;      % external/interparticle void fraction, bed voidage
                        % , Rege and Yang 1997
taup = 3;               % particle tortuosity factor
Cs = 0.28;              % adsorbent (solid phase) heat capacity, cal/g/K,
                        % Rege and Yang 1997
ks = 4.78e-4;           % dry sand thermal conductivity, 298K, cal/cm/sec/K
%..........................................................................
% Mass transfer kinetic and heat properties
%..........................................................................
Dek1 = 0.0307;          % O2 effective knudsen diffusivity in Zeochem LiLSX
                        % at 1-4 atm, cm2/sec, Todd and Webley 2006
Dek2 = 0.03277;         % N2 effective knudsen diffusivity in Zeochem LiLSX
                        % at 1-4 atm, cm2/sec, Todd and Webley 2006
Dem1 = 0.0348;          % effective molecular diffusivity in Zeochem LiLSX
                        % at 1 atm, cm2/sec, Todd and Webley 2006
Dem4 = 0.008707;        % effective molecular diffusivity in Zeochem LiLSX
                        % at 4 atm, cm2/sec, Todd and Webley 2006

if adsorptionORpurge == 111
    De1 = (epsilonp/taup)*(1/(1/Dek1 + 1/Dem4));
    De2 = (epsilonp/taup)*(1/(1/Dek2 + 1/Dem4));
end
if adsorptionORpurge == 222
    De1 = (epsilonp/taup)*(1/(1/Dek1 + 1/Dem1));
    De2 = (epsilonp/taup)*(1/(1/Dek2 + 1/Dem1));
end
k1 = 15*De1/(dp/2)^2;   % O2 overall mass transfer coefficient, sec-1
k2 = 15*De2/(dp/2)^2;   % N2 overall mass transfer coefficient, sec-1
q1 = 3.16;              % isosteric heat of adsorption of O2 on LiX, cal/
                        % mmol, Rege and Yang 1997
q2 = 5.60;              % isosteric heat of adsorption of N2 on LiX, cal/
                        % mmol, Rege and Yang 1997
%k1 = 50000;            % equilibrium O2 overall mass transfer rate, sec-1
%k2 = 50000;            % equilibrium N2 overall mass transfer rate, sec-1
%..........................................................................
% Feed conditions at i = 1 (adsorption)
%                 at i = n (purge)
%..........................................................................
if adsorptionORpurge == 111
    y1feed = 0.21;      % feed gas composition, 21% O2 + 79% N2
    y2feed = 0.79;      % feed gas composition, 21% O2 + 79% N2
    Tfeed = 298;        % feed temperature, K
    tfeed = 2;          % feed step time, sec
    qfeed = 300/tfeed;  % feed mass flow rate, mmol/sec
    Pout = 4.0*1013250; % exit pressure at i=n during adsorption, g/cm/sec2
end
if adsorptionORpurge == 222
    y1feed = 1.00;      % purge gas composition, 100% O2 + 0% N2
    y2feed = 0.00;      % purge gas composition, 100% O2 + 0% N2
    Tfeed = 298;        % purge temperature, K
    tfeed = 2;          % purge step time, sec
    qfeed = 306/tfeed;  % purge mass flow rate, mmol/sec
    Pout = 1.0*1013250; % outlet pressure at i=1 during purge, g/cm/sec2
end
Qfeed = qfeed/A;        % mass flux, mmol/cm2/sec
Qin = FE * Qfeed;       % assign directions for the absolute value of Qfeed
% usfeed = Qfeed/rhog;  % superficial velocity, cm/sec
% I should not specify Pfeed because I need to find out the pressure drop
% starting from Pout using Ergun equation in momentum balance.
% Feed gas density is unknown because rhogfeed = Pfeed/(R*Tfeed) mmol/cm3
% Equilibrium capacity at feed inlet (i=1) or (i=n) is unknown bcoz Pfeed
%..........................................................................
% Bed initial conditions
%..........................................................................
if adsorptionORpurge == 111
    y10 = 1.00;         % saturated with 100% O2 + 0% N2
    P0 = 3.0*1013250;   % pressure after pressurization step, g/cm/sec2
    T0 = 298;           % temperature after pressurization step, K
end
if adsorptionORpurge == 222
    y10 = 0.21;         % saturated with 21% O2 + 79% N2
    P0 = 1.0*1013250;   % pressure after depressurization step, g/cm/sec2
    T0 = 298;           % temperature after depressurization step, K
end

Q0 = FE * 0;            % without gas flux, mmol/cm2/sec
rhog0 = P0/(R*T0);      % gas phase density, mmol/cm3
us0 = Q0/rhog0;         % superficial velocity, cm/sec
n1eqm0 = 1.11*10^(-3)*exp(1593.0/T0)*(y10)  *(P0/1013250)/(1 + 1.03*10^....
    (-4)*exp(2061.9/T0)*(y10)*(P0/1013250) + 2.07*10^(-4)*exp(2455.5/T0)...
    *(1-y10)*(P0/1013250));
n2eqm0 = 1.25*10^(-3)*exp(2168.6/T0)*(1-y10)*(P0/1013250)/(1 + 1.03*10^....
    (-4)*exp(2061.9/T0)*(y10)*(P0/1013250) + 2.07*10^(-4)*exp(2455.5/T0)...
    *(1-y10)*(P0/1013250));
n10 = n1eqm0;           % O2 adsorbed, mmol O2/g
n20 = n2eqm0;           % N2 adsorbed, mmol N2/g
% Langmuir isotherms of O2 and N2 on LiX from Salil U. Rege and Ralph T.
% Yang, Ind. Eng. Chem. Res. 1997, 36, 5358-5365.
Mg0 = Mg1*y10 + Mg2*(1-y10);          % average gas phase molecular weight,
g/mmol
%..........................................................................
% Important note: variables must be parameterized within the minimum and
% maximum range in order to avoid singular matrix of A.x = b
%..........................................................................
% Additional parameters required for proper nondimensionalization of
% dependent variables
%..........................................................................
Pmax = Pout;            % maximum pressure, g/cm/sec2
Pmin = 0;               % minimum pressure, g/cm/sec2

Tref = T0;
Tmax = Tfeed;           % maximum temperature, K
Tmin = 273;             % minimum temperature, K
%..........................................................................
% Nonlinear variable
%..........................................................................
% Y1bar = y1bar * rhogbar
%..........................................................................
% Initial conditions for y1bar, n1bar, n2bar for i = 1:n
%..........................................................................
% Note that boundary conditions at i = 1 (adsorption) or i = n (purge)
% after t=0 should not be brought in here because BC should be dealt with
% in Method of Lines routine
%..........................................................................
for i=1:n
    Y0(0*n+i) = y10*rhog0*L/(Qfeed*tfeed);      % for Y1bar
    Y0(1*n+i) = (k1*rhob*L/Qfeed)*n10;          % for n1bar
    Y0(2*n+i) = (k2*rhob*L/Qfeed)*n20;           % for n2bar
end
%...........................................................................
% ODE integration
%...........................................................................
ncall = 0;
reltol = 1.0e-04;
abstol = 1.0e-04;
options = odeset('RelTol',reltol,'AbsTol',abstol);

[t,Y] = ode15s(@psa_1_sequence_model_ABC, tbar, Y0, options);
% Embedded in MOL routine are 1st order (two-points) and 2nd order (three-
% points) upwind finite volume methods, and 2nd order TVD Superbee Flux
% Limiter to discretize dYbardZbar
%...........................................................................
% One vector to three vectors
%...........................................................................
for it = 1:nout     % time
for i = 1:n         % position
    Y1bar(it,i)     = Y(it,0*n+i);
    n1bar(it,i)     = Y(it,1*n+i);
    n2bar(it,i)     = Y(it,2*n+i);
end
end

for it = 1:nout     % time
for i = 1:n         % position
    y1bar(it,i)     = Y1bar(it,i)/(rhog0*L/(Qfeed*tfeed));
    Mg(it,i)        = Mg1*y1bar(it,i) + Mg2*(1-y1bar(it,i));
end
end
%...........................................................................
% Fix the boundary conditions
%...........................................................................
% ODE solver does take care of initial conditions at it = 1
for it = 2:nout     % time

if adsorptionORpurge == 111
   i = 1;
end
if adsorptionORpurge == 222
   i = n;
end

y1bar(it,i)         = y1feed;
end
%...........................................................................
% Isobaric, Isothermal
%...........................................................................
for it = 1:nout
    for i=1:n
    Pbar(it,i) = P0/(Pmax-Pmin);
    thetabar(it,i) = (T0-Tref)/(Tmax-Tmin);
    rhogbar(it,i) = rhog0*L/(Qfeed*tfeed);
    end
end
%...........................................................................
% Formulate n1eqmbar(i), n2eqmbar(i), Qbar(i), usbar(i) with initial and
% boundary conditions
%...........................................................................
```

```
for it = 1:nout
    for i = 1:n
      n1eqmbar(it,i) = (k1*rhob*L/Qfeed)*1.11*10^(-3)*exp(1593.0/((Tmax-...
          Tmin)*thetabar(it,i)+Tref))*(y1bar(it,i))  *((Pmax-Pmin)*........
          Pbar(it,i)/1013250)/(1 + 1.03*10^(-4)*exp(2061.9/((Tmax-Tmin)....
          *thetabar(it,i)+Tref))*(y1bar(it,i))*((Pmax-Pmin)*Pbar(it,i)/....
          1013250) + 2.07*10^(-4)*exp(2455.5/((Tmax-Tmin)*thetabar(it,i)...
          +Tref))*(1-y1bar(it,i))*((Pmax-Pmin)*Pbar(it,i)/1013250));

      n2eqmbar(it,i) = (k2*rhob*L/Qfeed)*1.25*10^(-3)*exp(2168.6/((Tmax-...
          Tmin)*thetabar(it,i)+Tref))*(1-y1bar(it,i))*((Pmax-Pmin)*........
          Pbar(it,i)/1013250)/(1 + 1.03*10^(-4)*exp(2061.9/((Tmax-Tmin)....
          *thetabar(it,i)+Tref))*(y1bar(it,i))*((Pmax-Pmin)*Pbar(it,i)/....
          1013250) + 2.07*10^(-4)*exp(2455.5/((Tmax-Tmin)*thetabar(it,i)...
          +Tref))*(1-y1bar(it,i))*((Pmax-Pmin)*Pbar(it,i)/1013250));
    end
end

for it = 1
    for i = 1:n
        Qbar(it,i) = Q0/Qfeed;
        usbar(it,i) = Qbar(it,i)/rhogbar(it,i);
    end
end

for it = 2:nout
    if adsorptionORpurge == 111
    for i = 1
        Qbar(it,i) = Qin/Qfeed;
        usbar(it,i) = Qbar(it,i)/rhogbar(it,i);
    end
    for i = 2:n
        Qbar(it,i) = -((n1eqmbar(it,i)-n1bar(it,i)) + (n2eqmbar(it,i)-.....
            n2bar(it,i)))*dzbar + Qbar(it,i-1);
        usbar(it,i) = Qbar(it,i)/rhogbar(it,i);
    end
    end

    if adsorptionORpurge == 222
    for i = n
        Qbar(it,i) = Qin/Qfeed;
        usbar(it,i) = Qbar(it,i)/rhogbar(it,i);
    end
    for i = n-1:-1:1
        Qbar(it,i) = +((n1eqmbar(it,i)-n1bar(it,i)) + (n2eqmbar(it,i)-.....
            n2bar(it,i)))*dzbar + Qbar(it,i+1);
        usbar(it,i) = Qbar(it,i)/rhogbar(it,i);
    end
    end
end
%..........................................................................
% Mass flux
%..........................................................................
AccQ = 0;
for i = 1:n
        AccQ = AccQ + Qfeed*abs(Qbar(nout,i));
end
AveQ_nout_mmolecm2sec = AccQ/n
AveQ_nout_lbmoleft2hr = AveQ_nout_mmolecm2sec*3600*929.0304/454000
%..........................................................................
```

```
% Formulate gas phase axial dispersion coefficient in mass balance,
% DL, cm2/sec
%..........................................................................
if MassAxial == 881
for i = 1:n
    DL(nout,i) = 0;
end
end

if MassAxial == 882
if adsorptionORpurge == 111
    DM = Dem4;
end
if adsorptionORpurge == 222
    DM = Dem1;
end
for i = 1:n
    DL(nout,i) = 0.7*DM + 0.5*dp*(abs(Qbar(nout,i))*Qfeed)/(...............
        rhogbar(nout,i)*(Qfeed*tfeed/L)*epsilonbar);
end
end

AccDL = 0;
for i = 1:n
    AccDL = AccDL + DL(nout,i);
end
AveDL_nout_cm2sec = AccDL/n
%..........................................................................
% Calculate Reynolds number of particle
%..........................................................................
AccReynolds = 0;
for i=1:n
    Reynolds(i) = (Qfeed*abs(Qbar(nout,i))*dp*Mg(nout,i))/miu;
    AccReynolds = AccReynolds + Reynolds(i);
end
AveReynolds_nout = AccReynolds/n
%..........................................................................
% Calculate pressure drop across the bed
%..........................................................................
if adsorptionORpurge == 111
Pressuredropacrossbed_psi = ((Pmax-Pmin).*Pbar(nout,1) - Pout)*14.69595/...
    1013250;
end

if adsorptionORpurge == 222
Pressuredropacrossbed_psi = ((Pmax-Pmin).*Pbar(nout,n) - Pout)*14.69595/...
    1013250;
end
%..........................................................................
% Display selected output
%..........................................................................
fprintf('\n abstol = %8.1e   reltol = %8.1e\n', abstol, reltol);
fprintf('\n ncall = %4d\n',ncall);
%..........................................................................
% Display important parameters
%..........................................................................
fprintf('\n process step = %d       dYbardZbar differential method = %d',
adsorptionORpurge, dYbardZbarDifferentialMethod);
```

```
fprintf('\n P0 = %g atm          T0 = %g K       yN20 = %g       Q0 = %g
mmol/cm2/sec', P0/1013250, T0, (1-y10), Q0);
fprintf('\n tfeed = %g sec      Tfeed = %g K    yN2feed = %g    Qfeed = %g
mmol/cm2/sec    qfeed = %g mmol/sec', tfeed, Tfeed, (1-y1feed), Qfeed,
qfeed);
fprintf('\n dp = %g um          L = %g cm        D = %g cm       Pressure drop
across bed at nout = %g psi', dp*10000, L, D, Pressuredropacrossbed_psi);
fprintf('\n k1 = %g sec-1      k2 = %g sec-1    rhob = %g g/cc\n\n', k1, k2,
rhob);
%..........................................................................
% Plot numerical solutions vs. independent variables (t,z)
%..........................................................................
z0 = 0.0;
zend = L;
z = linspace(z0,zend,n);

t0 = 0.0;
tend = tfeed;
t = linspace(t0,tend,nout);
%..........................................................................
if choose == 1 % start of choose == 1 : 2D subplots of y1, y2, n1, n2, T, P
%..........................................................................
% Subplots of y1, y2, n1, n2, T, P
%..........................................................................
figure(1);

subplot(2,2,1)
plot(t,y1bar(:,(n+1)/2),'-r',t,(1-y1bar(:,(n+1)/2)),'-b'); %axis tight
h = legend('y_{O2}','y_{N2}',1);
title('y_{O2} and y_{N2} vs. time at L/2'); xlabel('time (sec)'); .........
    ylabel('y_{O2} and y_{N2}')

subplot(2,2,2)
plot(t,(Qfeed/(k1*rhob*L)).*n1bar(:,(n+1)/2),'-r',t,(Qfeed/(k2*rhob*.......
    L)).*n2bar(:,(n+1)/2),'-b'); %axis tight
h = legend('n_{O2}','n_{N2}',1);
title('n_{O2} and n_{N2} vs. time at L/2'); xlabel('time (sec)'); .........
    ylabel('n_{O2} and n_{N2} (mmol/g adsorbent)')

subplot(2,2,3)
plot(t,Tref+(Tmax-Tmin).*thetabar(:,(n+1)/2)); %axis tight
title('T vs. time at L/2'); xlabel('time (sec)'); ylabel('T (K)')

subplot(2,2,4)
plot(z,((Pmax-Pmin)/1013250).*Pbar(nout,:)); %axis tight
title('P vs. bed axial position at tfinal'); ..............................
    xlabel('Bed axial position, z (cm)'); ylabel('P (atm)')
end % end of choose == 1
%..........................................................................
if choose == 2 % start of choose == 2 : Individual 2D plots
%..........................................................................
% Gas phase mole fraction, y1 and y2, vs. time at L/2
%..........................................................................
figure(2)
plot(t,y1bar(:,(n+1)/2),'-r>',t,(1-y1bar(:,(n+1)/2)),'-bs','LineWidth',....
    2,'MarkerSize',3)
h = legend('y_{O2}','y_{N2}',1);
xlabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
ylabel('Gas phase mole fraction, y_{O2} and y_{N2}','color','k',...........
    'fontsize',10,'fontweight','b')
```

```
title('Gas phase mole fraction vs. time at L/2','color','k','fontsize',....
    12,'fontweight','b')
axis([0 tfeed 0 1])
grid on
%.......................................................................
% Amount adsorbed on adsorbent, n1 and n2, vs. time at L/2
%.......................................................................
figure(3)
plot(t,(Qfeed/(k1*rhob*L)).*n1bar(:,(n+1)/2),'-
r>',t,(Qfeed/(k2*rhob*L)).*n2bar(:,(n+1)/2),'-
bs','LineWidth',2,'MarkerSize',3)
h = legend('n_{O2}','n_{N2}',1);
xlabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
ylabel('Amount adsorbed on adsorbent, n_{O2} and n_{N2} (mmol/g
adsorbent)','color','k','fontsize',10,'fontweight','b')
title('Amount adsorbed on adsorbent vs. time at
L/2','color','k','fontsize',12,'fontweight','b')
grid on
%.......................................................................
% Temperature, T, vs. time at L/2
%.......................................................................
figure(4)
plot(t,(Tref+(Tmax-Tmin).*thetabar(:,(n+1)/2)),'LineWidth',2)
xlabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
ylabel('Temperature, T (K)','color','k','fontsize',10,'fontweight','b')
title('Temperature vs. time at
L/2','color','k','fontsize',12,'fontweight','b')
grid on
%.......................................................................
% Pressure, P, vs. bed axial position at t=tfinal
%.......................................................................
figure(5)
plot(z,((Pmax-Pmin)/1013250).*Pbar(nout,:),'LineWidth',2)
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Pressure, P (atm)','color','k','fontsize',10,'fontweight','b')
title('Pressure vs. bed axial position at
tfinal','color','k','fontsize',12,'fontweight','b')
grid on
%.......................................................................
% Gas phase concentration, rhog, vs. bed axial position at t=tfinal
%.......................................................................
figure(6)
plot(z,(Qfeed*tfeed/L).*rhogbar(nout,:),'LineWidth',2)
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Gas phase concentration, \rho_g
(mmol/cm^3)','color','k','fontsize',10,'fontweight','b')
title('Gas phase concentration vs. bed axial position at
tfinal','color','k','fontsize',12,'fontweight','b')
grid on
%.......................................................................
% Mass flux, Q, vs. bed axial position at t=tfinal
%.......................................................................
figure(7)
plot(z,Qfeed.*Qbar(nout,:),'LineWidth',2)
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Mass flux, Q
(mmol/cm^2/s)','color','k','fontsize',10,'fontweight','b')
```

```
title('Mass flux vs. bed axial position at
tfinal','color','k','fontsize',12,'fontweight','b')
grid on
%...........................................................................
% Superficial flow rate, us, vs. bed axial position at t=tfinal
%...........................................................................
figure(8)
plot(z,(L/tfeed).*usbar(nout,:),'LineWidth',2)
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Superficial velocity, u_s
(cm/s)','color','k','fontsize',10,'fontweight','b')
title('Superficial velocity vs. bed axial position at
tfinal','color','k','fontsize',12,'fontweight','b')
grid on
end % end of choose == 2
%...........................................................................
if choose == 3 % start of choose == 3 : Surface plots of y1, y2, n1, n2, T,
P, rhog, Q, us
%...........................................................................
% Gas phase mole fraction, y1
%...........................................................................
figure(9)
% mesh(z,t,y1bar,'EdgeColor','black')
surf(z,t,y1bar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Gas phase mole fraction,
y_{O2}','color','k','fontsize',10,'fontweight','b')
axis tight
%...........................................................................
% Gas phase mole fraction, y2
%...........................................................................
figure(10)
% mesh(z,t,(1-y1bar),'EdgeColor','black')
surf(z,t,(1-y1bar))
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Gas phase mole fraction,
y_{N2}','color','k','fontsize',10,'fontweight','b')
axis tight
%...........................................................................
% Amount adsorbed on adsorbent, n1
%...........................................................................
figure(11)
% mesh(z,t,(Qfeed/(k1*rhob*L)).*n1bar,'EdgeColor','black')
surf(z,t,(Qfeed/(k1*rhob*L)).*n1bar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Amount adsorbed on adsorbent, n_{O2} (mmol/g
adsorbent)','color','k','fontsize',10,'fontweight','b')
axis tight
%...........................................................................
% Amount adsorbed on adsorbent, n2
```

```
%............................................................................
figure(12)
% mesh(z,t,(Qfeed/(k2*rhob*L)).*n2bar,'EdgeColor','black')
surf(z,t,(Qfeed/(k2*rhob*L)).*n2bar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Amount adsorbed on adsorbent, n_{N2} (mmol/g
adsorbent)','color','k','fontsize',10,'fontweight','b')
axis tight
%............................................................................
% Temperature, T
%............................................................................
figure(13)
% mesh(z,t,(Tmin+(Tmax-Tmin).*thetabar),'EdgeColor','black')
surf(z,t,(Tref+(Tmax-Tmin).*thetabar))
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Temperature, T (K)','color','k','fontsize',10,'fontweight','b')
axis tight
%............................................................................
% Pressure, P
%............................................................................
figure(14)
% mesh(z,t,((Pmax-Pmin)/1013250).*Pbar),'EdgeColor','black')
surf(z,t,((Pmax-Pmin)/1013250).*Pbar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Pressure, P (atm)','color','k','fontsize',10,'fontweight','b')
axis tight
%............................................................................
% Gas phase concentration, rhog
%............................................................................
figure(15)
% mesh(z,t,(Qfeed*tfeed/L).*rhogbar),'EdgeColor','black')
surf(z,t,(Qfeed*tfeed/L).*rhogbar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Gas phase concentration, \rho_g
(mmol/cm^3)','color','k','fontsize',10,'fontweight','b')
axis tight
%............................................................................
% Mass flux, Q
%............................................................................
figure(16)
% mesh(z,t,Qfeed.*Qbar),'EdgeColor','black')
surf(z,t,Qfeed.*Qbar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Mass flux, Q
(mmol/cm^2/s)','color','k','fontsize',10,'fontweight','b')
```

```
axis tight
%..........................................................................
% Superficial flow rate, us
%..........................................................................
figure(17)
% mesh(z,t,(L/tfeed).*usbar),'EdgeColor','black')
surf(z,t,(L/tfeed).*usbar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Superficial velocity, u_s
(cm/s)','color','k','fontsize',10,'fontweight','b')
axis tight
end % end of choose == 3
%..........................................................................
if choose == 4 % start of choose = 4 : The 2D plots that Dr. Sircar is
interested in
%..........................................................................
% Among nout points, take 5 time points spreading evenly between [0,tfeed]
%..........................................................................
for m=1:5
    it(m) = 1+ (m-1)*(nout-1)/4;
end
% If nout = 101 ==> it = 1    26    51    76   101

for m=1:5
    time((m)) = t(it(m));
end
% Resultstime = time          % Evenly spaced time for displayed results
%..........................................................................
% Now I want to take 5 time points between [0,tfeed] that display the
% results beautifully
%..........................................................................
% Resultstime = [t(1); t(11); t(21); t(31); t(41); t(51); t(101)]
%..........................................................................
% Adsorption / Desorption Profiles
%..........................................................................
% particle diameter in micrometer
dp = dp *10000;

if adsorptionORpurge == 111
% O2 initially present in the bed
O2ingas = epsilon*A*L*(y10)*rhog0;      % mmoles O2
O2insolid = A*L*rhob*n10;               % mmoles O2
TotalO2inbed0_mmolesO2 = O2ingas + O2insolid    % mmoles O2

% O2 remaining at every it, averaged over the whole bed i=1:n
for it = 1:nout
    y1acc(it) = 0;
    rhogbaracc(it) = 0;
    n1baracc(it) = 0;
    for i = 1:n
        y1acc(it) = y1bar(it,i) + y1acc(it);
        rhogbaracc(it) = rhogbar(it,i) + rhogbaracc(it);
        n1baracc(it) = n1bar(it,i) + n1baracc(it);
    end
    y1ave(it) = y1acc(it)/n;
    rhogave(it) = (Qfeed*tfeed/L)*rhogbaracc(it)/n;
    n1ave(it) = (Qfeed/(k1*rhob*L))*n1baracc(it)/n;
```

```
end

for it = 1:nout
O2remainingas(it) = epsilon*A*L*y1ave(it)*rhogave(it);    % mmoles O2
O2remaininsolid(it) = A*L*rhob*n1ave(it);                 % mmoles O2
TotalO2remaininbed_mmolesO2(it) = O2remainingas(it) + O2remaininsolid(it);
% mmoles O2
frac_O2_desorbed(it) = (TotalO2inbed0_mmolesO2 -
TotalO2remaininbed_mmolesO2(it))/TotalO2inbed0_mmolesO2;
end

frac_O2_desorbedv = frac_O2_desorbed';
end


if adsorptionORpurge == 222
% N2 initially present in the bed
N2ingas = epsilon*A*L*(1-y10)*rhog0;     % mmoles N2
N2insolid = A*L*rhob*n20;                % mmoles N2
TotalN2inbed0_mmolesN2 = N2ingas + N2insolid     % mmoles N2

% N2 remaining at every it, averaged over the whole bed i=1:n
for it = 1:nout
    y2acc(it) = 0;
    rhogbaracc(it) = 0;
    n2baracc(it) = 0;
    for i = 1:n
        y2acc(it) = (1-y1bar(it,i)) + y2acc(it);
        rhogbaracc(it) = rhogbar(it,i) + rhogbaracc(it);
        n2baracc(it) = n2bar(it,i) + n2baracc(it);
    end
    y2ave(it) = y2acc(it)/n;
    rhogave(it) = (Qfeed*tfeed/L)*rhogbaracc(it)/n;
    n2ave(it) = (Qfeed/(k2*rhob*L))*n2baracc(it)/n;
end

for it = 1:nout
N2remainingas(it) = epsilon*A*L*y2ave(it)*rhogave(it);    % mmoles N2
N2remaininsolid(it) = A*L*rhob*n2ave(it);                 % mmoles N2
TotalN2remaininbed_mmolesN2(it) = N2remainingas(it) + N2remaininsolid(it);
% mmoles N2
frac_N2_desorbed(it) = (TotalN2inbed0_mmolesN2 -
TotalN2remaininbed_mmolesN2(it))/TotalN2inbed0_mmolesN2;
end

frac_N2_desorbedv = frac_N2_desorbed';
end
%...........................................................................

%plot(zbar,(1-y1bar(1,:)),zbar,(1-y1bar((nout-1)*0.005+1,:)),zbar,(1-
y1bar((nout-1)*0.01+1,:)),zbar,(1-y1bar((nout-1)*0.05+1,:)),zbar,(1-
y1bar((nout-1)*0.1+1,:)),zbar,(1-y1bar((nout-1)*0.5+1,:)),zbar,(1-
y1bar(nout,:)),'LineWidth',2,'MarkerSize',3); %axis tight
%h =
legend('tbar=0','tbar=0.005','tbar=0.010','tbar=0.050','tbar=0.100','tbar=0.5
00','tbar=1.000',1);

%plot(zbar,(1-y1bar(1,:)),zbar,(1-y1bar((nout-1)*0.1+1,:)),zbar,(1-
y1bar((nout-1)*0.2+1,:)),zbar,(1-y1bar((nout-1)*0.3+1,:)),zbar,(1-
y1bar((nout-1)*0.4+1,:)),zbar,(1-y1bar((nout-1)*0.5+1,:)),zbar,(1-
y1bar((nout-1)*0.6+1,:)),zbar,(1-y1bar((nout-1)*0.7+1,:)),zbar,(1-
y1bar((nout-1)*0.8+1,:)),zbar,(1-y1bar((nout-1)*0.9+1,:)),zbar,(1-
y1bar(nout,:)),'LineWidth',2,'MarkerSize',3); %axis tight
%h =
legend('tbar=0.0','tbar=0.1','tbar=0.2','tbar=0.3','tbar=0.4','tbar=0.5','tba
r=0.6','tbar=0.7','tbar=0.8','tbar=0.9','tbar=1.0',1);


figure(20)
plot(zbar,(y1bar(1,:)),zbar,(y1bar((nout-1)*0.2+1,:)),zbar,(y1bar((nout-
1)*0.4+1,:)),zbar,(y1bar((nout-1)*0.6+1,:)),zbar,(y1bar((nout-
1)*0.8+1,:)),zbar,(y1bar(nout,:)),'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
text(.02,1.05,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('y_{O2}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.1])


figure(21)
plot(zbar,(1-y1bar(1,:)),zbar,(1-y1bar((nout-1)*0.2+1,:)),zbar,(1-
y1bar((nout-1)*0.4+1,:)),zbar,(1-y1bar((nout-1)*0.6+1,:)),zbar,(1-
y1bar((nout-1)*0.8+1,:)),zbar,(1-
y1bar(nout,:)),'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
text(.02,1.05,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('y_{N2}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.1])


figure(22)
plot(zbar,((Qfeed/(k1*rhob*L)).*n1bar(1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar(
(nout-1)*0.2+1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar((nout-
1)*0.4+1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar((nout-
1)*0.6+1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar((nout-
1)*0.8+1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar(nout,:)),'LineWidth',2,'MarkerS
ize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
```

```
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{O2} (mmoles O_2/g)','color','k','fontsize',12,'fontweight','b')


figure(23)
plot(zbar,((Qfeed/(k2*rhob*L)).*n2bar(1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar(
(nout-1)*0.2+1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar((nout-
1)*0.4+1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar((nout-
1)*0.6+1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar((nout-
1)*0.8+1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar(nout,:)),'LineWidth',2,'MarkerS
ize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{N2} (mmoles N_2/g)','color','k','fontsize',12,'fontweight','b')


figure(24)
plot(zbar,((Tmax-Tmin).*thetabar(1,:)+Tref),zbar,((Tmax-
Tmin).*thetabar((nout-1)*0.2+1,:)+Tref),zbar,((Tmax-Tmin).*thetabar((nout-
1)*0.4+1,:)+Tref),zbar,((Tmax-Tmin).*thetabar((nout-
1)*0.6+1,:)+Tref),zbar,((Tmax-Tmin).*thetabar((nout-
1)*0.8+1,:)+Tref),zbar,((Tmax-Tmin).*thetabar((nout-
1)*1.0+1,:)+Tref),'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('T_g = T_s (K)','color','k','fontsize',12,'fontweight','b')
axis([0 1 282 300])


figure(25)
plot(zbar,(Pmax-Pmin).*Pbar(1,:)*14.69595/1013250,zbar,(Pmax-
Pmin).*Pbar((nout-1)*0.2+1,:)*14.69595/1013250,zbar,(Pmax-Pmin).*Pbar((nout-
1)*0.4+1,:)*14.69595/1013250,zbar,(Pmax-Pmin).*Pbar((nout-
1)*0.6+1,:)*14.69595/1013250,zbar,(Pmax-Pmin).*Pbar((nout-
1)*0.8+1,:)*14.69595/1013250,zbar,(Pmax-Pmin).*Pbar((nout-
1)*1.0+1,:)*14.69595/1013250,'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('Pressure (psi)','color','k','fontsize',12,'fontweight','b')


if adsorptionORpurge == 222
%.............................................................................
% Search the tbar that produces fracdes = 0.3, 0.5, 0.7
% Then plot the profiles for yN2, nN2, Tg, Ts at these tbar
%.............................................................................
minDF_3 = 1; % minimum difference between fraction desorbed and 0.3
for it=1:nout
    F_3 = abs(frac_N2_desorbed(1,it) - 0.3);
    if minDF_3 > F_3
        minDF_3 = F_3;
        it_3 = it;
    end
end
tbar_3 = tbarv(it_3,1)
minDF_3

minDF_5 = 1; % minimum difference between fraction desorbed and 0.5
for it=1:nout
    F_5 = abs(frac_N2_desorbed(1,it) - 0.5);
    if minDF_5 > F_5
        minDF_5 = F_5;
        it_5 = it;
    end
end
tbar_5 = tbarv(it_5,1)
minDF_5

minDF_7 = 1; % minimum difference between fraction desorbed and 0.7
for it=1:nout
    F_7 = abs(frac_N2_desorbed(1,it) - 0.7);
    if minDF_7 > F_7
        minDF_7 = F_7;
        it_7 = it;
    end
end
tbar_7 = tbarv(it_7,1)
minDF_7

minDF_8 = 1; % minimum difference between fraction desorbed and 0.8
for it=1:nout
    F_8 = abs(frac_N2_desorbed(1,it) - 0.8);
    if minDF_8 > F_8
        minDF_8 = F_8;
        it_8 = it;
    end
end
tbar_8 = tbarv(it_8,1)
minDF_8

minDF_85 = 1; % minimum difference between fraction desorbed and 0.85
for it=1:nout
    F_85 = abs(frac_N2_desorbed(1,it) - 0.85);
    if minDF_85 > F_85
        minDF_85 = F_85;
        it_85 = it;
    end
end
tbar_85 = tbarv(it_85,1)
minDF_85

minDF_9 = 1; % minimum difference between fraction desorbed and 0.9
for it=1:nout
    F_9 = abs(frac_N2_desorbed(1,it) - 0.9);
    if minDF_9 > F_9
        minDF_9 = F_9;
        it_9 = it;
    end
end
tbar_9 = tbarv(it_9,1)
minDF_9

minDF_95 = 1; % minimum difference between fraction desorbed and 0.95
for it=1:nout
    F_95 = abs(frac_N2_desorbed(1,it) - 0.95);
    if minDF_95 > F_95
        minDF_95 = F_95;
        it_95 = it;
    end
end
tbar_95 = tbarv(it_95,1)
minDF_95
%..........................................................................
for i = 1:n
    Nn1f357(1,i)=(Qfeed/(k1*rhob*L))*n1bar(1   ,i)/n10; % normalized with n10
    Nn1f357(2,i)=(Qfeed/(k1*rhob*L))*n1bar(it_3,i)/n10;
    Nn1f357(3,i)=(Qfeed/(k1*rhob*L))*n1bar(it_5,i)/n10;
    Nn1f357(4,i)=(Qfeed/(k1*rhob*L))*n1bar(it_7,i)/n10;
    Nn1f357(5,i)=(Qfeed/(k1*rhob*L))*n1bar(it_8,i)/n10;
    Nn1f357(6,i)=(Qfeed/(k1*rhob*L))*n1bar(it_85,i)/n10;
    Nn1f357(7,i)=(Qfeed/(k1*rhob*L))*n1bar(it_9,i)/n10;
    Nn1f357(8,i)=(Qfeed/(k1*rhob*L))*n1bar(it_95,i)/n10;

    n1f357(1,i)=(Qfeed/(k1*rhob*L))*n1bar(1   ,i); % mmol O2/g
    n1f357(2,i)=(Qfeed/(k1*rhob*L))*n1bar(it_3,i);
    n1f357(3,i)=(Qfeed/(k1*rhob*L))*n1bar(it_5,i);
    n1f357(4,i)=(Qfeed/(k1*rhob*L))*n1bar(it_7,i);
    n1f357(5,i)=(Qfeed/(k1*rhob*L))*n1bar(it_8,i);
    n1f357(6,i)=(Qfeed/(k1*rhob*L))*n1bar(it_85,i);
    n1f357(7,i)=(Qfeed/(k1*rhob*L))*n1bar(it_9,i);
    n1f357(8,i)=(Qfeed/(k1*rhob*L))*n1bar(it_95,i);

    Nn2f357(1,i)=(Qfeed/(k2*rhob*L))*n2bar(1   ,i)/n20; % normalized with n20
    Nn2f357(2,i)=(Qfeed/(k2*rhob*L))*n2bar(it_3,i)/n20;
    Nn2f357(3,i)=(Qfeed/(k2*rhob*L))*n2bar(it_5,i)/n20;
    Nn2f357(4,i)=(Qfeed/(k2*rhob*L))*n2bar(it_7,i)/n20;
    Nn2f357(5,i)=(Qfeed/(k2*rhob*L))*n2bar(it_8,i)/n20;
    Nn2f357(6,i)=(Qfeed/(k2*rhob*L))*n2bar(it_85,i)/n20;
    Nn2f357(7,i)=(Qfeed/(k2*rhob*L))*n2bar(it_9,i)/n20;
    Nn2f357(8,i)=(Qfeed/(k2*rhob*L))*n2bar(it_95,i)/n20;

    n2f357(1,i)=(Qfeed/(k2*rhob*L))*n2bar(1   ,i); % mmol N2/g
    n2f357(2,i)=(Qfeed/(k2*rhob*L))*n2bar(it_3,i);
    n2f357(3,i)=(Qfeed/(k2*rhob*L))*n2bar(it_5,i);
    n2f357(4,i)=(Qfeed/(k2*rhob*L))*n2bar(it_7,i);
    n2f357(5,i)=(Qfeed/(k2*rhob*L))*n2bar(it_8,i);
    n2f357(6,i)=(Qfeed/(k2*rhob*L))*n2bar(it_85,i);
    n2f357(7,i)=(Qfeed/(k2*rhob*L))*n2bar(it_9,i);
    n2f357(8,i)=(Qfeed/(k2*rhob*L))*n2bar(it_95,i);

    totn10=(A*L*rhob)*n10 + epsilon*A*L*(y10)*rhog0;    % total initial mmol
O2 in solid and gas phases
    totn20=(A*L*rhob)*n20 + epsilon*A*L*(1-y10)*rhog0; % total initial mmol
N2 in solid and gas phases

    totn1f357(1,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(1,i) +
epsilon*A*L*(y1bar(1,i))*rhogbar(1,i)*(Qfeed*tfeed/L); % total mmol O2 in gas
and solid phases
    totn1f357(2,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_3,i) +
epsilon*A*L*(y1bar(it_3,i))*rhogbar(it_3,i)*(Qfeed*tfeed/L);
    totn1f357(3,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_5,i) +
epsilon*A*L*(y1bar(it_5,i))*rhogbar(it_5,i)*(Qfeed*tfeed/L);
    totn1f357(4,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_7,i) +
epsilon*A*L*(y1bar(it_7,i))*rhogbar(it_7,i)*(Qfeed*tfeed/L);
    totn1f357(5,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_8,i) +
epsilon*A*L*(y1bar(it_8,i))*rhogbar(it_8,i)*(Qfeed*tfeed/L);
    totn1f357(6,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_85,i) +
epsilon*A*L*(y1bar(it_85,i))*rhogbar(it_85,i)*(Qfeed*tfeed/L);
    totn1f357(7,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_9,i) +
epsilon*A*L*(y1bar(it_9,i))*rhogbar(it_9,i)*(Qfeed*tfeed/L);
    totn1f357(8,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_95,i) +
epsilon*A*L*(y1bar(it_95,i))*rhogbar(it_95,i)*(Qfeed*tfeed/L);

    Ntotn1f357(1,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(1,i) +
epsilon*A*L*(y1bar(1,i))*rhogbar(1,i)*(Qfeed*tfeed/L))/totn10; % normalized
with totn10
    Ntotn1f357(2,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_3,i) +
epsilon*A*L*(y1bar(it_3,i))*rhogbar(it_3,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(3,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_5,i) +
epsilon*A*L*(y1bar(it_5,i))*rhogbar(it_5,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(4,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_7,i) +
epsilon*A*L*(y1bar(it_7,i))*rhogbar(it_7,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(5,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_8,i) +
epsilon*A*L*(y1bar(it_8,i))*rhogbar(it_8,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(6,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_85,i) +
epsilon*A*L*(y1bar(it_85,i))*rhogbar(it_85,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(7,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_9,i) +
epsilon*A*L*(y1bar(it_9,i))*rhogbar(it_9,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(8,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_95,i) +
epsilon*A*L*(y1bar(it_95,i))*rhogbar(it_95,i)*(Qfeed*tfeed/L))/totn10;

    totn2f357(1,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(1,i) +
epsilon*A*L*(1-y1bar(1,i))*rhogbar(1,i)*(Qfeed*tfeed/L); % total mmol N2 in
gas and solid phases
```

```
    totn2f357(2,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_3,i) +
epsilon*A*L*(1-y1bar(it_3,i))*rhogbar(it_3,i)*(Qfeed*tfeed/L);
    totn2f357(3,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_5,i) +
epsilon*A*L*(1-y1bar(it_5,i))*rhogbar(it_5,i)*(Qfeed*tfeed/L);
    totn2f357(4,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_7,i) +
epsilon*A*L*(1-y1bar(it_7,i))*rhogbar(it_7,i)*(Qfeed*tfeed/L);
    totn2f357(5,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_8,i) +
epsilon*A*L*(1-y1bar(it_8,i))*rhogbar(it_8,i)*(Qfeed*tfeed/L);
    totn2f357(6,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_85,i) +
epsilon*A*L*(1-y1bar(it_85,i))*rhogbar(it_85,i)*(Qfeed*tfeed/L);
    totn2f357(7,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_9,i) +
epsilon*A*L*(1-y1bar(it_9,i))*rhogbar(it_9,i)*(Qfeed*tfeed/L);
    totn2f357(8,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_95,i) +
epsilon*A*L*(1-y1bar(it_95,i))*rhogbar(it_95,i)*(Qfeed*tfeed/L);

    Ntotn2f357(1,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(1,i) +
epsilon*A*L*(1-y1bar(1,i))*rhogbar(1,i)*(Qfeed*tfeed/L))/totn20; % normalized
with totn20
    Ntotn2f357(2,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_3,i) +
epsilon*A*L*(1-y1bar(it_3,i))*rhogbar(it_3,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(3,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_5,i) +
epsilon*A*L*(1-y1bar(it_5,i))*rhogbar(it_5,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(4,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_7,i) +
epsilon*A*L*(1-y1bar(it_7,i))*rhogbar(it_7,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(5,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_8,i) +
epsilon*A*L*(1-y1bar(it_8,i))*rhogbar(it_8,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(6,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_85,i) +
epsilon*A*L*(1-y1bar(it_85,i))*rhogbar(it_85,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(7,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_9,i) +
epsilon*A*L*(1-y1bar(it_9,i))*rhogbar(it_9,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(8,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_95,i) +
epsilon*A*L*(1-y1bar(it_95,i))*rhogbar(it_95,i)*(Qfeed*tfeed/L))/totn20;

    Remain_N2_inbed_percent(1,i) = 100*totn2f357(1,i)/(totn1f357(1,i) +
totn2f357(1,i));
    Remain_N2_inbed_percent(2,i) = 100*totn2f357(2,i)/(totn1f357(2,i) +
totn2f357(2,i));
    Remain_N2_inbed_percent(3,i) = 100*totn2f357(3,i)/(totn1f357(3,i) +
totn2f357(3,i));
    Remain_N2_inbed_percent(4,i) = 100*totn2f357(4,i)/(totn1f357(4,i) +
totn2f357(4,i));
    Remain_N2_inbed_percent(5,i) = 100*totn2f357(5,i)/(totn1f357(5,i) +
totn2f357(5,i));
    Remain_N2_inbed_percent(6,i) = 100*totn2f357(6,i)/(totn1f357(6,i) +
totn2f357(6,i));
    Remain_N2_inbed_percent(7,i) = 100*totn2f357(7,i)/(totn1f357(7,i) +
totn2f357(7,i));
    Remain_N2_inbed_percent(8,i) = 100*totn2f357(8,i)/(totn1f357(8,i) +
totn2f357(8,i));

    y1barf357(1,i)=y1bar(1   ,i);
    y1barf357(2,i)=y1bar(it_3,i);
    y1barf357(3,i)=y1bar(it_5,i);
    y1barf357(4,i)=y1bar(it_7,i);
    y1barf357(5,i)=y1bar(it_8,i);
    y1barf357(6,i)=y1bar(it_85,i);
    y1barf357(7,i)=y1bar(it_9,i);
    y1barf357(8,i)=y1bar(it_95,i);

    y2barf357(1,i)=1-y1bar(1   ,i);
    y2barf357(2,i)=1-y1bar(it_3,i);
    y2barf357(3,i)=1-y1bar(it_5,i);
    y2barf357(4,i)=1-y1bar(it_7,i);
    y2barf357(5,i)=1-y1bar(it_8,i);
    y2barf357(6,i)=1-y1bar(it_85,i);
    y2barf357(7,i)=1-y1bar(it_9,i);
    y2barf357(8,i)=1-y1bar(it_95,i);

    Tgf357(1,i)=(Tmax-Tmin)*thetabar(1   ,i)+Tref; % gas phase temperature, K
    Tgf357(2,i)=(Tmax-Tmin)*thetabar(it_3,i)+Tref;
    Tgf357(3,i)=(Tmax-Tmin)*thetabar(it_5,i)+Tref;
    Tgf357(4,i)=(Tmax-Tmin)*thetabar(it_7,i)+Tref;
    Tgf357(5,i)=(Tmax-Tmin)*thetabar(it_8,i)+Tref;
    Tgf357(6,i)=(Tmax-Tmin)*thetabar(it_85,i)+Tref;
    Tgf357(7,i)=(Tmax-Tmin)*thetabar(it_9,i)+Tref;
    Tgf357(8,i)=(Tmax-Tmin)*thetabar(it_95,i)+Tref;

    thetabargf357(1,i)=thetabar(1   ,i);
    thetabargf357(2,i)=thetabar(it_3,i);
    thetabargf357(3,i)=thetabar(it_5,i);
    thetabargf357(4,i)=thetabar(it_7,i);
    thetabargf357(5,i)=thetabar(it_8,i);
    thetabargf357(6,i)=thetabar(it_85,i);
    thetabargf357(7,i)=thetabar(it_9,i);
    thetabargf357(8,i)=thetabar(it_95,i);

    Tsf357(1,i)=(Tmax-Tmin)*thetabar(1   ,i)+Tref; % solid phase temperature,
K
    Tsf357(2,i)=(Tmax-Tmin)*thetabar(it_3,i)+Tref;
    Tsf357(3,i)=(Tmax-Tmin)*thetabar(it_5,i)+Tref;
    Tsf357(4,i)=(Tmax-Tmin)*thetabar(it_7,i)+Tref;
    Tsf357(5,i)=(Tmax-Tmin)*thetabar(it_8,i)+Tref;
    Tsf357(6,i)=(Tmax-Tmin)*thetabar(it_85,i)+Tref;
    Tsf357(7,i)=(Tmax-Tmin)*thetabar(it_9,i)+Tref;
    Tsf357(8,i)=(Tmax-Tmin)*thetabar(it_95,i)+Tref;

    thetabarsf357(1,i)=thetabar(1   ,i);
    thetabarsf357(2,i)=thetabar(it_3,i);
    thetabarsf357(3,i)=thetabar(it_5,i);
    thetabarsf357(4,i)=thetabar(it_7,i);
    thetabarsf357(5,i)=thetabar(it_8,i);
    thetabarsf357(6,i)=thetabar(it_85,i);
    thetabarsf357(7,i)=thetabar(it_9,i);
    thetabarsf357(8,i)=thetabar(it_95,i);

    Pbarf357(1,i)=Pbar(1   ,i); % pressure in dimensionless
    Pbarf357(2,i)=Pbar(it_3,i);
    Pbarf357(3,i)=Pbar(it_5,i);
    Pbarf357(4,i)=Pbar(it_7,i);
    Pbarf357(5,i)=Pbar(it_8,i);
    Pbarf357(6,i)=Pbar(it_85,i);
    Pbarf357(7,i)=Pbar(it_9,i);
    Pbarf357(8,i)=Pbar(it_95,i);
end
%..........................................................................
figure(28)
plot(zbar,y1barf357(1,:), zbar,y1barf357(2,:), zbar,y1barf357(3,:),
zbar,y1barf357(4,:), zbar,y1barf357(5,:), zbar,y1barf357(6,:) ,
zbar,y1barf357(7,:), zbar,y1barf357(8,:),'LineWidth',2,'MarkerSize',3); %axis
tight
```

```
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
text(.02,1.25,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
text(.02,1.15,['\fontsize{9}(f=0.30 at tbar=',num2str(tbar_3),');
\fontsize{9}(f=0.50 at tbar=',num2str(tbar_5) ,');  \fontsize{9}(f=0.70 at
tbar=',num2str(tbar_7),');'])
text(.02,1.07,['\fontsize{9}(f=0.80 at tbar=',num2str(tbar_8),');
\fontsize{9}(f=0.85 at tbar=',num2str(tbar_85),');  \fontsize{9}(f=0.90 at
tbar=',num2str(tbar_9),');  \fontsize{9}(f=0.95 at
tbar=',num2str(tbar_95),');'])
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('y_{O2}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.3])

figure(29)
plot(zbar,y2barf357(1,:), zbar,y2barf357(2,:), zbar,y2barf357(3,:),
zbar,y2barf357(4,:), zbar,y2barf357(5,:), zbar,y2barf357(6,:) ,
zbar,y2barf357(7,:), zbar,y2barf357(8,:),'LineWidth',2,'MarkerSize',3); %axis
tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
text(.02,1.25,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
text(.02,1.15,['\fontsize{9}(f=0.30 at tbar=',num2str(tbar_3),');
\fontsize{9}(f=0.50 at tbar=',num2str(tbar_5) ,');  \fontsize{9}(f=0.70 at
tbar=',num2str(tbar_7),');'])
text(.02,1.07,['\fontsize{9}(f=0.80 at tbar=',num2str(tbar_8),');
\fontsize{9}(f=0.85 at tbar=',num2str(tbar_85),');  \fontsize{9}(f=0.90 at
tbar=',num2str(tbar_9),');  \fontsize{9}(f=0.95 at
tbar=',num2str(tbar_95),');'])
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('y_{N2}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.3])

figure(30)
plot(zbar,n1f357(1,:), zbar,n1f357(2,:), zbar,n1f357(3,:), zbar,n1f357(4,:),
zbar,n1f357(5,:), zbar,n1f357(6,:), zbar,n1f357(7,:), zbar,n1f357(8,:)
,'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{O2} (mmoles O_2/g)','color','k','fontsize',12,'fontweight','b')

figure(31)
plot(zbar,n2f357(1,:), zbar,n2f357(2,:), zbar,n2f357(3,:), zbar,n2f357(4,:),
zbar,n2f357(5,:), zbar,n2f357(6,:), zbar,n2f357(7,:), zbar,n2f357(8,:)
,'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{N2} (mmoles N_2/g)','color','k','fontsize',12,'fontweight','b')

figure(32)
plot(zbar,Nn1f357(1,:), zbar,Nn1f357(2,:), zbar,Nn1f357(3,:),
zbar,Nn1f357(4,:), zbar,Nn1f357(5,:), zbar,Nn1f357(6,:), zbar,Nn1f357(7,:),
zbar,Nn1f357(8,:) ,'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
```

```
ylabel('n_{O2}/n_{O2_0}','color','k','fontsize',12,'fontweight','b')

figure(33)
plot(zbar,Nn2f357(1,:), zbar,Nn2f357(2,:), zbar,Nn2f357(3,:),
zbar,Nn2f357(4,:), zbar,Nn2f357(5,:), zbar,Nn2f357(6,:), zbar,Nn2f357(7,:),
zbar,Nn2f357(8,:) ,'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
text(.02,1.25,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
text(.02,1.15,['\fontsize{9}(f=0.30 at tbar=',num2str(tbar_3),');
\fontsize{9}(f=0.50 at tbar=',num2str(tbar_5) ,');  \fontsize{9}(f=0.70 at
tbar=',num2str(tbar_7),');'])
text(.02,1.07,['\fontsize{9}(f=0.80 at tbar=',num2str(tbar_8),');
\fontsize{9}(f=0.85 at tbar=',num2str(tbar_85),');  \fontsize{9}(f=0.90 at
tbar=',num2str(tbar_9),');  \fontsize{9}(f=0.95 at
tbar=',num2str(tbar_95),');'])
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{N2}/n_{N2_0}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.3])

figure(34)
plot(zbar,totn2f357(1,:), zbar,totn2f357(2,:), zbar,totn2f357(3,:),
zbar,totn2f357(4,:), zbar,totn2f357(5,:), zbar,totn2f357(6,:),
zbar,totn2f357(7,:), zbar,totn2f357(8,:) ,'LineWidth',2,'MarkerSize',3);
%axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('total N_2 in gas and solid phases, totn_{N2}, (mmoles
N_2)','color','k','fontsize',12,'fontweight','b')

figure(35)
plot(zbar,Ntotn2f357(1,:), zbar,Ntotn2f357(2,:), zbar,Ntotn2f357(3,:),
zbar,Ntotn2f357(4,:), zbar,Ntotn2f357(5,:), zbar,Ntotn2f357(6,:),
zbar,Ntotn2f357(7,:), zbar,Ntotn2f357(8,:) ,'LineWidth',2,'MarkerSize',3);
%axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
text(.02,1.25,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
text(.02,1.15,['\fontsize{9}(f=0.30 at tbar=',num2str(tbar_3),');
\fontsize{9}(f=0.50 at tbar=',num2str(tbar_5) ,');  \fontsize{9}(f=0.70 at
tbar=',num2str(tbar_7),');'])
text(.02,1.07,['\fontsize{9}(f=0.80 at tbar=',num2str(tbar_8),');
\fontsize{9}(f=0.85 at tbar=',num2str(tbar_85),');  \fontsize{9}(f=0.90 at
tbar=',num2str(tbar_9),');  \fontsize{9}(f=0.95 at
tbar=',num2str(tbar_95),');'])
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('totn_{N2}/totn_{N2_0}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.3])

figure(36)
plot(zbar,(Pmax-Pmin).*Pbarf357(1,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(2,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(3,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(4,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(5,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(6,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(7,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(8,:)*14.69595/1013250,'LineWidth',2,'MarkerSize',3); %axis
tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('Pressure (psi)','color','k','fontsize',12,'fontweight','b')

figure(37)
plot(zbar,Tgf357(1,:),'--k',zbar,Tsf357(1,:),'-k',zbar,Tgf357(2,:),'--
m',zbar,Tsf357(2,:),'-m', zbar,Tgf357(3,:),'--b',zbar,Tsf357(3,:),'-b',
zbar,Tgf357(4,:),'--r',zbar,Tsf357(4,:),'-r', zbar,Tgf357(5,:),'--
g',zbar,Tsf357(5,:),'-g', zbar,Tgf357(6,:),'--y', zbar,Tsf357(6,:),'-y',
zbar,Tgf357(7,:),'--c',zbar,Tsf357(7,:),'-c', zbar,Tgf357(8,:),'--p',
zbar,Tsf357(8,:),'-p','LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('Tg_{f=0.0}','Ts_{f=0.0}','Tg_{f=0.3}','Ts_{f=0.3}','Tg_{f=0.5}','Ts_{
f=0.5}','Tg_{f=0.7}','Ts_{f=0.7}','Tg_{f=0.8}','Ts_{f=0.8}','Tg_{f=0.85}','Ts
_{f=0.85}','Tg_{f=0.9}','Ts_{f=0.9}','Tg_{f=0.95}','Ts_{f=0.95}',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('T_g and T_s (K)','color','k','fontsize',12,'fontweight','b')

figure(38)
plot(zbar,Tgf357(1,:)-Tref,'--k',zbar,Tsf357(1,:)-Tref,'-k',zbar,Tgf357(2,:)-
Tref,'--m',zbar,Tsf357(2,:)-Tref,'-m', zbar,Tgf357(3,:)-Tref,'--
b',zbar,Tsf357(3,:)-Tref,'-b', zbar,Tgf357(4,:)-Tref,'--r',zbar,Tsf357(4,:)-
Tref,'-r', zbar,Tgf357(5,:)-Tref,'--g',zbar,Tsf357(5,:)-Tref,'-g',
zbar,Tgf357(6,:)-Tref,'--y', zbar,Tsf357(6,:)-Tref,'-y', zbar,Tgf357(7,:)-
Tref,'--c',zbar,Tsf357(7,:)-Tref,'-c', zbar,Tgf357(8,:)-Tref,'--p',
zbar,Tsf357(8,:)-Tref,'-p', 'LineWidth',2,'MarkerSize',3); %axis tight
h = legend('Tg-T0_{f=0.0}','Ts-T0_{f=0.0}','Tg-T0_{f=0.3}','Ts-
T0_{f=0.3}','Tg-T0_{f=0.5}','Ts-T0_{f=0.5}','Tg-T0_{f=0.7}','Ts-
T0_{f=0.7}','Tg-T0_{f=0.8}','Ts-T0_{f=0.8}','Tg-T0_{f=0.85}','Ts-
T0_{f=0.85}','Tg-T0_{f=0.9}','Ts-T0_{f=0.9}','Tg-T0_{f=0.95}','Ts-
T0_{f=0.95}',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('(T_g - T_0) and (T_s - T_0)
(K)','color','k','fontsize',12,'fontweight','b')
end
end % end of choose == 4
%.......................................................................
```

Appendix 3

Models A, B, C

MOL Routine

```
%..........................................................................
% File: psa_1_sequence_model_ABC
%..........................................................................
% Chai Siew Wah, Chemical Engineering, Lehigh University, 20 May 2011
%..........................................................................
% MATHEMATICAL MODEL OF PRESSURE SWING ADSORPTION SYSTEM FOR ONE STEP :
% ADSORPTION OR PURGE
%..........................................................................
%
%                  :......................................:
%   ADSORPTION --> :                                      : --> PRODUCT
%   ADSORPTION --> :                                      : --> PRODUCT
%   ADSORPTION --> :                                      : --> PRODUCT
%                  :......................................:
%                 i=1                                    i=n
%
%                  :......................................:
%     EXHAUST <--  :                                      : <-- PURGE
%     EXHAUST <--  :                                      : <-- PURGE
%     EXHAUST <--  :                                      : <-- PURGE
%                  :......................................:
%                 i=1                                    i=n
%..........................................................................
% Assumptions
%..........................................................................
% (1) Ideal gas law
% (2) Empirical Langmuirian type binary isotherms - from Rege and Yang 1997
% (3) Isobaric
% (4) Isothermal
% (5) Model A - free of all reistances
% (6) Model B - adding Linear driving force model for mass transfer kinetic
% (7) Model C - adding Mass axial dispersion in gas phase
% (7) Absence of solid phase thermal axial conduction
% (8) Absence of radial distribution of mass and heat
% (9) Absence of gas mal-distribution or particle agglomeration
%..........................................................................
% 3 ordinary differential equations : y1, n1, n2
% 1 direct calculation for          : rhogbar (ideal gas law)
% 1 algebraic equation for          : Q (from overall mass balance)
% 1 calculated for                  : us (= Q/rhog) - but reduce its use
% Solve for y1, n1, n2, P, rhog, Q, us
%..........................................................................
function yt=psa_1_sequence_model_ABC(t,Y)
%..........................................................................
% Parameters shared with the ODE routine
%..........................................................................
global  Mg1 Mg2 adsorptionORpurge dYbardZbarDifferentialMethod rhog0 T0....
    ncall n dzbar dtbar miu epsilon epsilonbar k1 k2 q1 q2 Cs Cg rhob R....
    dp L tfeed y1feed y2feed Qin Qfeed P0 Pout Tfeed Pmax Pmin Tmax Tmin...
    Tref Q0 kg MassAxial HeatAxial HeateqmORnoneqm Dem4 Dem1
%..........................................................................
% One vector to three vectors
```

```
%..........................................................................
for i = 1:n
    Y1bar(i)    = Y(0*n+i);
    n1bar(i)    = Y(1*n+i);
    n2bar(i)    = Y(2*n+i);
end

for i = 1:n
    y1bar(i)    = Y1bar(i)/(rhog0*L/(Qfeed*tfeed));
    Mg(i)       = Mg1*y1bar(i) + Mg2*(1-y1bar(i));
end
%..........................................................................
% Boundary conditions after initial conditions
%..........................................................................
if (ncall~=0)       % not initial condition

if adsorptionORpurge == 111
   i = 1;           % Only use boundary conditions at the adsorption inlet
end
if adsorptionORpurge == 222
   i = n;           % Only use boundary conditions at the purge inlet
end

y1bar(i)         = y1feed;
Y1bar(i)         = y1feed*(rhog0*L/(Qfeed*tfeed));
end
%..........................................................................
% Formulate pre-parameters for ODEs
%..........................................................................
% Isobaric, Isothermal
%..........................................................................
for i = 1:n
    Pbar(i) = P0/(Pmax-Pmin);
    thetabar(i) = (T0-Tref)/(Tmax-Tmin);
    rhogbar(i) = rhog0*L/(Qfeed*tfeed);
end
%..........................................................................
% Formulate n1eqmbar(i), n2eqmbar(i), Qbar(i), usbar(i) with initial and
% boundary conditions
%..........................................................................
for i = 1:n
      n1eqmbar(i) = (k1*rhob*L/Qfeed)*1.11*10^(-3)*exp(1593.0/((Tmax-......
          Tmin)*thetabar(i)+Tref))*(y1bar(i))   *((Pmax-Pmin)*Pbar(i)/......
          1013250)/(1 + 1.03*10^(-4)*exp(2061.9/((Tmax-Tmin)*thetabar(i)...
          +Tref))*(y1bar(i))*((Pmax-Pmin)*Pbar(i)/1013250) + 2.07*.........
          10^(-4)*exp(2455.5/((Tmax-Tmin)*thetabar(i)+Tref))*(1-..........
          y1bar(i))*((Pmax-Pmin)*Pbar(i)/1013250));

      n2eqmbar(i) = (k2*rhob*L/Qfeed)*1.25*10^(-3)*exp(2168.6/((Tmax-......
          Tmin)*thetabar(i)+Tref))*(1-y1bar(i))*((Pmax-Pmin)*Pbar(i)/......
          1013250)/(1 + 1.03*10^(-4)*exp(2061.9/((Tmax-Tmin)*thetabar(i)...
          +Tref))*(y1bar(i))*((Pmax-Pmin)*Pbar(i)/1013250) + 2.07*.........
          10^(-4)*exp(2455.5/((Tmax-Tmin)*thetabar(i)+Tref))*(1-..........
          y1bar(i))*((Pmax-Pmin)*Pbar(i)/1013250));
end

if (ncall==0)
    for i = 1:n
        Qbar(i) = Q0/Qfeed;
        usbar(i) = Qbar(i)/rhogbar(i);
    end
end

if (ncall~=0)
    if adsorptionORpurge == 111
    for i = 1
        Qbar(i) = Qin/Qfeed;
        usbar(i) = Qbar(i)/rhogbar(i);
    end
    for i = 2:n
        Qbar(i) = -((n1eqmbar(i)-n1bar(i)) + (n2eqmbar(i)-n2bar(i)))*......
            dzbar + Qbar(i-1);
        usbar(i) = Qbar(i)/rhogbar(i);
    end
    end

    if adsorptionORpurge == 222
    for i = n
        Qbar(i) = Qin/Qfeed;
        usbar(i) = Qbar(i)/rhogbar(i);
    end
    for i = n-1:-1:1
        Qbar(i) = +((n1eqmbar(i)-n1bar(i)) + (n2eqmbar(i)-n2bar(i)))*......
            dzbar + Qbar(i+1);
        usbar(i) = Qbar(i)/rhogbar(i);
    end
    end
end
%..........................................................................
% Formulate gas phase axial dispersion coefficient in mass balance,
% DL, cm2/sec
%..........................................................................
if MassAxial == 881
for i = 1:n
    DL(i) = 0;
end
end

if MassAxial == 882
if adsorptionORpurge == 111
    DM = Dem4;
end
if adsorptionORpurge == 222
    DM = Dem1;
end
for i = 1:n
    DL(i) = 0.7*DM + 0.5*dp*(abs(Qbar(i))*Qfeed)/(rhogbar(i)*(Qfeed*.......
        tfeed/L)*epsilonbar);
end
end
%..........................................................................




%..........................................................................
% Start of 1st order (2 points) upwind difference
if dYbardZbarDifferentialMethod == 333
%..........................................................................
% This Flux Limiter formulation is valid for any regularly or non-regularly
% spaced grid, and for any constant or varying advection velocity
% Dullemon, Numerical Fluid Dynamics - Lecture Notes, Chapter 4
%..........................................................................
% Using linear terms y1bar, and mass flux Q
%..........................................................................
% Formulate convective terms for i = 3:n-2
%..........................................................................
for i = 3:n-2
    % Le = left edge of cell
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    % Re = right edge of cell
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;

        ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));

    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;

        ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
            y1bar(i-1));
    end

    superbeey1barLe     = 0;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     +.............
        0.5*abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...........
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;

        ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
            y1bar(i));

    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;

        ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /(y1bar(i+1).......
            - y1bar(i));
    end

    superbeey1barRe     = 0;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     +.............
        0.5*abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...........
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%..........................................................................
% Formulate convective terms for i = 1, 2, n-1, n
%..........................................................................
```

```
for i = 1
    Qbar_eps_Le = (Qbar(i) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = 1;
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = 1;
    end

    superbeey1barLe     = 0;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*.........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)      - y1bar(i));

    % Directional velocity has been taken care of in this flux formulation,
    % please do not change (Q) into abs(Q) anymore

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = 0;
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = (y1bar(i+2)      - y1bar(i+1))    /(y1bar(i+1).......
            - y1bar(i));
    end

    superbeey1barRe     = 0;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*.......
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)      - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%...........................................................................
for i = 2
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = 0;
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)      - y1bar(i))    /(y1bar(i)      -.....
            y1bar(i-1));
    end

    superbeey1barLe     = 0;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;

        ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
            y1bar(i));
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;

        ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /(y1bar(i+1).......
            - y1bar(i));
    end

    superbeey1barRe     = 0;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%..........................................................................
for i = n-1
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
            y1bar(i-1));
    end

    superbeey1barLe     = 0;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
            y1bar(i));
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = 0;
    end

    superbeey1barRe     = 0;
    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*.......
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%...........................................................................
for i = n
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;

        ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = 0;
    end

    superbeey1barLe     = 0;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*.......
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = 1;
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = 1;
    end

    superbeey1barRe     = 0;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i))     + 0.5*.........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i)     - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%...........................................................................
end % end of dYbardZbarDifferentialMethod == 333
%...........................................................................
%...........................................................................
% Start of 2nd order (3 points) upwind difference
if dYbardZbarDifferentialMethod == 444
%...........................................................................
% This Flux Limiter formula is valid for any regularly or non-regularly
% spaced grid, and for any constant or varying advection velocity
% Dullemon, Numerical Fluid Dynamics - Lecture Notes, Chapter 4
%...........................................................................
```

```
% Formulate convective terms for i = 3:n-2
%..........................................................................
for i = 3:n-2

    % Le = left edge of cell
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    % Re = right edge of cell
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe      = (y1bar(i-1)      - y1bar(i-2))    /(y1bar(i)      -...
            y1bar(i-1));
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe      = (y1bar(i+1)      - y1bar(i))    /(y1bar(i)      -.....
            y1bar(i-1));
    end

    superbeey1barLe     = 1;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)      - y1bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe      = (y1bar(i)      - y1bar(i-1))    /(y1bar(i+1)      -...
            y1bar(i));
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe      = (y1bar(i+2)      - y1bar(i+1))    /..................
            (y1bar(i+1)      - y1bar(i));
    end

    superbeey1barRe     = 1;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)      - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%..........................................................................
% Formulate convective terms for i = 1, 2, n-1, n
%..........................................................................
for i = 1
    Qbar_eps_Le = (Qbar(i) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe      = 1;
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = 1;
    end

    superbeey1barLe     = 1;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*.........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = 0;
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /(y1bar(i+1).......
            - y1bar(i));
    end

    superbeey1barRe     = 1;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*.......
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%.........................................................................
for i = 2
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = 0;
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
            y1bar(i-1));
    end

    superbeey1barLe     = 1;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*.......
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
            y1bar(i));

    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /..................
```

```
            (y1bar(i+1)     - y1bar(i));
    end

    superbeey1barRe     = 1;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%...........................................................................
for i = n-1
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));

    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
            y1bar(i-1));
    end

    superbeey1barLe     = 1;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
            y1bar(i));

    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = 0;
    end

    superbeey1barRe     = 1;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%...........................................................................
for i = n
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i))/(2*epsilon);
    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = 0;
    end

    superbeey1barLe     = 1;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*.......
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = 1;
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = 1;
    end

    superbeey1barRe     = 1;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i))     + 0.5*..........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i)     - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%..........................................................................
end % end of dYbardZbarDifferentialMethod == 444
%..........................................................................
%..........................................................................
% Start of Superbee Flux Limiter
if dYbardZbarDifferentialMethod == 777
%..........................................................................
% This Flux Limiter formula is valid for any regularly or non-regularly
% spaced grid, and for any constant or varying advection velocity
% Dullemon, Numerical Fluid Dynamics - Lecture Notes, Chapter 4
%..........................................................................
% Formulate convective terms for i = 3:n-2
%..........................................................................
for i = 3:n-2
    % Le = left edge of cell
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    % Re = right edge of cell
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
            y1bar(i-1));
    end

    superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,     .............
        min(2,ry1barLe)));

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
            y1bar(i));
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /..................
            (y1bar(i+1)     - y1bar(i));
    end

    superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,     .............
        min(2,ry1barRe)));

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%..........................................................................
% Formulate convective terms for i = 1, 2, n-1, n
%..........................................................................
for i = 1
    Qbar_eps_Le = (Qbar(i) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = 1;
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = 1;
    end

    superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,     .............
        min(2,ry1barLe)));

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*..........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = 0;
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /..................
            (y1bar(i+1)     - y1bar(i));
    end

    superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,     .............
        min(2,ry1barRe)));

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%..........................................................................
for i = 2
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = 0;
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
            y1bar(i-1));
    end

    superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,     .............
        min(2,ry1barLe)));

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
            y1bar(i));

    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /(y1bar(i+1)    ...
            - y1bar(i));
    end

    superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,     .............
        min(2,ry1barRe)));

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%...........................................................................
for i = n-1
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
            y1bar(i-1));
    end

    superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,    .............
        min(2,ry1barLe)));

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
            y1bar(i));
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = 0;
    end

    superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,    .............
        min(2,ry1barRe)));

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%...........................................................................
for i = n
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = 0;
    end

    superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,     .............
        min(2,ry1barLe)));

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*.......
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = 1;
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = 1;
    end

    superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,     .............
        min(2,ry1barRe)));

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i))     + 0.5*..........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i)     - y1bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
end
%..........................................................................
end % end of dYbardZbarDifferentialMethod == 777
%..........................................................................
%..........................................................................
% Formulate ODEs - can be used for adsorption and purge - both directions
%..........................................................................
for i=1
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon + ....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y1bar(i+2)-2*Y1bar(i+1)+...
        Y1bar(i))/(dzbar)^2;
end
for i=2:n-1
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon + ....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y1bar(i+1)-2*Y1bar(i)+.....
        Y1bar(i-1))/(dzbar)^2;
end
for i=n
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon + ....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y1bar(n)-2*Y1bar(n-1)+.....
        Y1bar(n-2))/(dzbar)^2;
end
%..........................................................................
if MassAxial == 882 % Turn on mass axial dispersion
if adsorptionORpurge == 111
    % 2nd BC for d2YdZ2 at zbar = 1 : dY/dz = 0 (wrong); just eliminate
    % d2YdZ2 from the complete equation (Schiesser)
    for i=n
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon + 0;
    end
end
```

```
if adsorptionORpurge == 222
    % 2nd BC for d2YdZ2 at zbar = 1 : dY/dz = 0 (wrong); just eliminate
    % d2YdZ2 from the complete equation (Schiesser)
    for i=1
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon + 0;
    end
end
end
%.........................................................................
for i=1:n
      dn1bardtbar(i) = k1*tfeed*(n1eqmbar(i)-n1bar(i));
      dn2bardtbar(i) = k2*tfeed*(n2eqmbar(i)-n2bar(i));
end
%.........................................................................
% Three vectors into one vector
%.........................................................................
for i = 1:n
    Yt(0*n+i) = dY1bardtbar(i);
    Yt(1*n+i) = dn1bardtbar(i);
    Yt(2*n+i) = dn2bardtbar(i);
end

if adsorptionORpurge == 111
    i = 1;
    Yt(0*n+i) = 0;      % because Y1bar(1) = y1feed*rhog(1)
    %Yt(3*n+i) = 0;     % because Thetabarg(1) = Tfeed*rhog(1)
    %Yt(4*n+i) = 0;     % because Thetabars(1) = Tfeed
    %Yt(5*n+i) = 0;     % because Y2bar(1) = y2feed*rhog(1)
end
if adsorptionORpurge == 222
    i = n;
    Yt(0*n+i) = 0;     % because Y1bar(n) = y1feed*rhog(n)
    %Yt(3*n+i) = 0;     % because Thetabarg(n) = Tfeed*rhog(n)
    %Yt(4*n+i) = 0;     % because Thetabars(n) = Tfeed
    %Yt(5*n+i) = 0;     % because Y2bar(n) = y2feed*rhog(n)
end
%.........................................................................
  yt=Yt';

% Increment calls to psa_1
  ncall=ncall+1;
%.........................................................................
```

Appendix 4

Model D

- **Model D**, an adsorption (`adsorptionORpurge = 111`) or desorption (`adsorptionORpurge = 222`) step that includes **gas-solid adsorption kinetic, gas phase mass axial dispersion**, **pressure drop**, is modeled in this routine by setting (`%k1 = 50000, %k2 = 50000`), (`MassAxial = 882`).

- One setting that especially needs your attention is (`nout, n`). This combination must be set properly to achieve numerical stability and convergence.

- Only one spatial differential method is available in this routine (`dYbardZbarDifferentialMethod = 777`).

Main Routine

```
%..........................................................................
% File: psa_main_sequence_model_D
%..........................................................................
% Chai Siew Wah, Chemical Engineering, Lehigh University, May 2011
%..........................................................................
% MATHEMATICAL MODEL OF PRESSURE SWING ADSORPTION SYSTEM FOR ONE STEP :
% ADSORPTION OR PURGE
%..........................................................................
%
%                  :......................................:
%   ADSORPTION --> :                                      : --> PRODUCT
%   ADSORPTION --> :                                      : --> PRODUCT
%   ADSORPTION --> :                                      : --> PRODUCT
%                  :......................................:
%                 i=1                                    i=n
%
%                  :......................................:
%     EXHAUST <--  :                                      : <-- PURGE
%     EXHAUST <--  :                                      : <-- PURGE
%     EXHAUST <--  :                                      : <-- PURGE
%                  :......................................:
%                 i=1                                    i=n
%..........................................................................
% Assumptions
%..........................................................................
% (1) Ideal gas law
% (2) Empirical Langmuirian type binary isotherms - from Rege and Yang 1997
% (3) Nonisobaric - Ergun equation solved analytically
% (4) Isothermal
% (5) Linear driving force model for mass transfer kinetic
% (6) Mass axial dispersion in gas phase
% (7) Absence of solid phase thermal axial conduction
% (8) Absence of radial distribution of mass and heat
% (9) Absence of gas mal-distribution or particle agglomeration
%..........................................................................
% 4 ordinary differential equations : y1, y2, n1, n2
% 1 direct calculation for          : rhogbar (= Y1bar + Y2bar)
% 1 algebral equation for           : Q  (= Ergun equation)
% 1 algebral equation for           : P  (= Ideal gas law)
% 1 calculated for                  : us  (= Q/rhog) - but reduce its use
% Solve for y1, y2, n1, n2, P, rhog, Q, us
%..........................................................................
% (1) Solve Ergun equation for Qbar and properly dealing its positive or
% negative flow direction

% (2) Use Schiesser's boundary condition for mass and heat balances that
% have axial dispersion terms (retain the whole equation except the 2nd
% derivative of space) instead of Danwerts' boundary condition of dY/dz = 0
% at exit

% (3) Eliminate overall mass balance, use 2nd component mass balance, so
% that I do not need a boundary condition for rhog (there is no defined
% rhog at inlet or exit), and I can use flux conservation formulation of
% Q throughout, and rhogbar = Y1bar + Y2bar.

% (4) Gas compositions y1feed y2feed are specified, rhogbar(feed end) =
% Y1bar + Y2bar is not. But I need the right BCs for Y1bar, Y2bar,
% Thetabarg, which depend on rhogbar(feed end).

% (5) Normally Pbar(i) is calculated from rhogbar(i) using ideal gas law.
% But at feed end, rhogbar(feed end) is undetermined. So, Pressure comes
% in to help by using its previous dP/dz and Pbar to calculate
% Pbar(feed end), then calculate rhogbar(feed end) from ideal gas law,
% then calculate the right BCs for Y1bar, Y2bar, Thetabarg.

% (6) Correct use of: epsilon (gas phase accumulation), epsilonbar(Ergun,
% axial dispersions), mass axial dispersion, from literature values (Rege
% and Yang 1997 provides Cs, Cg, q1, q2, isotherms), adsorbent density from
bed
% dimensions and 160 g LiX in Invacare test unit.

% (7) Only allow average velocity,uz = us/epsilon, and mass flux, Q, to
% acquire positive or negative value, all the rest, y1, y2, n1, n2, Tg, Ts,
% P, rhog, must use positive values
%..........................................................................
% Clear previous files
clc
clear all
format short
%..........................................................................
% Parameters shared with the ODE routine
%..........................................................................
global  Mg1 Mg2 adsorptionORpurge dYbardZbarDifferentialMethod T0......
    ncall n dzbar dtbar miu epsilon epsilonbar k1 k2 q1 q2 Cs Cg rhob R....
    dp L tfeed y1feed y2feed Qin Qfeed P0 Pout Tfeed Pmax Pmin Tmax Tmin...
    Tref Q0 kg MassAxial HeatAxial HeateqmORnoneqm Dem4 Dem1
%..........................................................................
% Select process step
%..........................................................................
adsorptionORpurge = 222;       %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% adsorptionORpurge = 111;     % 111 for adsorption
% adsorptionORpurge = 222;     % 222 for purge
%..........................................................................
% Assign positive or negative sign to feed gas mass flux (Q) and average
% velocity (uz = us/epsilon) at feed and initial condition (which are
% within our control). Do not assign +ve / -ve to Qbar(i) and usbar(i),
% which are calculated by the program
```

```
%..........................................................................
if adsorptionORpurge == 111
    FE = +1;    % for mass flux at feed end i = 1
end
if adsorptionORpurge == 222
    FE = -1;    % for mass flux at feed end i = n
end
%..........................................................................
% Select differential method for dYbardZbar
%..........................................................................
dYbardZbarDifferentialMethod = 777;       %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% dYbardZbarDifferentialMethod = 777; % 2nd order TVD Superbee Flux Limiter
%..........................................................................
% Select Mass and Heat Axial Dispersions
%..........................................................................
MassAxial = 882;       %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% MassAxial = 881;     % Shut off mass axial dispersion
% MassAxial = 882;     % Turn on mass axial dispersion, need 2nd BC
%..........................................................................
% Select type of plots
%..........................................................................
choose = 4;  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% choose = 1 : 2D subplots of y1, y2, n1, n2, T, P
% choose = 2 : Individual 2D plots of y1, y2, n1, n2, T, P, rhog, Q, us
% choose = 3 : Surface plots of y1, y2, n1, n2, T, P, rhog, Q, us
% choose = 4 : The 2D plots that Dr. Sircar is interested in
%..........................................................................
% Define number of nodes along t and along z
%..........................................................................
nout = 1001;        % nodes along t
n = 151;            % nodes along z
%..........................................................................
% Define dimensionless z and dz
%..........................................................................
zbarl = 0.0;                    % zbar lower limit, zbar = 0/L
zbaru = 1.0;                    % zbar upper limit, zbar = L/L
dzbar = (zbaru-zbarl)/(n-1);    % zbar differential space
zbar  = linspace(zbarl,zbaru,n);
%..........................................................................
% Define dimensionless t and dt
% Independent variable, tbar, for ODE integration
%..........................................................................
tbarl = 0.0;                    % tbar lower limit, tbar = 0/tfeed
tbaru = 1.0;                    % tbar upper limit, tbar = tfeed/tfeed
dtbar = (tbaru-tbarl)/(nout-1); % tbar differential space
tbar  = linspace(tbarl,tbaru,nout);
tbarv = tbar';
%..........................................................................
% Gas properties
%..........................................................................
R = 8.314472*10^4;      % gas constant, g.cm2/K/mmol/sec2
Cg = 0.00687;           % gas phase heat capacity, cal/mmol/K, Rege&Yang 97
%kg = 6.19e-5;          % gas phase (N2 at 1 atm 300K) thermal conductivity
                        % , cal/cm/sec/K
%Cg = 0.00721;          % gas phase (Air at 300K) heat capacity, cal/mmol/K
kg = 6.44e-5;           % gas phase (Air at 1 atm 295K) thermal
                        % conductivity, cal/cm/sec/K
miu = 18.2385e-5;       % air dynamic viscosity at 25C and 1 atm, g/cm/sec
Mg1 = 0.032;            % O2 molecular weight, g/mmol
Mg2 = 0.028;            % N2 molecular weight, g/mmol
```

```
%..........................................................................
% Packed bed dimensions
%..........................................................................
% Conventional adsorber
L = 8.382;              % bed length, cm (3.3 in = 8.382 cm)
D = 6.0325;             % bed diameter, cm (2 3/8 in)

% Pancake adsorber
%L = 2.12;              % bed length, cm (3.3 in = 8.382 cm)
%D = 12;                % bed diameter, cm (2 3/8 in)

A = pi*(D/2)^2;         % bed cross-sectional area, cm2
rhob = 0.668;           % bulk density, g/cm3, according to 160 g LiX
%rhob = 0.632;          % bulk density, g/cm3, at 1/3 lb (151.33 g) LiX
%..........................................................................
% Adsorbent properties
%..........................................................................
dp = 0.0200;            % particle diameter, cm   %%%%%%%%%%%%%%%%%%%%%%%%%
epsilon = 0.61;         % total column (helium) void fraction= ep x (1-e)+e
epsilonp = 0.35;        % internal/intraparticle void fraction
epsilonbar = 0.40;      % external/interparticle void fraction, bed voidage
                        % , Rege and Yang 1997
taup = 3;               % particle tortuosity factor
Cs = 0.28;              % adsorbent (solid phase) heat capacity, cal/g/K,
                        % Rege and Yang 1997
ks = 4.78e-4;           % dry sand thermal conductivity, 298K, cal/cm/sec/K
%..........................................................................
% Mass transfer kinetic and heat properties
%..........................................................................
Dek1 = 0.0307;          % O2 effective knudsen diffusivity in Zeochem LiLSX
                        % at 1-4 atm, cm2/sec, Todd and Webley 2006
Dek2 = 0.03277;         % N2 effective knudsen diffusivity in Zeochem LiLSX
                        % at 1-4 atm, cm2/sec, Todd and Webley 2006
Dem1 = 0.0348;          % effective molecular diffusivity in Zeochem LiLSX
                        % at 1 atm, cm2/sec, Todd and Webley 2006
Dem4 = 0.008707;        % effective molecular diffusivity in Zeochem LiLSX
                        % at 4 atm, cm2/sec, Todd and Webley 2006

if adsorptionORpurge == 111
    De1 = (epsilonp/taup)*(1/(1/Dek1 + 1/Dem4));
    De2 = (epsilonp/taup)*(1/(1/Dek2 + 1/Dem4));
end
if adsorptionORpurge == 222
    De1 = (epsilonp/taup)*(1/(1/Dek1 + 1/Dem1));
    De2 = (epsilonp/taup)*(1/(1/Dek2 + 1/Dem1));
end
k1 = 15*De1/(dp/2)^2;   % O2 overall mass transfer coefficient, sec-1
k2 = 15*De2/(dp/2)^2;   % N2 overall mass transfer coefficient, sec-1
q1 = 3.16;              % isosteric heat of adsorption of O2 on LiX, cal/
                        % mmol, Rege and Yang 1997
q2 = 5.60;              % isosteric heat of adsorption of N2 on LiX, cal/
                        % mmol, Rege and Yang 1997
%k1 = 500;              % equilibrium O2 overall mass transfer rate, sec-1
%k2 = 500;              % equilibrium N2 overall mass transfer rate, sec-1
%..........................................................................
% Feed conditions at i = 1 (adsorption)
%                 at i = n (purge)
%..........................................................................
if adsorptionORpurge == 111
    y1feed = 0.21;      % feed gas composition, 21% O2 + 79% N2
    y2feed = 0.79;      % feed gas composition, 21% O2 + 79% N2
    Tfeed = 298;        % feed temperature, K
    tfeed = 2;          % feed step time, sec
    qfeed = 300/tfeed;  % feed mass flow rate, mmol/sec
    Pout = 4.0*1013250; % exit pressure at i=n during adsorption, g/cm/sec2
end
if adsorptionORpurge == 222
    y1feed = 1.00;      % purge gas composition, 100% O2 + 0% N2
    y2feed = 0.00;      % purge gas composition, 100% O2 + 0% N2
    Tfeed = 298;        % purge temperature, K
    tfeed = 2;          % purge step time, sec
    qfeed = 306/tfeed;  % purge mass flow rate, mmol/sec
    Pout = 1.0*1013250; % outlet pressure at i=1 during purge, g/cm/sec2
end
Qfeed = qfeed/A;        % mass flux, mmol/cm2/sec
Qin = FE * Qfeed;       % assign directions for the absolute value of Qfeed
% usfeed = Qfeed/rhog;  % superficial velocity, cm/sec
% I should not specify Pfeed because I need to find out the pressure drop
% starting from Pout using Ergun equation in momentum balance.
% Feed gas density is unknown because rhogfeed = Pfeed/(R*Tfeed) mmol/cm3
% Equilibrium capacity at feed inlet (i=1) or (i=n) is unknown bcoz Pfeed
%..........................................................................
% Bed initial conditions
%..........................................................................
if adsorptionORpurge == 111
    y10 = 1.00;         % saturated with 100% O2 + 0% N2
    y20 = 0.00;         % saturated with 100% O2 + 0% N2
    P0 = 3.0*1013250;   % pressure after pressurization step, g/cm/sec2
    T0 = 298;           % temperature after pressurization step, K
end
if adsorptionORpurge == 222
    y10 = 0.21;         % saturated with 21% O2 + 79% N2
    y20 = 0.79;         % saturated with 21% O2 + 79% N2
    P0 = 1.0*1013250;   % pressure after depressurization step, g/cm/sec2
    T0 = 298;           % temperature after depressurization step, K
end

Q0 = FE * 0;            % without gas flux, mmol/cm2/sec
rhog0 = P0/(R*T0);      % gas phase density, mmol/cm3
us0 = Q0/rhog0;         % superficial velocity, cm/sec
n1eqm0 = 1.11*10^(-3)*exp(1593.0/T0)*(y10)   *(P0/1013250)/(1 + 1.03*10^....
    (-4)*exp(2061.9/T0)*(y10)*(P0/1013250) + 2.07*10^(-4)*exp(2455.5/T0)...
    *(1-y10)*(P0/1013250));
n2eqm0 = 1.25*10^(-3)*exp(2168.6/T0)*(1-y10)*(P0/1013250)/(1 + 1.03*10^....
    (-4)*exp(2061.9/T0)*(y10)*(P0/1013250) + 2.07*10^(-4)*exp(2455.5/T0)...
    *(1-y10)*(P0/1013250));
n10 = n1eqm0;           % O2 adsorbed, mmol O2/g
n20 = n2eqm0;           % N2 adsorbed, mmol N2/g
% Langmuir isotherms of O2 and N2 on LiX from Salil U. Rege and Ralph T.
% Yang, Ind. Eng. Chem. Res. 1997, 36, 5358-5365.
Mg0 = Mg1*y10 + Mg2*(1-y10); % average gas phase molecular weight, g/mmol
%..........................................................................
% Important note: variables must be parameterized within the minimum and
% maximum range in order to avoid singular matrix of A.x = b
%..........................................................................
% Additional parameters required for proper nondimensionalization of
% dependent variables
%..........................................................................
Pmax = Pout;            % maximum pressure, g/cm/sec2
Pmin = 0;               % minimum pressure, g/cm/sec2
```

```
Tref = T0;
Tmax = Tfeed;           % maximum temperature, K
Tmin = 273;             % minimum temperature, K
%..........................................................................
% Nonlinear variables
%..........................................................................
% Y1bar = y1bar * rhogbar
% Y2bar = y2bar * rhogbar
%..........................................................................
% Initial conditions for y1bar, n1bar, n2bar, y2bar for i = 1:n
%..........................................................................
% Note that boundary conditions at i = 1 (adsorption) or i = n (purge)
% after t=0 should not be brought in here because BC should be dealt with
% in the Method of Lines routine
%..........................................................................
for i=1:n
    Y0(0*n+i) = y10*rhog0*L/(Qfeed*tfeed);      % for Y1bar
    Y0(1*n+i) = (k1*rhob*L/Qfeed)*n10;          % for n1bar
    Y0(2*n+i) = (k2*rhob*L/Qfeed)*n20;          % for n2bar
    %Y0(3*n+i) = ((T0-Tref)/(Tmax-Tmin))*rhog0*L/(Qfeed*tfeed); % Thetabarg
    %Y0(4*n+i) = ((T0-Tref)/(Tmax-Tmin));                       % Thetabars
    Y0(3*n+i) = y20*rhog0*L/(Qfeed*tfeed);      % for Y2bar
end
%..........................................................................
% ODE integration
%..........................................................................
ncall = 0;
reltol = 1.0e-04;
abstol = 1.0e-04;
options = odeset('RelTol',reltol,'AbsTol',abstol);

[t,Y] = ode15s(@psa_1_sequence_model_D, tbar, Y0, options);
%..........................................................................
% One vector to four vectors
%..........................................................................
for it = 1:nout     % time
for i = 1:n         % position
    Y1bar(it,i)     = Y(it,0*n+i);
    n1bar(it,i)     = Y(it,1*n+i);
    n2bar(it,i)     = Y(it,2*n+i);
    %Thetabarg(it,i)= Y(it,3*n+i);
    %Thetabars(it,i)= Y(it,4*n+i);
    Y2bar(it,i)     = Y(it,3*n+i);
end
end

for it = 1:nout
for i = 1:n
    y1bar(it,i)    = Y1bar(it,i)/(Y1bar(it,i)+Y2bar(it,i));
    y2bar(it,i)    = Y2bar(it,i)/(Y1bar(it,i)+Y2bar(it,i));
    rhogbar(it,i)  = Y1bar(it,i) + Y2bar(it,i);
    Mg(it,i)       = Mg1*y1bar(it,i) + Mg2*(1-y1bar(it,i));
    %thetabarg(it,i)= Thetabarg(it,i)/rhogbar(it,i);
    %thetabars(it,i)= Thetabars(it,i);
end
end
%..........................................................................
% Fix the boundary conditions
%..........................................................................
```

```
% ODE solver only solves equations of dY/dt (without dY/dz in it), boundary
% conditions are not cared for. Though boundary conditions were forced in
% MOL routine, ODE solver only stores the solutions of Y calculated from
% governing ODEs and previous Y, boundary conditions at i=1 (adsorption) or
% i=n (purge) are not stored in ODE solver.
%..........................................................................
% ODE solver does take care of initial conditions at it = 1
for it = 2:nout     % time

if adsorptionORpurge == 111
   i = 1;
end
if adsorptionORpurge == 222
   i = n;
end

y1bar(it,i)         = y1feed;
y2bar(it,i)         = y2feed;
Y1bar(it,i)         = y1feed*rhogbar(it,i);
Y2bar(it,i)         = y2feed*rhogbar(it,i);
%thetabarg(it,i)     = (Tfeed-Tref)/(Tmax-Tmin);
%Thetabarg(it,i)     = ((Tfeed-Tref)/(Tmax-Tmin))*rhogbar(it,i);
%thetabars(it,i)     = (Tfeed-Tref)/(Tmax-Tmin);
%Thetabars(it,i)     = (Tfeed-Tref)/(Tmax-Tmin);
end
%..........................................................................
% Isothermal
%..........................................................................
for it = 1:nout
    for i=1:n
        thetabar(it,i) = (T0-Tref)/(Tmax-Tmin);
    end
end
%..........................................................................
% Formulate Pbar(i) using ideal gas law except that at inlet point
%..........................................................................
for it = 1              % initial condition
    for i=1:n           % position
        Pbar(it,i) = P0/(Pmax-Pmin);
    end
end

for it = 2:nout         % time
%..........................................................................
    if adsorptionORpurge == 111
        i = n;          % BC for adsorption step
        Pbar(it,i) = Pout/(Pmax-Pmin);

        for i = n-1:-1:2
            Pbar(it,i) = rhogbar(it,i)*(thetabar(it,i)*(Tmax-Tmin)+.......
                Tref)*(R*Qfeed*tfeed)/((Pmax-Pmin)*L);
            dPbardzbar(it,i) = (Pbar(it,i+1)-Pbar(it,i))/dzbar;
        end

        for i = 1
            Pbar(it,i)       = Pbar(it,i+1) - dPbardzbar(it,i+1)*dzbar;
            rhogbar(it,i)    = Pbar(it,i)*(Pmax-Pmin)*L/((.................
                thetabar(it,i)*(Tmax-Tmin)+Tref)*R*Qfeed*tfeed);
            dPbardzbar(it,i) = (Pbar(it,i+1)-Pbar(it,i))/dzbar;
            Y1bar(it,i)      = y1feed*rhogbar(it,i);
            Y2bar(it,i)      = y2feed*rhogbar(it,i);
        end

        for i = n           % special upwinding
            dPbardzbar(it,i) = (Pbar(it,i)-Pbar(it,i-1))/dzbar;
        end
    end
%..........................................................................
    if adsorptionORpurge == 222
        i = 1;          % BC for purge step
        Pbar(it,i) = Pout/(Pmax-Pmin);

        for i = 2:n-1
            Pbar(it,i) = rhogbar(it,i)*(thetabar(it,i)*(Tmax-Tmin)+.......
                Tref)*(R*Qfeed*tfeed)/((Pmax-Pmin)*L);
            dPbardzbar(it,i) = (Pbar(it,i)-Pbar(it,i-1))/dzbar;
        end

        for i = n
            Pbar(it,i)       = dPbardzbar(it,i-1)*dzbar + Pbar(it,i-1);
            rhogbar(it,i)    = Pbar(it,i)*(Pmax-Pmin)*L/((.................
                thetabar(it,i)*(Tmax-Tmin)+Tref)*R*Qfeed*tfeed);
            dPbardzbar(it,i) = (Pbar(it,i)-Pbar(it,i-1))/dzbar;
            Y1bar(it,i)      = y1feed*rhogbar(it,i);
            Y2bar(it,i)      = y2feed*rhogbar(it,i);
        end

        for i = 1           % special upwinding
            dPbardzbar(it,i) = (Pbar(it,i+1)-Pbar(it,i))/dzbar;
        end
    end
%..........................................................................
end
%..........................................................................
% Solving for usbar(i) in Ergun equationa using quadratic formula - failed
% Solving for Qbar(i) in Ergun equationa using quadratic formula
%..........................................................................
for it = 1              % initial condition
    for i = 1:n
        Qbar(it,i) = Q0/Qfeed;
    end
end

for it = 2:nout         % time
    if adsorptionORpurge == 111
        Qbar(it,1) = Qin/Qfeed;
        for i = 2:n
            a = 1.75*Mg(it,i)*L*(1-epsilonbar)/(dp*rhogbar(it,i)*..........
                epsilonbar^2);
            b = 150*miu*L*(1-epsilonbar)^2/((dp^2)*Qfeed*rhogbar(it,i)*....
                epsilonbar^2);
            c = epsilonbar*tfeed*(Pmax-Pmin)*dPbardzbar(it,i)/(Qfeed*L);
            solQbar1 = (-b + sqrt(b^2-4*a*c))/(2*a);
            solQbar2 = (-b - sqrt(b^2-4*a*c))/(2*a);
            Qbar(it,i) = max(solQbar1,solQbar2);
        end
    end

    if adsorptionORpurge == 222
        Qbar(it,n) = Qin/Qfeed;
        for i = 1:n-1
            a = -1.75*Mg(it,i)*L*(1-epsilonbar)/(dp*rhogbar(it,i)*.........
                epsilonbar^2);
            b = 150*miu*L*(1-epsilonbar)^2/((dp^2)*Qfeed*rhogbar(it,i)*....
                epsilonbar^2);
            c = epsilonbar*tfeed*(Pmax-Pmin)*dPbardzbar(it,i)/(Qfeed*L);
            solQbar1 = (-b + sqrt(b^2-4*a*c))/(2*a);
            solQbar2 = (-b - sqrt(b^2-4*a*c))/(2*a);
            Qbar(it,i) = min(solQbar1,solQbar2);
        end
    end
end
%..........................................................................
% Formulate usbar(i) = Qbar(i)/rhogbar(i)
%..........................................................................
for it = 1              % initial condition
    for i=1:n           % position
        usbar(it,i) = Q0/Qfeed/rhogbar(it,i);
    end
end

for it = 2:nout         % time
    if adsorptionORpurge == 111
        i = 1;          % BC for adsorption step
        usbar(it,i) = Qin/Qfeed/rhogbar(it,i);

        for i = 2:n
            usbar(it,i) = Qbar(it,i)/rhogbar(it,i);
        end
    end

    if adsorptionORpurge == 222
        i = n;          % BC for purge step
        usbar(it,i) = Qin/Qfeed/rhogbar(it,i);

        for i = n-1:-1:1
            usbar(it,i) = Qbar(it,i)/rhogbar(it,i);
        end
    end
end
%..........................................................................
% Mass flux
%..........................................................................
AccQ = 0;
for i = 1:n
        AccQ = AccQ + Qfeed*abs(Qbar(nout,i));
end
AveQ_nout_mmolecm2sec = AccQ/n
AveQ_nout_lbmoleft2hr = AveQ_nout_mmolecm2sec*3600*929.0304/454000
%..........................................................................
% Formulate gas phase axial dispersion coefficient in mass balance,
% DL, cm2/sec
%..........................................................................
if MassAxial == 881
for i = 1:n
    DL(nout,i) = 0;
end
end

if MassAxial == 882
if adsorptionORpurge == 111
    DM = Dem4;
end
if adsorptionORpurge == 222
    DM = Dem1;
end
for i = 1:n
    DL(nout,i) = 0.7*DM + 0.5*dp*(abs(Qbar(nout,i))*Qfeed)/(...............
        rhogbar(nout,i)*(Qfeed*tfeed/L)*epsilonbar);
end
end

AccDL = 0;
for i = 1:n
    AccDL = AccDL + DL(nout,i);
end
AveDL_nout_cm2sec = AccDL/n
%..........................................................................
% Calculate Reynolds number of particle
%..........................................................................
AccReynolds = 0;
for i=1:n
    Reynolds(i) = (Qfeed*abs(Qbar(nout,i))*dp*Mg(nout,i))/miu;
    AccReynolds = AccReynolds + Reynolds(i);
end
AveReynolds_nout = AccReynolds/n
%..........................................................................
% Calculate pressure drop across the bed
%..........................................................................
if adsorptionORpurge == 111
Pressuredropacrossbed_psi = ((Pmax-Pmin).*Pbar(nout,1) - Pout)*14.69595/...
    1013250;
end

if adsorptionORpurge == 222
Pressuredropacrossbed_psi = ((Pmax-Pmin).*Pbar(nout,n) - Pout)*14.69595/...
    1013250;
end
%..........................................................................
% Display selected output
%..........................................................................
fprintf('\n abstol = %8.1e   reltol = %8.1e\n', abstol, reltol);
fprintf('\n ncall = %4d\n',ncall);
%..........................................................................
% Display important parameters
%..........................................................................
fprintf('\n process step = %d       dYbardZbar differential method = %d',
adsorptionORpurge, dYbardZbarDifferentialMethod);

fprintf('\n P0 = %g atm          T0 = %g K       yN20 = %g       Q0 = %g
mmol/cm2/sec', P0/1013250, T0, (1-y10), Q0);
fprintf('\n tfeed = %g sec      Tfeed = %g K    yN2feed = %g    Qfeed = %g
mmol/cm2/sec    qfeed = %g mmol/sec', tfeed, Tfeed, (1-y1feed), Qfeed,
qfeed);
fprintf('\n dp = %g um         L = %g cm        D = %g cm       Pressure drop
across bed at nout = %g psi', dp*10000, L, D, Pressuredropacrossbed_psi);
fprintf('\n k1 = %g sec-1      k2 = %g sec-1    rhob = %g g/cc\n\n', k1, k2,
rhob);
%..........................................................................
% Plot numerical solutions vs. independent variables (t,z)
```

```
%..........................................................................
z0 = 0.0;
zend = L;
z = linspace(z0,zend,n);

t0 = 0.0;
tend = tfeed;
t = linspace(t0,tend,nout);
%..........................................................................
if choose == 1 % start of choose == 1 : 2D subplots of y1, y2, n1, n2, T, P
%..........................................................................
% Subplots of y1, y2, n1, n2, T, P
%..........................................................................
figure(1);

subplot(2,2,1)
plot(t,y1bar(:,(n+1)/2),'-r',t,(1-y1bar(:,(n+1)/2)),'-b'); %axis tight
h = legend('y_{O2}','y_{N2}',1);
title('y_{O2} and y_{N2} vs. time at L/2'); xlabel('time (sec)'); .........
    ylabel('y_{O2} and y_{N2}')

subplot(2,2,2)
plot(t,(Qfeed/(k1*rhob*L)).*n1bar(:,(n+1)/2),'-r',t,(Qfeed/(k2*rhob*.......
    L)).*n2bar(:,(n+1)/2),'-b'); %axis tight
h = legend('n_{O2}','n_{N2}',1);
title('n_{O2} and n_{N2} vs. time at L/2'); xlabel('time (sec)'); .........
    ylabel('n_{O2} and n_{N2} (mmol/g adsorbent)')

subplot(2,2,3)
plot(t,Tref+(Tmax-Tmin).*thetabar(:,(n+1)/2)); %axis tight
title('T vs. time at L/2'); xlabel('time (sec)'); ylabel('T (K)')

subplot(2,2,4)
plot(z,((Pmax-Pmin)/1013250).*Pbar(nout,:)); %axis tight
title('P vs. bed axial position at tfinal'); ..............................
    xlabel('Bed axial position, z (cm)'); ylabel('P (atm)')
end % end of choose == 1
%..........................................................................
if choose == 2 % start of choose == 2 : Individual 2D plots
%..........................................................................
% Gas phase mole fraction, y1 and y2, vs. time at L/2
%..........................................................................
figure(2)
plot(t,y1bar(:,(n+1)/2),'-r>',t,(1-y1bar(:,(n+1)/2)),'-bs','LineWidth',....
    2,'MarkerSize',3)
h = legend('y_{O2}','y_{N2}',1);
xlabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
ylabel('Gas phase mole fraction, y_{O2} and y_{N2}','color','k',...........
    'fontsize',10,'fontweight','b')
title('Gas phase mole fraction vs. time at L/2','color','k','fontsize',....
    12,'fontweight','b')
axis([0 tfeed 0 1])
grid on
%..........................................................................
% Amount adsorbed on adsorbent, n1 and n2, vs. time at L/2
%..........................................................................
figure(3)
plot(t,(Qfeed/(k1*rhob*L)).*n1bar(:,(n+1)/2),'-
r>',t,(Qfeed/(k2*rhob*L)).*n2bar(:,(n+1)/2),'-
bs','LineWidth',2,'MarkerSize',3)
```

```
h = legend('n_{O2}','n_{N2}',1);
xlabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
ylabel('Amount adsorbed on adsorbent, n_{O2} and n_{N2} (mmol/g
adsorbent)','color','k','fontsize',10,'fontweight','b')
title('Amount adsorbed on adsorbent vs. time at
L/2','color','k','fontsize',12,'fontweight','b')
grid on
%...........................................................................
% Temperature, T, vs. time at L/2
%...........................................................................
figure(4)
plot(t,(Tref+(Tmax-Tmin).*thetabar(:,(n+1)/2)),'LineWidth',2)
xlabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
ylabel('Temperature, T (K)','color','k','fontsize',10,'fontweight','b')
title('Temperature vs. time at
L/2','color','k','fontsize',12,'fontweight','b')
grid on
%...........................................................................
% Pressure, P, vs. bed axial position at t=tfinal
%...........................................................................
figure(5)
plot(z,((Pmax-Pmin)/1013250).*Pbar(nout,:),'LineWidth',2)
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Pressure, P (atm)','color','k','fontsize',10,'fontweight','b')
title('Pressure vs. bed axial position at
tfinal','color','k','fontsize',12,'fontweight','b')
grid on
%...........................................................................
% Gas phase concentration, rhog, vs. bed axial position at t=tfinal
%...........................................................................
figure(6)
plot(z,(Qfeed*tfeed/L).*rhogbar(nout,:),'LineWidth',2)
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Gas phase concentration, \rho_g
(mmol/cm^3)','color','k','fontsize',10,'fontweight','b')
title('Gas phase concentration vs. bed axial position at
tfinal','color','k','fontsize',12,'fontweight','b')
grid on
%...........................................................................
% Mass flux, Q, vs. bed axial position at t=tfinal
%...........................................................................
figure(7)
plot(z,Qfeed.*Qbar(nout,:),'LineWidth',2)
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Mass flux, Q
(mmol/cm^2/s)','color','k','fontsize',10,'fontweight','b')
title('Mass flux vs. bed axial position at
tfinal','color','k','fontsize',12,'fontweight','b')
grid on
%...........................................................................
% Superficial flow rate, us, vs. bed axial position at t=tfinal
%...........................................................................
figure(8)
plot(z,(L/tfeed).*usbar(nout,:),'LineWidth',2)
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
```

```
ylabel('Superficial velocity, u_s
(cm/s)','color','k','fontsize',10,'fontweight','b')
title('Superficial velocity vs. bed axial position at
tfinal','color','k','fontsize',12,'fontweight','b')
grid on
end % end of choose == 2
%..........................................................................
if choose == 3 % start of choose == 3 : Surface plots of y1, y2, n1, n2, T,
P, rhog, Q, us
%..........................................................................
% Gas phase mole fraction, y1
%..........................................................................
figure(9)
% mesh(z,t,y1bar,'EdgeColor','black')
surf(z,t,y1bar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Gas phase mole fraction,
y_{O2}','color','k','fontsize',10,'fontweight','b')
axis tight
%..........................................................................
% Gas phase mole fraction, y2
%..........................................................................
figure(10)
% mesh(z,t,(1-y1bar),'EdgeColor','black')
surf(z,t,(1-y1bar))
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Gas phase mole fraction,
y_{N2}','color','k','fontsize',10,'fontweight','b')
axis tight
%..........................................................................
% Amount adsorbed on adsorbent, n1
%..........................................................................
figure(11)
% mesh(z,t,(Qfeed/(k1*rhob*L)).*n1bar,'EdgeColor','black')
surf(z,t,(Qfeed/(k1*rhob*L)).*n1bar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Amount adsorbed on adsorbent, n_{O2} (mmol/g
adsorbent)','color','k','fontsize',10,'fontweight','b')
axis tight
%..........................................................................
% Amount adsorbed on adsorbent, n2
%..........................................................................
figure(12)
% mesh(z,t,(Qfeed/(k2*rhob*L)).*n2bar,'EdgeColor','black')
surf(z,t,(Qfeed/(k2*rhob*L)).*n2bar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Amount adsorbed on adsorbent, n_{N2} (mmol/g
adsorbent)','color','k','fontsize',10,'fontweight','b')
```

```
axis tight
%...........................................................................
% Temperature, T
%...........................................................................
figure(13)
% mesh(z,t,(Tmin+(Tmax-Tmin).*thetabar),'EdgeColor','black')
surf(z,t,(Tref+(Tmax-Tmin).*thetabar))
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Temperature, T (K)','color','k','fontsize',10,'fontweight','b')
axis tight
%...........................................................................
% Pressure, P
%...........................................................................
figure(14)
% mesh(z,t,((Pmax-Pmin)/1013250).*Pbar),'EdgeColor','black')
surf(z,t,((Pmax-Pmin)/1013250).*Pbar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Pressure, P (atm)','color','k','fontsize',10,'fontweight','b')
axis tight
%...........................................................................
% Gas phase concentration, rhog
%...........................................................................
figure(15)
% mesh(z,t,(Qfeed*tfeed/L).*rhogbar),'EdgeColor','black')
surf(z,t,(Qfeed*tfeed/L).*rhogbar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Gas phase concentration, \rho_g
(mmol/cm^3)','color','k','fontsize',10,'fontweight','b')
axis tight
%...........................................................................
% Mass flux, Q
%...........................................................................
figure(16)
% mesh(z,t,Qfeed.*Qbar),'EdgeColor','black')
surf(z,t,Qfeed.*Qbar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Mass flux, Q
(mmol/cm^2/s)','color','k','fontsize',10,'fontweight','b')
axis tight
%...........................................................................
% Superficial flow rate, us
%...........................................................................
figure(17)
% mesh(z,t,(L/tfeed).*usbar),'EdgeColor','black')
surf(z,t,(L/tfeed).*usbar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
```

```
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Superficial velocity, u_s
(cm/s)','color','k','fontsize',10,'fontweight','b')
axis tight
end % end of choose == 3
%..........................................................................
if choose == 4 % start of choose = 4 : The 2D plots that Dr. Sircar is
interested in
%..........................................................................
% Among nout points, take 5 time points spreading evenly between [0,tfeed]
%..........................................................................
for m=1:5
    it(m) = 1+ (m-1)*(nout-1)/4;
end
% If nout = 101 ==> it = 1    26    51    76   101

for m=1:5
    time((m)) = t(it(m));
end
% Resultstime = time          % Evenly spaced time for displayed results
%..........................................................................
% Now I want to take 5 time points between [0,tfeed] that display the
% results beautifully
%..........................................................................
% Resultstime = [t(1); t(11); t(21); t(31); t(41); t(51); t(101)]
%..........................................................................
% Adsorption / Desorption Profiles
%..........................................................................
% particle diameter in micrometer
dp = dp *10000;

if adsorptionORpurge == 111
% O2 initially present in the bed
O2ingas = epsilon*A*L*(y10)*rhog0;      % mmoles O2
O2insolid = A*L*rhob*n10;               % mmoles O2
TotalO2inbed0_mmolesO2 = O2ingas + O2insolid    % mmoles O2

% O2 remaining at every it, averaged over the whole bed i=1:n
for it = 1:nout
    y1acc(it) = 0;
    rhogbaracc(it) = 0;
    n1baracc(it) = 0;
    for i = 1:n
        y1acc(it) = y1bar(it,i) + y1acc(it);
        rhogbaracc(it) = rhogbar(it,i) + rhogbaracc(it);
        n1baracc(it) = n1bar(it,i) + n1baracc(it);
    end
    y1ave(it) = y1acc(it)/n;
    rhogave(it) = (Qfeed*tfeed/L)*rhogbaracc(it)/n;
    n1ave(it) = (Qfeed/(k1*rhob*L))*n1baracc(it)/n;
end

for it = 1:nout
O2remainingas(it) = epsilon*A*L*y1ave(it)*rhogave(it);    % mmoles O2
O2remaininsolid(it) = A*L*rhob*n1ave(it);                 % mmoles O2
TotalO2remaininbed_mmolesO2(it) = O2remainingas(it) + O2remaininsolid(it);
% mmoles O2
frac_O2_desorbed(it) = (TotalO2inbed0_mmolesO2 -
TotalO2remaininbed_mmolesO2(it))/TotalO2inbed0_mmolesO2;
end
```

```
frac_O2_desorbedv = frac_O2_desorbed';
end


if adsorptionORpurge == 222
% N2 initially present in the bed
N2ingas = epsilon*A*L*(1-y10)*rhog0;     % mmoles N2
N2insolid = A*L*rhob*n20;                % mmoles N2
TotalN2inbed0_mmolesN2 = N2ingas + N2insolid    % mmoles N2

% N2 remaining at every it, averaged over the whole bed i=1:n
for it = 1:nout
    y2acc(it) = 0;
    rhogbaracc(it) = 0;
    n2baracc(it) = 0;
    for i = 1:n
        y2acc(it) = (1-y1bar(it,i)) + y2acc(it);
        rhogbaracc(it) = rhogbar(it,i) + rhogbaracc(it);
        n2baracc(it) = n2bar(it,i) + n2baracc(it);
    end
    y2ave(it) = y2acc(it)/n;
    rhogave(it) = (Qfeed*tfeed/L)*rhogbaracc(it)/n;
    n2ave(it) = (Qfeed/(k2*rhob*L))*n2baracc(it)/n;
end

for it = 1:nout
N2remainingas(it) = epsilon*A*L*y2ave(it)*rhogave(it);    % mmoles N2
N2remaininsolid(it) = A*L*rhob*n2ave(it);                 % mmoles N2
TotalN2remaininbed_mmolesN2(it) = N2remainingas(it) + N2remaininsolid(it);
% mmoles N2
frac_N2_desorbed(it) = (TotalN2inbed0_mmolesN2 -
TotalN2remaininbed_mmolesN2(it))/TotalN2inbed0_mmolesN2;
end

frac_N2_desorbedv = frac_N2_desorbed';
end
%.....................................................................

%plot(zbar,(1-y1bar(1,:)),zbar,(1-y1bar((nout-1)*0.005+1,:)),zbar,(1-
y1bar((nout-1)*0.01+1,:)),zbar,(1-y1bar((nout-1)*0.05+1,:)),zbar,(1-
y1bar((nout-1)*0.1+1,:)),zbar,(1-y1bar((nout-1)*0.5+1,:)),zbar,(1-
y1bar(nout,:)),'LineWidth',2,'MarkerSize',3); %axis tight
%h =
legend('tbar=0','tbar=0.005','tbar=0.010','tbar=0.050','tbar=0.100','tbar=0.5
00','tbar=1.000',1);

%plot(zbar,(1-y1bar(1,:)),zbar,(1-y1bar((nout-1)*0.1+1,:)),zbar,(1-
y1bar((nout-1)*0.2+1,:)),zbar,(1-y1bar((nout-1)*0.3+1,:)),zbar,(1-
y1bar((nout-1)*0.4+1,:)),zbar,(1-y1bar((nout-1)*0.5+1,:)),zbar,(1-
y1bar((nout-1)*0.6+1,:)),zbar,(1-y1bar((nout-1)*0.7+1,:)),zbar,(1-
y1bar((nout-1)*0.8+1,:)),zbar,(1-y1bar((nout-1)*0.9+1,:)),zbar,(1-
y1bar(nout,:)),'LineWidth',2,'MarkerSize',3); %axis tight
%h =
legend('tbar=0.0','tbar=0.1','tbar=0.2','tbar=0.3','tbar=0.4','tbar=0.5','tba
r=0.6','tbar=0.7','tbar=0.8','tbar=0.9','tbar=1.0',1);


figure(20)
```

```
plot(zbar,(y1bar(1,:)),zbar,(y1bar((nout-1)*0.2+1,:)),zbar,(y1bar((nout-
1)*0.4+1,:)),zbar,(y1bar((nout-1)*0.6+1,:)),zbar,(y1bar((nout-
1)*0.8+1,:)),zbar,(y1bar(nout,:)),'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
text(.02,1.05,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('y_{O2}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.1])


figure(21)
plot(zbar,(1-y1bar(1,:)),zbar,(1-y1bar((nout-1)*0.2+1,:)),zbar,(1-
y1bar((nout-1)*0.4+1,:)),zbar,(1-y1bar((nout-1)*0.6+1,:)),zbar,(1-
y1bar((nout-1)*0.8+1,:)),zbar,(1-
y1bar(nout,:)),'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
text(.02,1.05,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('y_{N2}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.1])


figure(22)
plot(zbar,((Qfeed/(k1*rhob*L)).*n1bar(1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar(
(nout-1)*0.2+1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar((nout-
1)*0.4+1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar((nout-
1)*0.6+1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar((nout-
1)*0.8+1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar(nout,:)),'LineWidth',2,'MarkerS
ize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
```

```
ylabel('n_{O2} (mmoles O_2/g)','color','k','fontsize',12,'fontweight','b')


figure(23)
plot(zbar,((Qfeed/(k2*rhob*L)).*n2bar(1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar(
(nout-1)*0.2+1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar((nout-
1)*0.4+1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar((nout-
1)*0.6+1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar((nout-
1)*0.8+1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar(nout,:)),'LineWidth',2,'MarkerS
ize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{N2} (mmoles N_2/g)','color','k','fontsize',12,'fontweight','b')


figure(24)
plot(zbar,((Tmax-Tmin).*thetabar(1,:)+Tref),zbar,((Tmax-
Tmin).*thetabar((nout-1)*0.2+1,:)+Tref),zbar,((Tmax-Tmin).*thetabar((nout-
1)*0.4+1,:)+Tref),zbar,((Tmax-Tmin).*thetabar((nout-
1)*0.6+1,:)+Tref),zbar,((Tmax-Tmin).*thetabar((nout-
1)*0.8+1,:)+Tref),zbar,((Tmax-Tmin).*thetabar((nout-
1)*1.0+1,:)+Tref),'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('T_g = T_s (K)','color','k','fontsize',12,'fontweight','b')
axis([0 1 282 300])


figure(25)
plot(zbar,(Pmax-Pmin).*Pbar(1,:)*14.69595/1013250,zbar,(Pmax-
Pmin).*Pbar((nout-1)*0.2+1,:)*14.69595/1013250,zbar,(Pmax-Pmin).*Pbar((nout-
1)*0.4+1,:)*14.69595/1013250,zbar,(Pmax-Pmin).*Pbar((nout-
1)*0.6+1,:)*14.69595/1013250,zbar,(Pmax-Pmin).*Pbar((nout-
1)*0.8+1,:)*14.69595/1013250,zbar,(Pmax-Pmin).*Pbar((nout-
1)*1.0+1,:)*14.69595/1013250,'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
```

```
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('Pressure (psi)','color','k','fontsize',12,'fontweight','b')


if adsorptionORpurge == 222
%.........................................................................
% Search the tbar that produces fracdes = 0.3, 0.5, 0.7
% Then plot the profiles for yN2, nN2, Tg, Ts at these tbar
%.........................................................................
minDF_3 = 1; % minimum difference between fraction desorbed and 0.3
for it=1:nout
    F_3 = abs(frac_N2_desorbed(1,it) - 0.3);
    if minDF_3 > F_3
        minDF_3 = F_3;
        it_3 = it;
    end
end
tbar_3 = tbarv(it_3,1)
minDF_3

minDF_5 = 1; % minimum difference between fraction desorbed and 0.5
for it=1:nout
    F_5 = abs(frac_N2_desorbed(1,it) - 0.5);
    if minDF_5 > F_5
        minDF_5 = F_5;
        it_5 = it;
    end
end
tbar_5 = tbarv(it_5,1)
minDF_5

minDF_7 = 1; % minimum difference between fraction desorbed and 0.7
for it=1:nout
    F_7 = abs(frac_N2_desorbed(1,it) - 0.7);
    if minDF_7 > F_7
        minDF_7 = F_7;
        it_7 = it;
    end
end
tbar_7 = tbarv(it_7,1)
minDF_7

minDF_8 = 1; % minimum difference between fraction desorbed and 0.8
for it=1:nout
    F_8 = abs(frac_N2_desorbed(1,it) - 0.8);
    if minDF_8 > F_8
        minDF_8 = F_8;
        it_8 = it;
    end
end
tbar_8 = tbarv(it_8,1)
minDF_8

minDF_85 = 1; % minimum difference between fraction desorbed and 0.85
for it=1:nout
    F_85 = abs(frac_N2_desorbed(1,it) - 0.85);
    if minDF_85 > F_85
```

```
        minDF_85 = F_85;
        it_85 = it;
    end
end
tbar_85 = tbarv(it_85,1)
minDF_85

minDF_9 = 1; % minimum difference between fraction desorbed and 0.9
for it=1:nout
    F_9 = abs(frac_N2_desorbed(1,it) - 0.9);
    if minDF_9 > F_9
        minDF_9 = F_9;
        it_9 = it;
    end
end
tbar_9 = tbarv(it_9,1)
minDF_9

minDF_95 = 1; % minimum difference between fraction desorbed and 0.95
for it=1:nout
    F_95 = abs(frac_N2_desorbed(1,it) - 0.95);
    if minDF_95 > F_95
        minDF_95 = F_95;
        it_95 = it;
    end
end
tbar_95 = tbarv(it_95,1)
minDF_95
%.....................................................................
for i = 1:n
    Nn1f357(1,i)=(Qfeed/(k1*rhob*L))*n1bar(1   ,i)/n10; % normalized with n10
    Nn1f357(2,i)=(Qfeed/(k1*rhob*L))*n1bar(it_3,i)/n10;
    Nn1f357(3,i)=(Qfeed/(k1*rhob*L))*n1bar(it_5,i)/n10;
    Nn1f357(4,i)=(Qfeed/(k1*rhob*L))*n1bar(it_7,i)/n10;
    Nn1f357(5,i)=(Qfeed/(k1*rhob*L))*n1bar(it_8,i)/n10;
    Nn1f357(6,i)=(Qfeed/(k1*rhob*L))*n1bar(it_85,i)/n10;
    Nn1f357(7,i)=(Qfeed/(k1*rhob*L))*n1bar(it_9,i)/n10;
    Nn1f357(8,i)=(Qfeed/(k1*rhob*L))*n1bar(it_95,i)/n10;

    n1f357(1,i)=(Qfeed/(k1*rhob*L))*n1bar(1   ,i); % mmol O2/g
    n1f357(2,i)=(Qfeed/(k1*rhob*L))*n1bar(it_3,i);
    n1f357(3,i)=(Qfeed/(k1*rhob*L))*n1bar(it_5,i);
    n1f357(4,i)=(Qfeed/(k1*rhob*L))*n1bar(it_7,i);
    n1f357(5,i)=(Qfeed/(k1*rhob*L))*n1bar(it_8,i);
    n1f357(6,i)=(Qfeed/(k1*rhob*L))*n1bar(it_85,i);
    n1f357(7,i)=(Qfeed/(k1*rhob*L))*n1bar(it_9,i);
    n1f357(8,i)=(Qfeed/(k1*rhob*L))*n1bar(it_95,i);

    Nn2f357(1,i)=(Qfeed/(k2*rhob*L))*n2bar(1   ,i)/n20; % normalized with n20
    Nn2f357(2,i)=(Qfeed/(k2*rhob*L))*n2bar(it_3,i)/n20;
    Nn2f357(3,i)=(Qfeed/(k2*rhob*L))*n2bar(it_5,i)/n20;
    Nn2f357(4,i)=(Qfeed/(k2*rhob*L))*n2bar(it_7,i)/n20;
    Nn2f357(5,i)=(Qfeed/(k2*rhob*L))*n2bar(it_8,i)/n20;
    Nn2f357(6,i)=(Qfeed/(k2*rhob*L))*n2bar(it_85,i)/n20;
    Nn2f357(7,i)=(Qfeed/(k2*rhob*L))*n2bar(it_9,i)/n20;
    Nn2f357(8,i)=(Qfeed/(k2*rhob*L))*n2bar(it_95,i)/n20;

    n2f357(1,i)=(Qfeed/(k2*rhob*L))*n2bar(1   ,i); % mmol N2/g
    n2f357(2,i)=(Qfeed/(k2*rhob*L))*n2bar(it_3,i);
    n2f357(3,i)=(Qfeed/(k2*rhob*L))*n2bar(it_5,i);
    n2f357(4,i)=(Qfeed/(k2*rhob*L))*n2bar(it_7,i);
    n2f357(5,i)=(Qfeed/(k2*rhob*L))*n2bar(it_8,i);
    n2f357(6,i)=(Qfeed/(k2*rhob*L))*n2bar(it_85,i);
    n2f357(7,i)=(Qfeed/(k2*rhob*L))*n2bar(it_9,i);
    n2f357(8,i)=(Qfeed/(k2*rhob*L))*n2bar(it_95,i);

    totn10=(A*L*rhob)*n10 + epsilon*A*L*(y10)*rhog0;   % total initial mmol
O2 in solid and gas phases
    totn20=(A*L*rhob)*n20 + epsilon*A*L*(1-y10)*rhog0; % total initial mmol
N2 in solid and gas phases

    totn1f357(1,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(1,i) +
epsilon*A*L*(y1bar(1,i))*rhogbar(1,i)*(Qfeed*tfeed/L); % total mmol O2 in gas
and solid phases
    totn1f357(2,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_3,i) +
epsilon*A*L*(y1bar(it_3,i))*rhogbar(it_3,i)*(Qfeed*tfeed/L);
    totn1f357(3,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_5,i) +
epsilon*A*L*(y1bar(it_5,i))*rhogbar(it_5,i)*(Qfeed*tfeed/L);
    totn1f357(4,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_7,i) +
epsilon*A*L*(y1bar(it_7,i))*rhogbar(it_7,i)*(Qfeed*tfeed/L);
    totn1f357(5,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_8,i) +
epsilon*A*L*(y1bar(it_8,i))*rhogbar(it_8,i)*(Qfeed*tfeed/L);
    totn1f357(6,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_85,i) +
epsilon*A*L*(y1bar(it_85,i))*rhogbar(it_85,i)*(Qfeed*tfeed/L);
    totn1f357(7,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_9,i) +
epsilon*A*L*(y1bar(it_9,i))*rhogbar(it_9,i)*(Qfeed*tfeed/L);
    totn1f357(8,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_95,i) +
epsilon*A*L*(y1bar(it_95,i))*rhogbar(it_95,i)*(Qfeed*tfeed/L);

    Ntotn1f357(1,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(1,i) +
epsilon*A*L*(y1bar(1,i))*rhogbar(1,i)*(Qfeed*tfeed/L))/totn10; % normalized
with totn10
    Ntotn1f357(2,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_3,i) +
epsilon*A*L*(y1bar(it_3,i))*rhogbar(it_3,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(3,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_5,i) +
epsilon*A*L*(y1bar(it_5,i))*rhogbar(it_5,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(4,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_7,i) +
epsilon*A*L*(y1bar(it_7,i))*rhogbar(it_7,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(5,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_8,i) +
epsilon*A*L*(y1bar(it_8,i))*rhogbar(it_8,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(6,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_85,i) +
epsilon*A*L*(y1bar(it_85,i))*rhogbar(it_85,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(7,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_9,i) +
epsilon*A*L*(y1bar(it_9,i))*rhogbar(it_9,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(8,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_95,i) +
epsilon*A*L*(y1bar(it_95,i))*rhogbar(it_95,i)*(Qfeed*tfeed/L))/totn10;

    totn2f357(1,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(1,i) +
epsilon*A*L*(1-y1bar(1,i))*rhogbar(1,i)*(Qfeed*tfeed/L); % total mmol N2 in
gas and solid phases
    totn2f357(2,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_3,i) +
epsilon*A*L*(1-y1bar(it_3,i))*rhogbar(it_3,i)*(Qfeed*tfeed/L);
    totn2f357(3,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_5,i) +
epsilon*A*L*(1-y1bar(it_5,i))*rhogbar(it_5,i)*(Qfeed*tfeed/L);
    totn2f357(4,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_7,i) +
epsilon*A*L*(1-y1bar(it_7,i))*rhogbar(it_7,i)*(Qfeed*tfeed/L);
    totn2f357(5,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_8,i) +
epsilon*A*L*(1-y1bar(it_8,i))*rhogbar(it_8,i)*(Qfeed*tfeed/L);
    totn2f357(6,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_85,i) +
epsilon*A*L*(1-y1bar(it_85,i))*rhogbar(it_85,i)*(Qfeed*tfeed/L);
```

```
    totn2f357(7,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_9,i) + 
epsilon*A*L*(1-y1bar(it_9,i))*rhogbar(it_9,i)*(Qfeed*tfeed/L);
    totn2f357(8,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_95,i) + 
epsilon*A*L*(1-y1bar(it_95,i))*rhogbar(it_95,i)*(Qfeed*tfeed/L);

    Ntotn2f357(1,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(1,i) + 
epsilon*A*L*(1-y1bar(1,i))*rhogbar(1,i)*(Qfeed*tfeed/L))/totn20; % normalized 
with totn20
    Ntotn2f357(2,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_3,i) + 
epsilon*A*L*(1-y1bar(it_3,i))*rhogbar(it_3,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(3,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_5,i) + 
epsilon*A*L*(1-y1bar(it_5,i))*rhogbar(it_5,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(4,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_7,i) + 
epsilon*A*L*(1-y1bar(it_7,i))*rhogbar(it_7,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(5,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_8,i) + 
epsilon*A*L*(1-y1bar(it_8,i))*rhogbar(it_8,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(6,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_85,i) + 
epsilon*A*L*(1-y1bar(it_85,i))*rhogbar(it_85,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(7,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_9,i) + 
epsilon*A*L*(1-y1bar(it_9,i))*rhogbar(it_9,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(8,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_95,i) + 
epsilon*A*L*(1-y1bar(it_95,i))*rhogbar(it_95,i)*(Qfeed*tfeed/L))/totn20;

    Remain_N2_inbed_percent(1,i) = 100*totn2f357(1,i)/(totn1f357(1,i) + 
totn2f357(1,i));
    Remain_N2_inbed_percent(2,i) = 100*totn2f357(2,i)/(totn1f357(2,i) + 
totn2f357(2,i));
    Remain_N2_inbed_percent(3,i) = 100*totn2f357(3,i)/(totn1f357(3,i) + 
totn2f357(3,i));
    Remain_N2_inbed_percent(4,i) = 100*totn2f357(4,i)/(totn1f357(4,i) + 
totn2f357(4,i));
    Remain_N2_inbed_percent(5,i) = 100*totn2f357(5,i)/(totn1f357(5,i) + 
totn2f357(5,i));
    Remain_N2_inbed_percent(6,i) = 100*totn2f357(6,i)/(totn1f357(6,i) + 
totn2f357(6,i));
    Remain_N2_inbed_percent(7,i) = 100*totn2f357(7,i)/(totn1f357(7,i) + 
totn2f357(7,i));
    Remain_N2_inbed_percent(8,i) = 100*totn2f357(8,i)/(totn1f357(8,i) + 
totn2f357(8,i));

    y1barf357(1,i)=y1bar(1   ,i);
    y1barf357(2,i)=y1bar(it_3,i);
    y1barf357(3,i)=y1bar(it_5,i);
    y1barf357(4,i)=y1bar(it_7,i);
    y1barf357(5,i)=y1bar(it_8,i);
    y1barf357(6,i)=y1bar(it_85,i);
    y1barf357(7,i)=y1bar(it_9,i);
    y1barf357(8,i)=y1bar(it_95,i);

    y2barf357(1,i)=1-y1bar(1   ,i);
    y2barf357(2,i)=1-y1bar(it_3,i);
    y2barf357(3,i)=1-y1bar(it_5,i);
    y2barf357(4,i)=1-y1bar(it_7,i);
    y2barf357(5,i)=1-y1bar(it_8,i);
    y2barf357(6,i)=1-y1bar(it_85,i);
    y2barf357(7,i)=1-y1bar(it_9,i);
    y2barf357(8,i)=1-y1bar(it_95,i);

    Tgf357(1,i)=(Tmax-Tmin)*thetabar(1   ,i)+Tref; % gas phase temperature, K
    Tgf357(2,i)=(Tmax-Tmin)*thetabar(it_3,i)+Tref;
    Tgf357(3,i)=(Tmax-Tmin)*thetabar(it_5,i)+Tref;
    Tgf357(4,i)=(Tmax-Tmin)*thetabar(it_7,i)+Tref;
    Tgf357(5,i)=(Tmax-Tmin)*thetabar(it_8,i)+Tref;
    Tgf357(6,i)=(Tmax-Tmin)*thetabar(it_85,i)+Tref;
    Tgf357(7,i)=(Tmax-Tmin)*thetabar(it_9,i)+Tref;
    Tgf357(8,i)=(Tmax-Tmin)*thetabar(it_95,i)+Tref;

    thetabargf357(1,i)=thetabar(1   ,i);
    thetabargf357(2,i)=thetabar(it_3,i);
    thetabargf357(3,i)=thetabar(it_5,i);
    thetabargf357(4,i)=thetabar(it_7,i);
    thetabargf357(5,i)=thetabar(it_8,i);
    thetabargf357(6,i)=thetabar(it_85,i);
    thetabargf357(7,i)=thetabar(it_9,i);
    thetabargf357(8,i)=thetabar(it_95,i);

    Tsf357(1,i)=(Tmax-Tmin)*thetabar(1   ,i)+Tref; % solid phase temperature,
K
    Tsf357(2,i)=(Tmax-Tmin)*thetabar(it_3,i)+Tref;
    Tsf357(3,i)=(Tmax-Tmin)*thetabar(it_5,i)+Tref;
    Tsf357(4,i)=(Tmax-Tmin)*thetabar(it_7,i)+Tref;
    Tsf357(5,i)=(Tmax-Tmin)*thetabar(it_8,i)+Tref;
    Tsf357(6,i)=(Tmax-Tmin)*thetabar(it_85,i)+Tref;
    Tsf357(7,i)=(Tmax-Tmin)*thetabar(it_9,i)+Tref;
    Tsf357(8,i)=(Tmax-Tmin)*thetabar(it_95,i)+Tref;

    thetabarsf357(1,i)=thetabar(1   ,i);
    thetabarsf357(2,i)=thetabar(it_3,i);
    thetabarsf357(3,i)=thetabar(it_5,i);
    thetabarsf357(4,i)=thetabar(it_7,i);
    thetabarsf357(5,i)=thetabar(it_8,i);
    thetabarsf357(6,i)=thetabar(it_85,i);
    thetabarsf357(7,i)=thetabar(it_9,i);
    thetabarsf357(8,i)=thetabar(it_95,i);

    Pbarf357(1,i)=Pbar(1   ,i); % pressure in dimensionless
    Pbarf357(2,i)=Pbar(it_3,i);
    Pbarf357(3,i)=Pbar(it_5,i);
    Pbarf357(4,i)=Pbar(it_7,i);
    Pbarf357(5,i)=Pbar(it_8,i);
    Pbarf357(6,i)=Pbar(it_85,i);
    Pbarf357(7,i)=Pbar(it_9,i);
    Pbarf357(8,i)=Pbar(it_95,i);
end
%.....................................................................
figure(28)
plot(zbar,y1barf357(1,:), zbar,y1barf357(2,:), zbar,y1barf357(3,:),
zbar,y1barf357(4,:), zbar,y1barf357(5,:), zbar,y1barf357(6,:) ,
zbar,y1barf357(7,:), zbar,y1barf357(8,:),'LineWidth',2,'MarkerSize',3); %axis
tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
text(.02,1.25,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
text(.02,1.15,['\fontsize{9}(f=0.30 at tbar=',num2str(tbar_3),');
\fontsize{9}(f=0.50 at tbar=',num2str(tbar_5) ,');  \fontsize{9}(f=0.70 at
tbar=',num2str(tbar_7),');'])
text(.02,1.07,['\fontsize{9}(f=0.80 at tbar=',num2str(tbar_8),');
\fontsize{9}(f=0.85 at tbar=',num2str(tbar_85),');  \fontsize{9}(f=0.90 at
tbar=',num2str(tbar_9),');  \fontsize{9}(f=0.95 at
tbar=',num2str(tbar_95),');'])
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('y_{O2}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.3])

figure(29)
plot(zbar,y2barf357(1,:), zbar,y2barf357(2,:), zbar,y2barf357(3,:),
zbar,y2barf357(4,:), zbar,y2barf357(5,:), zbar,y2barf357(6,:) ,
zbar,y2barf357(7,:), zbar,y2barf357(8,:),'LineWidth',2,'MarkerSize',3); %axis
tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
text(.02,1.25,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
text(.02,1.15,['\fontsize{9}(f=0.30 at tbar=',num2str(tbar_3),');
\fontsize{9}(f=0.50 at tbar=',num2str(tbar_5) ,');  \fontsize{9}(f=0.70 at
tbar=',num2str(tbar_7),');'])
text(.02,1.07,['\fontsize{9}(f=0.80 at tbar=',num2str(tbar_8),');
\fontsize{9}(f=0.85 at tbar=',num2str(tbar_85),');  \fontsize{9}(f=0.90 at
tbar=',num2str(tbar_9),');  \fontsize{9}(f=0.95 at
tbar=',num2str(tbar_95),');'])
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('y_{N2}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.3])

figure(30)
plot(zbar,n1f357(1,:), zbar,n1f357(2,:), zbar,n1f357(3,:), zbar,n1f357(4,:),
zbar,n1f357(5,:), zbar,n1f357(6,:), zbar,n1f357(7,:), zbar,n1f357(8,:)
,'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{O2} (mmoles O_2/g)','color','k','fontsize',12,'fontweight','b')

figure(31)
plot(zbar,n2f357(1,:), zbar,n2f357(2,:), zbar,n2f357(3,:), zbar,n2f357(4,:),
zbar,n2f357(5,:), zbar,n2f357(6,:), zbar,n2f357(7,:), zbar,n2f357(8,:)
,'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{N2} (mmoles N_2/g)','color','k','fontsize',12,'fontweight','b')

figure(32)
plot(zbar,Nn1f357(1,:), zbar,Nn1f357(2,:), zbar,Nn1f357(3,:),
zbar,Nn1f357(4,:), zbar,Nn1f357(5,:), zbar,Nn1f357(6,:), zbar,Nn1f357(7,:),
zbar,Nn1f357(8,:) ,'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{O2}/n_{O2_0}','color','k','fontsize',12,'fontweight','b')

figure(33)
plot(zbar,Nn2f357(1,:), zbar,Nn2f357(2,:), zbar,Nn2f357(3,:),
zbar,Nn2f357(4,:), zbar,Nn2f357(5,:), zbar,Nn2f357(6,:), zbar,Nn2f357(7,:),
zbar,Nn2f357(8,:) ,'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
```

```
text(.02,1.25,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
text(.02,1.15,['\fontsize{9}(f=0.30 at tbar=',num2str(tbar_3),');
\fontsize{9}(f=0.50 at tbar=',num2str(tbar_5) ,');  \fontsize{9}(f=0.70 at
tbar=',num2str(tbar_7),');'])
text(.02,1.07,['\fontsize{9}(f=0.80 at tbar=',num2str(tbar_8),');
\fontsize{9}(f=0.85 at tbar=',num2str(tbar_85),');  \fontsize{9}(f=0.90 at
tbar=',num2str(tbar_9),');  \fontsize{9}(f=0.95 at
tbar=',num2str(tbar_95),');'])
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{N2}/n_{N2_0}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.3])

figure(34)
plot(zbar,totn2f357(1,:), zbar,totn2f357(2,:), zbar,totn2f357(3,:),
zbar,totn2f357(4,:), zbar,totn2f357(5,:), zbar,totn2f357(6,:),
zbar,totn2f357(7,:), zbar,totn2f357(8,:) ,'LineWidth',2,'MarkerSize',3);
%axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('total N_2 in gas and solid phases, totn_{N2}, (mmoles
N_2)','color','k','fontsize',12,'fontweight','b')

figure(35)
plot(zbar,Ntotn2f357(1,:), zbar,Ntotn2f357(2,:), zbar,Ntotn2f357(3,:),
zbar,Ntotn2f357(4,:), zbar,Ntotn2f357(5,:), zbar,Ntotn2f357(6,:),
zbar,Ntotn2f357(7,:), zbar,Ntotn2f357(8,:) ,'LineWidth',2,'MarkerSize',3);
%axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
text(.02,1.25,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
text(.02,1.15,['\fontsize{9}(f=0.30 at tbar=',num2str(tbar_3),');
\fontsize{9}(f=0.50 at tbar=',num2str(tbar_5) ,');  \fontsize{9}(f=0.70 at
tbar=',num2str(tbar_7),');'])
text(.02,1.07,['\fontsize{9}(f=0.80 at tbar=',num2str(tbar_8),');
\fontsize{9}(f=0.85 at tbar=',num2str(tbar_85),');  \fontsize{9}(f=0.90 at
tbar=',num2str(tbar_9),');  \fontsize{9}(f=0.95 at
tbar=',num2str(tbar_95),');'])
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('totn_{N2}/totn_{N2_0}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.3])

figure(36)
plot(zbar,(Pmax-Pmin).*Pbarf357(1,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(2,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(3,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(4,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(5,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(6,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(7,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(8,:)*14.69595/1013250,'LineWidth',2,'MarkerSize',3); %axis
tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('Pressure (psi)','color','k','fontsize',12,'fontweight','b')

figure(37)
```

```
plot(zbar,Tgf357(1,:),'--k',zbar,Tsf357(1,:),'-k',zbar,Tgf357(2,:),'--
m',zbar,Tsf357(2,:),'-m', zbar,Tgf357(3,:),'--b',zbar,Tsf357(3,:),'-b',
zbar,Tgf357(4,:),'--r',zbar,Tsf357(4,:),'-r', zbar,Tgf357(5,:),'--
g',zbar,Tsf357(5,:),'-g', zbar,Tgf357(6,:),'--y', zbar,Tsf357(6,:),'-y',
zbar,Tgf357(7,:),'--c',zbar,Tsf357(7,:),'-c', zbar,Tgf357(8,:),'--p',
zbar,Tsf357(8,:),'-p','LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('Tg_{f=0.0}','Ts_{f=0.0}','Tg_{f=0.3}','Ts_{f=0.3}','Tg_{f=0.5}','Ts_{
f=0.5}','Tg_{f=0.7}','Ts_{f=0.7}','Tg_{f=0.8}','Ts_{f=0.8}','Tg_{f=0.85}','Ts
_{f=0.85}','Tg_{f=0.9}','Ts_{f=0.9}','Tg_{f=0.95}','Ts_{f=0.95}',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('T_g and T_s (K)','color','k','fontsize',12,'fontweight','b')

figure(38)
plot(zbar,Tgf357(1,:)-Tref,'--k',zbar,Tsf357(1,:)-Tref,'-k',zbar,Tgf357(2,:)-
Tref,'--m',zbar,Tsf357(2,:)-Tref,'-m', zbar,Tgf357(3,:)-Tref,'--
b',zbar,Tsf357(3,:)-Tref,'-b', zbar,Tgf357(4,:)-Tref,'--r',zbar,Tsf357(4,:)-
Tref,'-r', zbar,Tgf357(5,:)-Tref,'--g',zbar,Tsf357(5,:)-Tref,'-g',
zbar,Tgf357(6,:)-Tref,'--y', zbar,Tsf357(6,:)-Tref,'-y', zbar,Tgf357(7,:)-
Tref,'--c',zbar,Tsf357(7,:)-Tref,'-c', zbar,Tgf357(8,:)-Tref,'--p',
zbar,Tsf357(8,:)-Tref,'-p', 'LineWidth',2,'MarkerSize',3); %axis tight
h = legend('Tg-T0_{f=0.0}','Ts-T0_{f=0.0}','Tg-T0_{f=0.3}','Ts-
T0_{f=0.3}','Tg-T0_{f=0.5}','Ts-T0_{f=0.5}','Tg-T0_{f=0.7}','Ts-
T0_{f=0.7}','Tg-T0_{f=0.8}','Ts-T0_{f=0.8}','Tg-T0_{f=0.85}','Ts-
T0_{f=0.85}','Tg-T0_{f=0.9}','Ts-T0_{f=0.9}','Tg-T0_{f=0.95}','Ts-
T0_{f=0.95}',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('(T_g - T_0) and (T_s - T_0)
(K)','color','k','fontsize',12,'fontweight','b')
end
end % end of choose == 4
%...................................................................
```

Appendix 4

Model D

MOL Routine

```
%..........................................................................
% File: psa_1_sequence_model_D
%..........................................................................
% Chai Siew Wah, Chemical Engineering, Lehigh University, May 2011
%..........................................................................
% MATHEMATICAL MODEL OF PRESSURE SWING ADSORPTION SYSTEM FOR ONE STEP :
% ADSORPTION OR PURGE
%..........................................................................
%
%                  :......................................:
%   ADSORPTION --> :                                      : --> PRODUCT
%   ADSORPTION --> :                                      : --> PRODUCT
%   ADSORPTION --> :                                      : --> PRODUCT
%                  :......................................:
%                 i=1                                    i=n
%
%                  :......................................:
%     EXHAUST <--  :                                      : <-- PURGE
%     EXHAUST <--  :                                      : <-- PURGE
%     EXHAUST <--  :                                      : <-- PURGE
%                  :......................................:
%                 i=1                                    i=n
%..........................................................................
% Assumptions
%..........................................................................
% (1) Ideal gas law
% (2) Empirical Langmuirian type binary isotherms - from Rege and Yang 1997
% (3) Nonisobaric - Ergun equation solved analytically
% (4) Isothermal
% (5) Linear driving force model for mass transfer kinetic
% (6) Mass axial dispersion in gas phase
% (7) Absence of solid phase thermal axial conduction
% (8) Absence of radial distribution of mass and heat
% (9) Absence of gas mal-distribution or particle agglomeration
%..........................................................................
% 4 ordinary differential equations : y1, y2, n1, n2
% 1 direct calculation for          : rhogbar (= Y1bar + Y2bar)
% 1 algebraic equation for          : Q  (= Ergun equation)
% 1 algebraic equation for          : P  (= Ideal gas law)
% 1 calculated for                  : us (= Q/rhog) - but reduce its use
% Solve for y1, y2, n1, n2, P, rhog, Q, us
%..........................................................................
% (1) Solve Ergun equation for Qbar and properly dealing its positive or
% negative flow direction

% (2) Use Schiesser's boundary condition for mass and heat balances that
% have axial dispersion terms (retain the whole equation except the 2nd
% derivative of space) instead of Danwerts' boundary condition of dY/dz = 0
% at exit

% (3) Eliminate overall mass balance, use 2nd component mass balance, so
% that I do not need a boundary condition for rhog (there is no defined
% rhog at inlet or exit), and I can use flux conservation formulation of
% Q throughout, and rhogbar = Y1bar + Y2bar.

% (4) Gas compositions y1feed y2feed are specified, rhogbar(feed end) =
% Y1bar + Y2bar is not. But I need the right BCs for Y1bar, Y2bar,
% which depend on rhogbar(feed end).

% (5) Normally Pbar(i) is calculated from rhogbar(i) using ideal gas law.
% But at feed end, rhogbar(feed end) is undetermined. So, Pressure comes
% in to help by using its previous dP/dz and Pbar to calculate
% Pbar(feed end), then calculate rhogbar(feed end) from ideal gas law,
% then calculate the right BCs for Y1bar, Y2bar.

% (6) Correct use of: epsilon (gas phase accumulation), epsilonbar(Ergun,
% axial dispersions), mass axial dispersion, from literature values (Rege
% and Yang 1997 provides Cs, Cg, q1, q2, isotherms), adsorbent density from
bed
% dimensions and 160 g LiX in Invacare test unit.

% (7) Only allow average velocity,uz = us/epsilon, and mass flux, Q, to
% acquire positive or negative value, all the rest, y1, y2, n1, n2, Tg, Ts,
% P, rhog, must use positive values
%.......................................................................
function yt=psa_1_sequence_model_D(t,Y)
%.......................................................................
% Parameters shared with the ODE routine
%.......................................................................
global  Mg1 Mg2 adsorptionORpurge dYbardZbarDifferentialMethod T0......
    ncall n dzbar dtbar miu epsilon epsilonbar k1 k2 q1 q2 Cs Cg rhob R....
    dp L tfeed y1feed y2feed Qin Qfeed P0 Pout Tfeed Pmax Pmin Tmax Tmin...
    Tref Q0 kg MassAxial HeatAxial HeateqmORnoneqm Dem4 Dem1
%.......................................................................
% One vector to four vectors
%.......................................................................
for i = 1:n
    Y1bar(i)    = Y(0*n+i);
    n1bar(i)    = Y(1*n+i);
    n2bar(i)    = Y(2*n+i);
    %Thetabarg(i)= Y(3*n+i);
    %Thetabars(i)= Y(4*n+i);
    Y2bar(i)    = Y(3*n+i);
end

for i = 1:n
    y1bar(i)    = Y1bar(i)/(Y1bar(i)+Y2bar(i));
    y2bar(i)    = Y2bar(i)/(Y1bar(i)+Y2bar(i));
    rhogbar(i)  = Y1bar(i) + Y2bar(i);
    Mg(i)       = Mg1*y1bar(i) + Mg2*(1-y1bar(i));
    %thetabarg(i)= Thetabarg(i)/rhogbar(i);
    %thetabars(i)= Thetabars(i);
end
%.......................................................................
% Boundary conditions after initial conditions
%.......................................................................
if (ncall~=0)       % not initial condition

if adsorptionORpurge == 111
   i = 1;           % Only use boundary conditions at the adsorption inlet
end
if adsorptionORpurge == 222
```

```
   i = n;           % Only use boundary conditions at the purge inlet
end

y1bar(i)         = y1feed;
y2bar(i)         = y2feed;
Y1bar(i)         = y1feed*rhogbar(i);
Y2bar(i)         = y2feed*rhogbar(i);
%thetabarg(i)     = (Tfeed-Tref)/(Tmax-Tmin);
%Thetabarg(i)     = ((Tfeed-Tref)/(Tmax-Tmin))*rhogbar(i);
%thetabars(i)     = (Tfeed-Tref)/(Tmax-Tmin);
%Thetabars(i)     = (Tfeed-Tref)/(Tmax-Tmin);
% rhogbar(feed end) needs to be taken care of by Pbar(feed end)
end
%...........................................................................
% Formulate pre-parameters for ODEs
%...........................................................................
% Isothermal
%...........................................................................
for i = 1:n
      thetabar(i) = (T0-Tref)/(Tmax-Tmin);
end
%...........................................................................
% Formulate Pbar(i) using ideal gas law except that at inlet point
%...........................................................................
if (ncall==0)
    for i = 1:n
        Pbar(i) = P0/(Pmax-Pmin);
    end
end

if (ncall~=0)
%...........................................................................
    if adsorptionORpurge == 111
        i = n;          % BC for adsorption step
        Pbar(i) = Pout/(Pmax-Pmin);

        for i = n-1:-1:2
            Pbar(i) = rhogbar(i)*(thetabar(i)*(Tmax-Tmin)+Tref)*(R*.......
                Qfeed*tfeed)/((Pmax-Pmin)*L);
            dPbardzbar(i) = (Pbar(i+1)-Pbar(i))/dzbar;
        end

        % I cannot depend on rhogbar(feed end). dP/dz comes in to help
        for i = 1
            Pbar(i)       = Pbar(i+1) - dPbardzbar(i+1)*dzbar;
            rhogbar(i)    = Pbar(i)*(Pmax-Pmin)*L/((thetabar(i)*(Tmax-....
                Tmin)+Tref)*R*Qfeed*tfeed);
            dPbardzbar(i) = (Pbar(i+1)-Pbar(i))/dzbar;
            Y1bar(i)      = y1feed*rhogbar(i);
            Y2bar(i)      = y2feed*rhogbar(i);
            Thetabarg(i)  = ((Tfeed-Tref)/(Tmax-Tmin))*rhogbar(i);
        end

        for i = n           % special upwinding
            dPbardzbar(i) = (Pbar(i)-Pbar(i-1))/dzbar;
        end
    end
%...........................................................................
    if adsorptionORpurge == 222
        i = 1;          % BC for purge step
        Pbar(i) = Pout/(Pmax-Pmin);

        for i = 2:n-1
            Pbar(i) = rhogbar(i)*(thetabar(i)*(Tmax-Tmin)+Tref)*(R*........
                Qfeed*tfeed)/((Pmax-Pmin)*L);
            dPbardzbar(i) = (Pbar(i)-Pbar(i-1))/dzbar;
        end

        % I cannot depend on rhogbar(feed end). dP/dz comes in to help
        for i = n
            Pbar(i)        = dPbardzbar(i-1)*dzbar + Pbar(i-1);
            rhogbar(i)     = Pbar(i)*(Pmax-Pmin)*L/((thetabar(i)*(Tmax-.....
                Tmin)+Tref)*R*Qfeed*tfeed);
            dPbardzbar(i)  = (Pbar(i)-Pbar(i-1))/dzbar;
            Y1bar(i)       = y1feed*rhogbar(i);
            Y2bar(i)       = y2feed*rhogbar(i);
            Thetabarg(i)   = ((Tfeed-Tref)/(Tmax-Tmin))*rhogbar(i);
        end

        for i = 1           % special upwinding
            dPbardzbar(i) = (Pbar(i+1)-Pbar(i))/dzbar;
        end
    end
%..........................................................................
end
%..........................................................................
% Solving for usbar(i) in Ergun equation using quadratic formula - failed
% Solving for Qbar(i) in Ergun equation using quadratic formula
%..........................................................................
if (ncall==0)       % initial condition
    for i = 1:n
        Qbar(i) = Q0/Qfeed;
    end
end

if (ncall~=0)
    if adsorptionORpurge == 111
        Qbar(1) = Qin/Qfeed;
        for i = 2:n
            a = 1.75*Mg(i)*L*(1-epsilonbar)/(dp*rhogbar(i)*epsilonbar^2);
            b = 150*miu*L*(1-epsilonbar)^2/((dp^2)*Qfeed*rhogbar(i)*.......
                epsilonbar^2);
            c = epsilonbar*tfeed*(Pmax-Pmin)*dPbardzbar(i)/(Qfeed*L);
            solQbar1 = (-b + sqrt(b^2-4*a*c))/(2*a);
            solQbar2 = (-b - sqrt(b^2-4*a*c))/(2*a);
            Qbar(i) = max(solQbar1,solQbar2);
        end
    end

    if adsorptionORpurge == 222
        Qbar(n) = Qin/Qfeed;
        for i = 1:n-1
            a = -1.75*Mg(i)*L*(1-epsilonbar)/(dp*rhogbar(i)*epsilonbar^2);
            b = 150*miu*L*(1-epsilonbar)^2/((dp^2)*Qfeed*rhogbar(i)*.......
                epsilonbar^2);
            c = epsilonbar*tfeed*(Pmax-Pmin)*dPbardzbar(i)/(Qfeed*L);
            solQbar1 = (-b + sqrt(b^2-4*a*c))/(2*a);
            solQbar2 = (-b - sqrt(b^2-4*a*c))/(2*a);
            Qbar(i) = min(solQbar1,solQbar2);
        end
    end
end
%...........................................................................
% Formulate usbar(i) = Qbar(i)/rhogbar(i)
%...........................................................................
if (ncall==0)       % initial condition
    for i = 1:n
        usbar(i) = Q0/Qfeed/rhogbar(i);
    end
end

if (ncall~=0)
    if adsorptionORpurge == 111
        i = 1;          % BC for adsorption step
        usbar(i) = Qin/Qfeed/rhogbar(i);

        for i = 2:n
            usbar(i) = Qbar(i)/rhogbar(i);
        end
    end

    if adsorptionORpurge == 222
        i = n;          % BC for purge step
        usbar(i) = Qin/Qfeed/rhogbar(i);

        for i = n-1:-1:1
            usbar(i) = Qbar(i)/rhogbar(i);
        end
    end
end
%...........................................................................
% Formulate gas phase axial dispersion coefficient in mass balance,
% DL, cm2/sec
%...........................................................................
if MassAxial == 881
    for i = 1:n
        DL(i) = 0;
    end
end

if MassAxial == 882
    if adsorptionORpurge == 111
        DM = Dem4;
    end

    if adsorptionORpurge == 222
        DM = Dem1;
    end

    for i = 1:n
        DL(i) = 0.7*DM + 0.5*dp*(abs(Qbar(i))*Qfeed)/(rhogbar(i)*..........
            (Qfeed*tfeed/L)*epsilonbar);
    end
end
%...........................................................................
% Formulate n1eqmbar(i), n2eqmbar(i) using isotherms onto LiX adsorbent
% from Rege and Yand 1997
%...........................................................................
for i = 1:n
      n1eqmbar(i) = (k1*rhob*L/Qfeed)*1.11*10^(-3)*exp(1593.0/.............
          ((Tmax-Tmin)*thetabar(i)+Tref))*(y1bar(i))   *((Pmax-Pmin)*.......
          Pbar(i)/1013250)/(1 + 1.03*10^(-4)*exp(2061.9/((Tmax-Tmin)*......
          thetabar(i)+Tref))*(y1bar(i))*((Pmax-Pmin)*Pbar(i)/1013250)......
          + 2.07*10^(-4)*exp(2455.5/((Tmax-Tmin)*thetabar(i)+Tref))*.......
          (1-y1bar(i))*((Pmax-Pmin)*Pbar(i)/1013250));

      n2eqmbar(i) = (k2*rhob*L/Qfeed)*1.25*10^(-3)*exp(2168.6/.............
          ((Tmax-Tmin)*thetabar(i)+Tref))*(1-y1bar(i))*((Pmax-Pmin)*.......
          Pbar(i)/1013250)/(1 + 1.03*10^(-4)*exp(2061.9/((Tmax-Tmin)*......
          thetabar(i)+Tref))*(y1bar(i))*((Pmax-Pmin)*Pbar(i)/1013250)......
          + 2.07*10^(-4)*exp(2455.5/((Tmax-Tmin)*thetabar(i)+Tref))*.......
          (1-y1bar(i))*((Pmax-Pmin)*Pbar(i)/1013250));
end
%..........................................................................
%..........................................................................
% Start of Superbee Flux Limiter
if dYbardZbarDifferentialMethod == 777
%..........................................................................
% This Flux Limiter formula is valid for any regularly or non-regularly
% spaced grid, and for any constant or varying advection velocity
% Dullemon, Numerical Fluid Dynamics - Lecture Notes, Chapter 4
%..........................................................................
% Using linear terms y1bar, thetabar, y2bar, and mass flux Q
%..........................................................................
% Formulate convective terms for i = 3:n-2
%..........................................................................
for i = 3:n-2
    % Le = left edge of cell
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    % Re = right edge of cell
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = (y1bar(i-1)      - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));
        ry2barLe     = (y2bar(i-1)      - y2bar(i-2))    /(y2bar(i)     -...
            y2bar(i-1));
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)      - y1bar(i))    /(y1bar(i)     -.....
            y1bar(i-1));
        ry2barLe     = (y2bar(i+1)      - y2bar(i))    /(y2bar(i)     -.....
            y2bar(i-1));
    end

    superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,     .............
        min(2,ry1barLe)));
    superbeey2barLe     = max(0 , max(min(1,2*ry2barLe) ,     .............
        min(2,ry2barLe)));

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*.......
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*..............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*.......
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*..............
        superbeey2barLe*    (y2bar(i)     - y2bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
            y1bar(i));
        ry2barRe     = (y2bar(i)     - y2bar(i-1))    /(y2bar(i+1)     -...
            y2bar(i));
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /..................
            (y1bar(i+1)     - y1bar(i));
        ry2barRe     = (y2bar(i+2)     - y2bar(i+1))    /..................
            (y2bar(i+1)     - y2bar(i));
    end

    superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,     .............
        min(2,ry1barRe)));
    superbeey2barRe     = max(0 , max(min(1,2*ry2barRe) ,     .............
        min(2,ry2barRe)));

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey2barRe*    (y2bar(i+1)     - y2bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
end
%..........................................................................
% Formulate convective terms for i = 1, 2, n-1, n
%..........................................................................
for i = 1
    Qbar_eps_Le = (Qbar(i) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = 1;
        ry2barLe     = 1;
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = 1;
        ry2barLe     = 1;
    end

    superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,     .............
        min(2,ry1barLe)));
    superbeey2barLe     = max(0 , max(min(1,2*ry2barLe) ,     .............
        min(2,ry2barLe)));

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*..........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)      - y1bar(i));

    fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y2bar(i)      + (1 - flowdirectionLe)*y2bar(i))      + 0.5*.........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey2barLe*    (y2bar(i)      - y2bar(i));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = 0;
        ry2barRe     = 0;
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = (y1bar(i+2)      - y1bar(i+1))    /..................
            (y1bar(i+1)      - y1bar(i));
        ry2barRe     = (y2bar(i+2)      - y2bar(i+1))    /..................
            (y2bar(i+1)      - y2bar(i));
    end

    superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,     .............
        min(2,ry1barRe)));
    superbeey2barRe     = max(0 , max(min(1,2*ry2barRe) ,     .............
        min(2,ry2barRe)));

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)      + (1 - flowdirectionRe)*y1bar(i+1))      + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)      - y1bar(i));

    fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y2bar(i)      + (1 - flowdirectionRe)*y2bar(i+1))      + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey2barRe*    (y2bar(i+1)      - y2bar(i));

    convecy1bar(i)      = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)      = (fluxy2barLe - fluxy2barRe)/dzbar;
end
%..........................................................................
for i = 2
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = 0;
        ry2barLe     = 0;
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)      - y1bar(i))    /(y1bar(i)      -.....
            y1bar(i-1));
        ry2barLe     = (y2bar(i+1)      - y2bar(i))    /(y2bar(i)      -.....
            y2bar(i-1));
    end

    superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,     .............
        min(2,ry1barLe)));
    superbeey2barLe     = max(0 , max(min(1,2*ry2barLe) ,     .............
        min(2,ry2barLe)));

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))      + 0.5*.......
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))      + 0.5*.......
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey2barLe*    (y2bar(i)     - y2bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = (y1bar(i)      - y1bar(i-1))    /(y1bar(i+1)     -...
            y1bar(i));
        ry2barRe     = (y2bar(i)      - y2bar(i-1))    /(y2bar(i+1)     -...
            y2bar(i));
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /(y1bar(i+1)    ...
            - y1bar(i));
        ry2barRe     = (y2bar(i+2)     - y2bar(i+1))    /(y2bar(i+1)    ...
            - y2bar(i));
    end

    superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,     .............
        min(2,ry1barRe)));
    superbeey2barRe     = max(0 , max(min(1,2*ry2barRe) ,     .............
        min(2,ry2barRe)));

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))      + 0.5*.......
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))      + 0.5*.......
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey2barRe*    (y2bar(i+1)     - y2bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
end
%..........................................................................
for i = n-1
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));
        ry2barLe     = (y2bar(i-1)     - y2bar(i-2))    /(y2bar(i)     -...
            y2bar(i-1));
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
            y1bar(i-1));
        ry2barLe     = (y2bar(i+1)     - y2bar(i))    /(y2bar(i)     -.....
            y2bar(i-1));
    end

    superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,     .............
        min(2,ry1barLe)));
    superbeey2barLe     = max(0 , max(min(1,2*ry2barLe) ,     .............
        min(2,ry2barLe)));

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey2barLe*    (y2bar(i)     - y2bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
            y1bar(i));
        ry2barRe     = (y2bar(i)     - y2bar(i-1))    /(y2bar(i+1)     -...
            y2bar(i));
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = 0;
        ry2barRe     = 0;
    end

    superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,     .............
        min(2,ry1barRe)));
    superbeey2barRe     = max(0 , max(min(1,2*ry2barRe) ,     .............
        min(2,ry2barRe)));

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey2barRe*    (y2bar(i+1)     - y2bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
end
%..........................................................................
for i = n
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));
        ry2barLe     = (y2bar(i-1)     - y2bar(i-2))    /(y2bar(i)     -...
            y2bar(i-1));
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = 0;
        ry2barLe     = 0;
    end

    superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,     .............
        min(2,ry1barLe)));
    superbeey2barLe     = max(0 , max(min(1,2*ry2barLe) ,     .............
        min(2,ry2barLe)));

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*.......
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*..............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*.......
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*..............
        superbeey2barLe*    (y2bar(i)     - y2bar(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = 1;
        ry2barRe     = 1;
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = 1;
        ry2barRe     = 1;
    end

    superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,     .............
        min(2,ry1barRe)));
    superbeey2barRe     = max(0 , max(min(1,2*ry2barRe) ,     .............
        min(2,ry2barRe)));

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i))     + 0.5*..........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*..............
        superbeey1barRe*    (y1bar(i)     - y1bar(i));

    fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y2bar(i)     + (1 - flowdirectionRe)*y2bar(i))     + 0.5*..........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*..............
        superbeey2barRe*    (y2bar(i)     - y2bar(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
end
%..........................................................................
end % end of dYbardZbarDifferentialMethod == 777
%..........................................................................
%..........................................................................
% Formulate ODEs - can be used for adsorption and purge - both directions
%..........................................................................
for i=1
```

```
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon +.....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y1bar(i+2)-2*Y1bar(i+1)....
        +Y1bar(i))/(dzbar)^2;

    dY2bardtbar(i) = convecy2bar(i) - (n2eqmbar(i)-n2bar(i))/epsilon +.....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y2bar(i+2)-2*Y2bar(i+1)....
        +Y2bar(i))/(dzbar)^2;
end
for i=2:n-1
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon +.....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y1bar(i+1)-2*Y1bar(i)+.....
        Y1bar(i-1))/(dzbar)^2;

    dY2bardtbar(i) = convecy2bar(i) - (n2eqmbar(i)-n2bar(i))/epsilon +.....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y2bar(i+1)-2*Y2bar(i)+.....
        Y2bar(i-1))/(dzbar)^2;
end
for i=n
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon +.....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y1bar(n)-2*Y1bar(n-1)+.....
        Y1bar(n-2))/(dzbar)^2;

    dY2bardtbar(i) = convecy2bar(i) - (n2eqmbar(i)-n2bar(i))/epsilon +.....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y2bar(n)-2*Y2bar(n-1)+.....
        Y2bar(n-2))/(dzbar)^2;
end
%..........................................................................
if MassAxial == 882 % Turn on mass axial dispersion
if adsorptionORpurge == 111
    % 2nd BC for d2YdZ2 at zbar = 1 : dY/dz = 0 (wrong); just eliminate
    % d2YdZ2 from the complete equation (Schiesser)
    for i=n
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon + 0;
    dY2bardtbar(i) = convecy2bar(i) - (n2eqmbar(i)-n2bar(i))/epsilon + 0;
    end
end
if adsorptionORpurge == 222
    % 2nd BC for d2YdZ2 at zbar = 1 : dY/dz = 0 (wrong); just eliminate
    % d2YdZ2 from the complete equation (Schiesser)
    for i=1
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon + 0;
    dY2bardtbar(i) = convecy2bar(i) - (n2eqmbar(i)-n2bar(i))/epsilon + 0;
    end
end
end
%..........................................................................
for i=1:n
      dn1bardtbar(i) = k1*tfeed*(n1eqmbar(i)-n1bar(i));
      dn2bardtbar(i) = k2*tfeed*(n2eqmbar(i)-n2bar(i));
end
%..........................................................................
% Six vectors into one vector
%..........................................................................
for i = 1:n
    Yt(0*n+i) = dY1bardtbar(i);
    Yt(1*n+i) = dn1bardtbar(i);
    Yt(2*n+i) = dn2bardtbar(i);
    %Yt(3*n+i) = dThetabargdtbar(i);
    %Yt(4*n+i) = dThetabarsdtbar(i);
    Yt(3*n+i) = dY2bardtbar(i);
```

```
end

if adsorptionORpurge == 111
    i = 1;
    Yt(0*n+i) = 0;     % because Y1bar(1) = y1feed*rhog(1)
    %Yt(3*n+i) = 0;    % because Thetabarg(1) = Tfeed*rhog(1)
    %Yt(4*n+i) = 0;    % because Thetabars(1) = Tfeed not
    Yt(3*n+i) = 0;     % because Y2bar(1) = y2feed*rhog(1)
end
if adsorptionORpurge == 222
    i = n;
    Yt(0*n+i) = 0;     % because Y1bar(n) = y1feed*rhog(n)
    %Yt(3*n+i) = 0;     % because Thetabarg(n) = Tfeed*rhog(n)
    %Yt(4*n+i) = 0;     % because Thetabars(n) = Tfeed not
    Yt(3*n+i) = 0;     % because Y2bar(n) = y2feed*rhog(n)
end
%.........................................................................
  yt=Yt';

% Increment calls to psa_1
  ncall=ncall+1;
%.........................................................................
```

Appendix 5

Models E, F (with or without gas thermal axial dispersion depending on the Nusselt correlation used)

- **Model E**, an adsorption (`adsorptionORpurge = 111`) or desorption (`adsorptionORpurge = 222`) step that includes **gas-solid adsorption kinetic, gas phase mass axial dispersion**, **pressure drop**, **non-isothermal** (**adiabatic, equilibrium gas-solid heat transfer kinetic)**, is modeled in this routine by setting (`%k1 = 50000, %k2 = 50000`), (`MassAxial = 882`), (`HeateqmORnoneqm = 994`), (`HeatAxial = 991 or 992`).

- **Model F**, an adsorption (`adsorptionORpurge = 111`) or desorption (`adsorptionORpurge = 222`) step that includes **gas-solid adsorption kinetic, gas phase mass axial dispersion**, **pressure drop**, **non-isothermal** (**adiabatic, non-equilibrium gas-solid heat transfer kinetic)**, is modeled in this routine by setting (`%k1 = 50000, %k2 = 50000`), (`MassAxial = 882`), (`HeateqmORnoneqm = 993`), (`HeatAxial = 991 or 992`).

- One setting that especially needs your attention is (`nout, n`). This combination must be set properly to achieve numerical stability and convergence.

- Three spatial differential methods are available in this routine (`dYbardZbarDifferentialMethod = 333, or 444, or 777`).

Main Routine

```
%..........................................................................
% File: psa_main_sequence_model_EF
%..........................................................................
% Chai Siew Wah, Chemical Engineering, Lehigh University, May 2011
%..........................................................................
% MATHEMATICAL MODEL OF PRESSURE SWING ADSORPTION SYSTEM FOR ONE STEP :
% ADSORPTION OR PURGE
%..........................................................................
%
%                  :......................................:
%   ADSORPTION --> :                                      : --> PRODUCT
%   ADSORPTION --> :                                      : --> PRODUCT
%   ADSORPTION --> :                                      : --> PRODUCT
%                  :......................................:
%                 i=1                                    i=n
%
%                  :......................................:
%     EXHAUST <--  :                                      : <-- PURGE
%     EXHAUST <--  :                                      : <-- PURGE
%     EXHAUST <--  :                                      : <-- PURGE
%                  :......................................:
%                 i=1                                    i=n
%..........................................................................
```

```
% Assumptions
%..........................................................................
% (1) Ideal gas law
% (2) Empirical Langmuirian type binary isotherms - from Rege and Yang 1997
% (3) Nonisobaric - Ergun equation solved analytically
% (4) Nonisothermal - adiabatic
% (5) Linear driving force model for mass transfer kinetic
% (6) Mass axial dispersion in gas phase
% (7) Wakao correlation for gas-solid heat transfer kinetic +
%     gas phase thermal axial dispersion
% (8) KuniiSuzuki correlation for gas-solid heat transfer kinetic without
%     gas phase thermal axial dispersion
% (9)  Absence of solid phase thermal axial conduction
% (10) Absence of radial distribution of mass and heat
% (11) Absence of gas mal-distribution or particle agglomeration
%..........................................................................
% 6 ordinary differential equations : y1, y2, n1, n2, Tg, Ts
% 1 direct calculation for          : rhogbar (= Y1bar + Y2bar)
% 1 algebraic equation for          : Q (= Ergun equation)
% 1 algebraic equation for          : P (= Ideal gas law)
% 1 calculated for                  : us (= Q/rhog) - but reduce its use
% Solve for y1, y2, n1, n2, Tg, Ts, P, rhog, Q, us
%..........................................................................
% (1) Solve Ergun equation for Qbar and properly dealing its positive or
% negative flow direction

% (2) Use Schiesser's boundary condition for mass and heat balances that
% have axial dispersion terms (retain the whole equation except the 2nd
% derivative of space) instead of Danwerts' boundary condition of dY/dz = 0
% at exit

% (3) Eliminate overall mass balance, use 2nd component mass balance, so
% that I do not need a boundary condition for rhog (there is no defined
% rhog at inlet or exit), and I can use flux conservation formulation of
% Q throughout, and rhogbar = Y1bar + Y2bar.

% (4) Gas compositions y1feed y2feed are specified, rhogbar(feed end) =
% Y1bar + Y2bar is not. But I need the right BCs for Y1bar, Y2bar,
% Thetabarg, which depend on rhogbar(feed end).

% (5) Normally Pbar(i) is calculated from rhogbar(i) using ideal gas law.
% But at feed end, rhogbar(feed end) is undetermined. So, Pressure comes
% in to help by using its previous dP/dz and Pbar to calculate
% Pbar(feed end), then calculate rhogbar(feed end) from ideal gas law,
% then calculate the right BCs for Y1bar, Y2bar, Thetabarg.

% (6) Correct use of: epsilon (gas phase accumulation), epsilonbar(Ergun,
% axial dispersions), mass and thermal axial dispersions, gas-solid heat
% transfer (Kunii/Wakao) from literature values (Rege and Yang 1997
% provides Cs, Cg, q1, q2, isotherms), adsorbent density from bed
% dimensions and 160 g LiX in Invacare test unit.

% (7) Only allow average velocity,uz = us/epsilon, and mass flux, Q, to
% acquire positive or negative value, all the rest, y1, y2, n1, n2, Tg, Ts,
% P, rhog, must use positive values
%..........................................................................
% Clear previous files
clc
clear all
format short
```

```
%...........................................................................
% Parameters shared with the ODE routine
%...........................................................................
global  Mg1 Mg2 adsorptionORpurge dYbardZbarDifferentialMethod T0......
    ncall n dzbar dtbar miu epsilon epsilonbar k1 k2 q1 q2 Cs Cg rhob R....
    dp L tfeed y1feed y2feed Qin Qfeed P0 Pout Tfeed Pmax Pmin Tmax Tmin...
    Tref Q0 kg MassAxial HeatAxial HeateqmORnoneqm Dem4 Dem1
%...........................................................................
% Select process step
%...........................................................................
adsorptionORpurge = 222;        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% adsorptionORpurge = 111;      % 111 for adsorption
% adsorptionORpurge = 222;      % 222 for purge
%...........................................................................
% Assign positive or negative sign to feed gas mass flux (Q) and average
% velocity (uz = us/epsilon) at boundary (feed) and initial condition
% (which are within our control). Do not assign +ve / -ve to dynamic
% Qbar(i) and usbar(i), which are calculated by the program
%...........................................................................
if adsorptionORpurge == 111
    FE = +1;    % for mass flux at feed end i = 1
end
if adsorptionORpurge == 222
    FE = -1;    % for mass flux at feed end i = n
end
%...........................................................................
% Select differential method for dYbardZbar
%...........................................................................
dYbardZbarDifferentialMethod = 777;   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% dYbardZbarDifferentialMethod = 333  % 1st order (2 points) upwind differe
% dYbardZbarDifferentialMethod = 444; % 2nd order (3 points) upwind differe
% dYbardZbarDifferentialMethod = 777; % 2nd order TVD Superbee Flux Limiter
%...........................................................................
% Select Mass and Heat Axial Dispersions, Gas-Solid Heat Transfer
% Equilibrium or Nonequilibrium
%...........................................................................
MassAxial = 882;       %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% MassAxial = 881;     % Shut off mass axial dispersion
% MassAxial = 882;     % Turn on mass axial dispersion, need 2nd BC
HeateqmORnoneqm = 993;
% HeateqmORnoneqm = 993;    % use (Kunni and Suzuki) or (Wakao) h
% HeateqmORnoneqm = 994;    % equilibrium gas-solid heat transfer
HeatAxial = 992;
% HeatAxial = 991;     % Kunii and Suzuki, no gas thermal axial dispersion
% HeatAxial = 992;     % Wakao et al., with gas thermal axial dispersion
%...........................................................................
% Select type of plots
%...........................................................................
choose = 4;  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% choose = 1 : 2D subplots of y1, y2, n1, n2, T, P
% choose = 2 : Individual 2D plots of y1, y2, n1, n2, T, P, rhog, Q, us
% choose = 3 : Surface plots of y1, y2, n1, n2, T, P, rhog, Q, us
% choose = 4 : The 2D plots that Dr. Sircar is interested in
%...........................................................................
% Define number of nodes along t and along z
%...........................................................................
nout = 1001;        % nodes along t  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
n = 151;            % nodes i=   01 - 101 along z direction for y1bar
                    % nodes i=  102 - 202 along z direction for n1bar
                    % nodes i=  203 - 303 along z direction for n2bar
                    % nodes i=  304 - 404 along z direction for thetabarg
                    % nodes i=  405 - 505 along z direction for thetabars
                    % nodes i=  506 - 606 along z direction for y2bar
                    % Above 6 dependent variables are solved by ODE routine
                    % rhogbar = Y1bar + Y2bar
                    % Pbar solved by Ideal gas law, but not at feed end
                    % Qbar solved by Ergun equation
%..........................................................................
% Define dimensionless z and dz
%..........................................................................
zbarl = 0.0;                    % zbar lower limit, zbar = 0/L
zbaru = 1.0;                    % zbar upper limit, zbar = L/L
dzbar = (zbaru-zbarl)/(n-1);    % zbar differential space
zbar  = linspace(zbarl,zbaru,n);
%..........................................................................
% Define dimensionless t and dt
% Independent variable, tbar, for ODE integration
%..........................................................................
tbarl = 0.0;                    % tbar lower limit, tbar = 0/tfeed
tbaru = 1.0;                    % tbar upper limit, tbar = tfeed/tfeed
dtbar = (tbaru-tbarl)/(nout-1); % tbar differential space
tbar  = linspace(tbarl,tbaru,nout); % horizontal tbar
tbarv = tbar';                      % vertical tbar
%..........................................................................
% Gas properties
%..........................................................................
R = 8.314472*10^4;      % gas constant, g.cm2/K/mmol/sec2
Cg = 0.00687;           % gas phase heat capacity, cal/mmol/K, Rege&Yang 97
%kg = 6.19e-5;          % gas phase (N2 at 1 atm 300K) thermal conductivity
                        % , cal/cm/sec/K
%Cg = 0.00721;          % gas phase (Air at 300K) heat capacity, cal/mmol/K
kg = 6.44e-5;           % gas phase (Air at 1 atm 295K) thermal
                        % conductivity, cal/cm/sec/K
miu = 18.2385e-5;       % air dynamic viscosity at 25C and 1 atm, g/cm/sec
Mg1 = 0.032;            % O2 molecular weight, g/mmol
Mg2 = 0.028;            % N2 molecular weight, g/mmol
%..........................................................................
% Packed bed dimensions
%..........................................................................
% Conventional adsorber
L = 8.382;              % bed length, cm (3.3 in = 8.382 cm)
D = 6.0325;             % bed diameter, cm (2 3/8 in)

% Pancake adsorber
%L = 2.12;              % bed length, cm (3.3 in = 8.382 cm)
%D = 12;                % bed diameter, cm (2 3/8 in)

A = pi*(D/2)^2;         % bed cross-sectional area, cm2
rhob = 0.668;           % bulk density, g/cm3, according to 160 g LiX
%rhob = 0.632;          % bulk density, g/cm3, at 1/3 lb (151.33 g) LiX
%..........................................................................
% Adsorbent properties
%..........................................................................
dp = 0.0200;            % particle diameter, cm   %%%%%%%%%%%%%%%%%%%%%%%%%
epsilon = 0.61;         % total column (helium) void fraction= ep x (1-e)+e
epsilonp = 0.35;        % internal/intraparticle void fraction
epsilonbar = 0.40;      % external/interparticle void fraction, bed voidage
                        % , Rege and Yang 1997
taup = 3;               % particle tortuosity factor
Cs = 0.28;              % adsorbent (solid phase) heat capacity, cal/g/K,
                        % Rege and Yang 1997
ks = 4.78e-4;           % dry sand thermal conductivity, 298K, cal/cm/sec/K
%..........................................................................
% Mass transfer kinetic and heat properties
%..........................................................................
Dek1 = 0.0307;          % O2 effective knudsen diffusivity in Zeochem LiLSX
                        % at 1-4 atm, cm2/sec, Todd and Webley 2006
Dek2 = 0.03277;         % N2 effective knudsen diffusivity in Zeochem LiLSX
                        % at 1-4 atm, cm2/sec, Todd and Webley 2006
Dem1 = 0.0348;          % effective molecular diffusivity in Zeochem LiLSX
                        % at 1 atm, cm2/sec, Todd and Webley 2006
Dem4 = 0.008707;        % effective molecular diffusivity in Zeochem LiLSX
                        % at 4 atm, cm2/sec, Todd and Webley 2006

if adsorptionORpurge == 111
    De1 = (epsilonp/taup)*(1/(1/Dek1 + 1/Dem4));
    De2 = (epsilonp/taup)*(1/(1/Dek2 + 1/Dem4));
end
if adsorptionORpurge == 222
    De1 = (epsilonp/taup)*(1/(1/Dek1 + 1/Dem1));
    De2 = (epsilonp/taup)*(1/(1/Dek2 + 1/Dem1));
end
k1 = 15*De1/(dp/2)^2;   % O2 overall mass transfer coefficient, sec-1
k2 = 15*De2/(dp/2)^2;   % N2 overall mass transfer coefficient, sec-1
q1 = 3.16;              % isosteric heat of adsorption of O2 on LiX, cal/
                        % mmol, Rege and Yang 1997
q2 = 5.60;              % isosteric heat of adsorption of N2 on LiX, cal/
                        % mmol, Rege and Yang 1997
%k1 = 500;              % equilibrium O2 overall mass transfer rate, sec-1
%k2 = 500;              % equilibrium N2 overall mass transfer rate, sec-1
%..........................................................................
% Feed conditions at i = 1 (adsorption)
%                 at i = n (purge)
%..........................................................................
if adsorptionORpurge == 111
    y1feed = 0.21;      % feed gas composition, 21% O2 + 79% N2
    y2feed = 0.79;      % feed gas composition, 21% O2 + 79% N2
    Tfeed = 298;        % feed temperature, K
    tfeed = 2;          % feed step time, sec
    qfeed = 300/tfeed;  % feed mass flow rate, mmol/sec
    Pout = 4.0*1013250; % exit pressure at i=n during adsorption, g/cm/sec2
end
if adsorptionORpurge == 222
    y1feed = 1.00;      % purge gas composition, 100% O2 + 0% N2
    y2feed = 0.00;      % purge gas composition, 100% O2 + 0% N2
    Tfeed = 298;        % purge temperature, K
    tfeed = 2;          % purge step time, sec
    qfeed = 306/tfeed;  % purge mass flow rate, mmol/sec
    Pout = 1.0*1013250; % outlet pressure at i=1 during purge, g/cm/sec2
end
Qfeed = qfeed/A;        % mass flux, mmol/cm2/sec
Qin = FE * Qfeed;       % assign directions for the absolute value of Qfeed
% usfeed = Qfeed/rhog;  % superficial velocity, cm/sec
% I should not specify Pfeed because I need to find out the pressure drop
% starting from Pout using Ergun equation in momentum balance.
% Feed gas density is unknown because rhogfeed = Pfeed/(R*Tfeed) mmol/cm3
% Equilibrium capacity at feed inlet (i=1) or (i=n) is unknown bcoz Pfeed
%..........................................................................
% Bed initial conditions
%..........................................................................
```

```
if adsorptionORpurge == 111
    y10 = 1.00;         % saturated with 100% O2 + 0% N2
    y20 = 0.00;         % saturated with 100% O2 + 0% N2
    P0 = 3.0*1013250;   % pressure after pressurization step, g/cm/sec2
    T0 = 298;           % temperature after pressurization step, K
end
if adsorptionORpurge == 222
    y10 = 0.21;         % saturated with 21% O2 + 79% N2
    y20 = 0.79;         % saturated with 21% O2 + 79% N2
    P0 = 1.0*1013250;   % pressure after depressurization step, g/cm/sec2
    T0 = 298;           % temperature after depressurization step, K
end

Q0 = FE * 0;            % without gas flux, mmol/cm2/sec
rhog0 = P0/(R*T0);      % gas phase density, mmol/cm3
us0 = Q0/rhog0;         % superficial velocity, cm/sec
n1eqm0 = 1.11*10^(-3)*exp(1593.0/T0)*(y10)  *(P0/1013250)/(1 + 1.03*10^....
    (-4)*exp(2061.9/T0)*(y10)*(P0/1013250) + 2.07*10^(-4)*exp(2455.5/T0)...
    *(1-y10)*(P0/1013250));
n2eqm0 = 1.25*10^(-3)*exp(2168.6/T0)*(1-y10)*(P0/1013250)/(1 + 1.03*10^....
    (-4)*exp(2061.9/T0)*(y10)*(P0/1013250) + 2.07*10^(-4)*exp(2455.5/T0)...
    *(1-y10)*(P0/1013250));
n10 = n1eqm0;           % O2 adsorbed, mmol O2/g
n20 = n2eqm0;           % N2 adsorbed, mmol N2/g
% Langmuir isotherms of O2 and N2 on LiX from Salil U. Rege and Ralph T.
% Yang, Ind. Eng. Chem. Res. 1997, 36, 5358-5365.
Mg0 = Mg1*y10 + Mg2*(1-y10); % average gas phase molecular weight, g/mmol
%...........................................................................
% Important note: variables must be parameterized within the minimum and
% maximum range in order to avoid singular matrix of A.x = b
%...........................................................................
% Additional parameters required for proper nondimensionalization of
% dependent variables
%...........................................................................
Pmax = Pout;            % maximum pressure, g/cm/sec2
Pmin = 0;               % minimum pressure, g/cm/sec2

Tref = T0;              % reference temperature, K
Tmax = Tfeed;           % maximum temperature, K
Tmin = 273;             % minimum temperature, K
%...........................................................................
% Nonlinear variables
%...........................................................................
% Y1bar = y1bar * rhogbar
% Thetabarg = thetabarg * rhogbar
% Y2bar = y2bar * rhogbar
%...........................................................................
% Initial conditions for y1bar, n1bar, n2bar, thetabarg, thetabars, y2bar
% for i = 1:n
%...........................................................................
% Note that boundary conditions at i = 1 (adsorption) or i = n (purge)
% after t=0 should not be brought in here because BC should be dealt with
% in the Method of Lines routine
%...........................................................................
for i=1:n
    Y0(0*n+i) = y10*rhog0*L/(Qfeed*tfeed);       % for Y1bar
    Y0(1*n+i) = (k1*rhob*L/Qfeed)*n10;           % for n1bar
    Y0(2*n+i) = (k2*rhob*L/Qfeed)*n20;           % for n2bar
    Y0(3*n+i) = ((T0-Tref)/(Tmax-Tmin))*rhog0*L/(Qfeed*tfeed); % Thetabarg
    Y0(4*n+i) = ((T0-Tref)/(Tmax-Tmin));                        % Thetabars
    Y0(5*n+i) = y20*rhog0*L/(Qfeed*tfeed);      % for Y2bar
end
%..........................................................................
% ODE integration
%..........................................................................
ncall = 0;
reltol = 1.0e-04;
abstol = 1.0e-04;
options = odeset('RelTol',reltol,'AbsTol',abstol);

[t,Y] = ode15s(@psa_1_sequence_model_EF, tbar, Y0, options);
% Embedded in MOL routine are 1st order (two-points) and 2nd order (three-
% points) upwind finite volume methods, and 2nd order TVD Superbee Flux
% Limiter to discretize dYbardZbar
%..........................................................................
% One vector to six vectors
%..........................................................................
for it = 1:nout     % time
for i = 1:n         % position
    Y1bar(it,i)     = Y(it,0*n+i);
    n1bar(it,i)     = Y(it,1*n+i);
    n2bar(it,i)     = Y(it,2*n+i);
    Thetabarg(it,i) = Y(it,3*n+i);
    Thetabars(it,i) = Y(it,4*n+i);
    Y2bar(it,i)     = Y(it,5*n+i);
end
end

for it = 1:nout
for i = 1:n
    y1bar(it,i)    = Y1bar(it,i)/(Y1bar(it,i)+Y2bar(it,i));
    y2bar(it,i)    = Y2bar(it,i)/(Y1bar(it,i)+Y2bar(it,i));
    rhogbar(it,i)  = Y1bar(it,i) + Y2bar(it,i);
    Mg(it,i)       = Mg1*y1bar(it,i) + Mg2*(1-y1bar(it,i));
    thetabarg(it,i)= Thetabarg(it,i)/rhogbar(it,i);
    thetabars(it,i)= Thetabars(it,i);
end
end
%..........................................................................
% Fix the boundary conditions
%..........................................................................
% ODE solver only solves equations of dY/dt (without dY/dz in it), boundary
% conditions are not cared for. Though boundary conditions were forced in
% MOL routine, ODE solver only stores the solutions of Y calculated from
% governing ODEs and previous Y, boundary conditions at i=1 (adsorption) or
% i=n (purge) are not stored in ODE solver.
%..........................................................................
% ODE solver does take care of initial conditions at it = 1
for it = 2:nout     % time

if adsorptionORpurge == 111
   i = 1;
end
if adsorptionORpurge == 222
   i = n;
end

y1bar(it,i)         = y1feed;
y2bar(it,i)         = y2feed;
Y1bar(it,i)         = y1feed*rhogbar(it,i);
```

```
Y2bar(it,i)         = y2feed*rhogbar(it,i);
thetabarg(it,i)     = (Tfeed-Tref)/(Tmax-Tmin);
Thetabarg(it,i)     = ((Tfeed-Tref)/(Tmax-Tmin))*rhogbar(it,i);
%thetabars(it,i)    = (Tfeed-Tref)/(Tmax-Tmin);
%Thetabars(it,i)    = (Tfeed-Tref)/(Tmax-Tmin);
end
%...........................................................................
% Formulate Pbar(i) using ideal gas law except that at inlet point
%...........................................................................
for it = 1              % initial condition
    for i=1:n           % position
        Pbar(it,i) = P0/(Pmax-Pmin);
    end
end

for it = 2:nout         % time
%...........................................................................
    if adsorptionORpurge == 111
        i = n;          % BC for adsorption step
        Pbar(it,i) = Pout/(Pmax-Pmin);

        for i = n-1:-1:2
            Pbar(it,i) = rhogbar(it,i)*(thetabarg(it,i)*(Tmax-Tmin)+.......
                Tref)*(R*Qfeed*tfeed)/((Pmax-Pmin)*L);
            dPbardzbar(it,i) = (Pbar(it,i+1)-Pbar(it,i))/dzbar;
        end

        for i = 1
            Pbar(it,i)       = Pbar(it,i+1) - dPbardzbar(it,i+1)*dzbar;
            rhogbar(it,i)    = Pbar(it,i)*(Pmax-Pmin)*L/((.................
                thetabarg(it,i)*(Tmax-Tmin)+Tref)*R*Qfeed*tfeed);
            dPbardzbar(it,i) = (Pbar(it,i+1)-Pbar(it,i))/dzbar;
            Y1bar(it,i)      = y1feed*rhogbar(it,i);
            Y2bar(it,i)      = y2feed*rhogbar(it,i);
            Thetabarg(it,i)  = ((Tfeed-Tref)/(Tmax-Tmin))*rhogbar(it,i);
        end

        for i = n           % special upwinding
            dPbardzbar(it,i) = (Pbar(it,i)-Pbar(it,i-1))/dzbar;
        end
    end
%...........................................................................
    if adsorptionORpurge == 222
        i = 1;          % BC for purge step
        Pbar(it,i) = Pout/(Pmax-Pmin);

        for i = 2:n-1
            Pbar(it,i) = rhogbar(it,i)*(thetabarg(it,i)*(Tmax-Tmin)+.......
                Tref)*(R*Qfeed*tfeed)/((Pmax-Pmin)*L);
            dPbardzbar(it,i) = (Pbar(it,i)-Pbar(it,i-1))/dzbar;
        end

        for i = n
            Pbar(it,i)       = dPbardzbar(it,i-1)*dzbar + Pbar(it,i-1);
            rhogbar(it,i)    = Pbar(it,i)*(Pmax-Pmin)*L/((.................
                thetabarg(it,i)*(Tmax-Tmin)+Tref)*R*Qfeed*tfeed);
            dPbardzbar(it,i) = (Pbar(it,i)-Pbar(it,i-1))/dzbar;
            Y1bar(it,i)      = y1feed*rhogbar(it,i);
            Y2bar(it,i)      = y2feed*rhogbar(it,i);
            Thetabarg(it,i)  = ((Tfeed-Tref)/(Tmax-Tmin))*rhogbar(it,i);
        end

        for i = 1           % special upwinding
            dPbardzbar(it,i) = (Pbar(it,i+1)-Pbar(it,i))/dzbar;
        end
    end
%..........................................................................
end
%..........................................................................
% Solving for usbar(i) in Ergun equation using quadratic formula - failed
% Solving for Qbar(i) in Ergun equation using quadratic formula
%..........................................................................
for it = 1              % initial condition
    for i = 1:n
        Qbar(it,i) = Q0/Qfeed;
    end
end

for it = 2:nout         % time
    if adsorptionORpurge == 111
        Qbar(it,1) = Qin/Qfeed;
        for i = 2:n
            a = 1.75*Mg(it,i)*L*(1-epsilonbar)/(dp*rhogbar(it,i)*..........
                epsilonbar^2);
            b = 150*miu*L*(1-epsilonbar)^2/((dp^2)*Qfeed*rhogbar(it,i)*....
                epsilonbar^2);
            c = epsilonbar*tfeed*(Pmax-Pmin)*dPbardzbar(it,i)/(Qfeed*L);
            solQbar1 = (-b + sqrt(b^2-4*a*c))/(2*a);
            solQbar2 = (-b - sqrt(b^2-4*a*c))/(2*a);
            Qbar(it,i) = max(solQbar1,solQbar2);
        end
    end

    if adsorptionORpurge == 222
        Qbar(it,n) = Qin/Qfeed;
        for i = 1:n-1
            a = -1.75*Mg(it,i)*L*(1-epsilonbar)/(dp*rhogbar(it,i)*.........
                epsilonbar^2);
            b = 150*miu*L*(1-epsilonbar)^2/((dp^2)*Qfeed*rhogbar(it,i)*....
                epsilonbar^2);
            c = epsilonbar*tfeed*(Pmax-Pmin)*dPbardzbar(it,i)/(Qfeed*L);
            solQbar1 = (-b + sqrt(b^2-4*a*c))/(2*a);
            solQbar2 = (-b - sqrt(b^2-4*a*c))/(2*a);
            Qbar(it,i) = min(solQbar1,solQbar2);
        end
    end
end
%..........................................................................
% Formulate usbar(i) = Qbar(i)/rhogbar(i)
%..........................................................................
for it = 1              % initial condition
    for i=1:n           % position
        usbar(it,i) = Q0/Qfeed/rhogbar(it,i);
    end
end

for it = 2:nout
    if adsorptionORpurge == 111
        i = 1;          % BC for adsorption step
        usbar(it,i) = Qin/Qfeed/rhogbar(it,i);
        for i = 2:n
            usbar(it,i) = Qbar(it,i)/rhogbar(it,i);
        end
    end

    if adsorptionORpurge == 222
        i = n;          % BC for purge step
        usbar(it,i) = Qin/Qfeed/rhogbar(it,i);

        for i = n-1:-1:1
            usbar(it,i) = Qbar(it,i)/rhogbar(it,i);
        end
    end
end
%...........................................................................
% Mass flux
%...........................................................................
AccQ = 0;
for i = 1:n
        AccQ = AccQ + Qfeed*abs(Qbar(nout,i));
end
AveQ_nout_mmolecm2sec = AccQ/n
AveQ_nout_lbmoleft2hr = AveQ_nout_mmolecm2sec*3600*929.0304/454000
%...........................................................................
% Formulate gas phase axial dispersion coefficient in mass balance,
% DL, cm2/sec
%...........................................................................
if MassAxial == 881
for i = 1:n
    DL(nout,i) = 0;
end
end

if MassAxial == 882
if adsorptionORpurge == 111
    DM = Dem4;
end
if adsorptionORpurge == 222
    DM = Dem1;
end
for i = 1:n
    DL(nout,i) = 0.7*DM + 0.5*dp*(abs(Qbar(nout,i))*Qfeed)/(................
        rhogbar(nout,i)*(Qfeed*tfeed/L)*epsilonbar);
end
end

AccDL = 0;
for i = 1:n
    AccDL = AccDL + DL(nout,i);
end
AveDL_nout_cm2sec = AccDL/n
%...........................................................................
% Formulate volumetric gas-solid heat transfer coefficient, ha
%...........................................................................
for it = 1:nout         % time

if HeatAxial == 991  % Kunni and Suzuki
for i = 1:n
    if HeateqmORnoneqm == 993
```

```
        % Kunii and Suzuki 1967, for packed bed, cal/cm2/sec/K
        h = (0.032*((Cg*miu/(Mg(it,i)*kg))*(Qfeed*abs(Qbar(it,i))*dp*......
            Mg(it,i)/miu))^(1.5))*(kg/dp);
        Nu(it,i) = (0.032*((Cg*miu/(Mg(it,i)*kg))*(Qfeed*abs(Qbar(it,i))...
            *dp*Mg(it,i)/miu))^(1.5));
        % cal/cm3/sec/K, assuming sphere particle, a = available surface
        % area/bed volume
        ha(it,i) = 6*(1-epsilonbar)*h/dp;
    end
    if HeateqmORnoneqm == 994
        ha(it,i) = 2000; % for equilibrium gas-solid heat transfer
        h(it,i) = ha(it,i)*dp/(6*(1-epsilonbar));
        Nu(it,i) = h(it,i)*dp/kg;
    end
%.........................................................................
% Formulate thermal axial dispersion coefficient, Dg, Ds, cm2/sec
%.........................................................................
        Dg(it,i) = 0; % Gas phase thermal diffusivity, cm2/sec
        % Ds = ks/(rhop*Cs);  % Solid phase thermal diffusivity, cm2/sec
        Ds = 0;               % Absence of solid phase thermal diffusivity
%.........................................................................
end
end
%.........................................................................
if HeatAxial == 992  % Wakao
for i = 1:n
    if HeateqmORnoneqm == 993
        % Wakao 1979, for packed bed, need to use axial dispersion,
        % cal/cm2/sec/K
        h = (2+1.1*((Cg*miu/(Mg(it,i)*kg))^(1/3))*((Qfeed*abs(............
            Qbar(it,i))*dp*Mg(it,i)/miu)^(0.6)))*kg/dp;
        Nu(it,i) = (2+1.1*((Cg*miu/(Mg(it,i)*kg))^(1/3))*((Qfeed*..........
            abs(Qbar(it,i))*dp*Mg(it,i)/miu)^(0.6)));
        ha(it,i) = 6*(1-epsilonbar)*h/dp;
%.........................................................................
% Formulate thermal axial dispersion coefficient, Dg, Ds, cm2/sec
%.........................................................................
        % Gas phase thermal diffusivity, cm2/sec
        Dg(it,i) = kg/(rhogbar(it,i)*(Qfeed*tfeed/L)*Cg);
        % Ds = ks/(rhop*Cs);  % Solid phase thermal diffusivity, cm2/sec
        Ds = 0;               % Absence of solid phase thermal diffusivity
%.........................................................................
    end
    if HeateqmORnoneqm == 994
        ha(it,i) = 2000; % for equilibrium gas-solid heat transfer
        h(it,i) = ha(it,i)*dp/(6*(1-epsilonbar));
        Nu(it,i) = h(it,i)*dp/kg;
%.........................................................................
% Formulate thermal axial dispersion coefficient, Dg, Ds, cm2/sec
%.........................................................................
        Dg(it,i) = 0; % Gas phase thermal diffusivity, cm2/sec
        % Ds = ks/(rhop*Cs);  % Solid phase thermal diffusivity, cm2/sec
        Ds = 0;               % Absence of solid phase thermal diffusivity
%.........................................................................
    end
end
end
end

Accha = 0;
```

```
AccNu = 0;
AccDg = 0;
for i = 1:n
        Accha = Accha + ha(nout,i);
        AccNu = AccNu + Nu(nout,i);
        AccDg = AccDg + Dg(nout,i);
end
Aveha_nout_calcm3secK = Accha/n
AveNu_nout = AccNu/n
AveDg_nout_cm2sec = AccDg/n
Ds = 0
%..........................................................................
% Calculate Reynolds number of particle
%..........................................................................
AccReynolds = 0;
for i=1:n
    Reynolds(i) = (Qfeed*abs(Qbar(nout,i))*dp*Mg(nout,i))/miu;
    AccReynolds = AccReynolds + Reynolds(i);
end
AveReynolds_nout = AccReynolds/n
%..........................................................................
% Calculate pressure drop across the bed
%..........................................................................
if adsorptionORpurge == 111
Pressuredropacrossbed_psi = ((Pmax-Pmin).*Pbar(nout,1) - Pout)*14.69595/...
    1013250;
end

if adsorptionORpurge == 222
Pressuredropacrossbed_psi = ((Pmax-Pmin).*Pbar(nout,n) - Pout)*14.69595/...
    1013250;
end
%..........................................................................
% Display selected output
%..........................................................................
fprintf('\n abstol = %8.1e   reltol = %8.1e\n', abstol, reltol);
fprintf('\n ncall = %4d\n',ncall);
%..........................................................................
% Display important parameters
%..........................................................................
fprintf('\n process step = %d       dYbardZbar differential method = %d',
adsorptionORpurge, dYbardZbarDifferentialMethod);
fprintf('\n MassAxial = %d       HeatAxial = %d         HeateqmORnoneqm =
%d\n', MassAxial, HeatAxial, HeateqmORnoneqm);

fprintf('\n P0 = %g atm          T0 = %g K       yN20 = %g       Q0 = %g
mmol/cm2/sec', P0/1013250, T0, (1-y10), Q0);
fprintf('\n tfeed = %g sec      Tfeed = %g K    yN2feed = %g    Qfeed = %g
mmol/cm2/sec    qfeed = %g mmol/sec', tfeed, Tfeed, (1-y1feed), Qfeed,
qfeed);
fprintf('\n dp = %g um         L = %g cm        D = %g cm       Pressure drop
across bed at nout = %g psi', dp*10000, L, D, Pressuredropacrossbed_psi);
fprintf('\n k1 = %g sec-1      k2 = %g sec-1    rhob = %g g/cc\n\n', k1, k2,
rhob);
%..........................................................................
% Plot numerical solutions vs. independent variables (t,z)
%..........................................................................
z0 = 0.0;
zend = L;
z = linspace(z0,zend,n);
```

```
t0 = 0.0;
tend = tfeed;
t = linspace(t0,tend,nout);
%..........................................................................
if choose == 1 % start of choose == 1 : 2D subplots of y1, y2, n1, n2, T, P
%..........................................................................
% Subplots of y1, y2, n1, n2, T, P
%..........................................................................
figure(1);

subplot(2,2,1)
plot(t,y1bar(:,(n+1)/2),'-r',t,(1-y1bar(:,(n+1)/2)),'-b'); %axis tight
h = legend('y_{O2}','y_{N2}',1);
title('y_{O2} and y_{N2} vs. time at L/2'); xlabel('time (sec)'); .........
    ylabel('y_{O2} and y_{N2}')

subplot(2,2,2)
plot(t,(Qfeed/(k1*rhob*L)).*n1bar(:,(n+1)/2),'-r',t,(Qfeed/(k2*rhob*.......
    L)).*n2bar(:,(n+1)/2),'-b'); %axis tight
h = legend('n_{O2}','n_{N2}',1);
title('n_{O2} and n_{N2} vs. time at L/2'); xlabel('time (sec)'); .........
    ylabel('n_{O2} and n_{N2} (mmol/g adsorbent)')

subplot(2,2,3)
plot(t,Tref+(Tmax-Tmin).*thetabar(:,(n+1)/2)); %axis tight
title('T vs. time at L/2'); xlabel('time (sec)'); ylabel('T (K)')

subplot(2,2,4)
plot(z,((Pmax-Pmin)/1013250).*Pbar(nout,:)); %axis tight
title('P vs. bed axial position at tfinal'); ..............................
    xlabel('Bed axial position, z (cm)'); ylabel('P (atm)')
end % end of choose == 1
%..........................................................................
if choose == 2 % start of choose == 2 : Individual 2D plots
%..........................................................................
% Gas phase mole fraction, y1 and y2, vs. time at L/2
%..........................................................................
figure(2)
plot(t,y1bar(:,(n+1)/2),'-r>',t,(1-y1bar(:,(n+1)/2)),'-bs','LineWidth',....
    2,'MarkerSize',3)
h = legend('y_{O2}','y_{N2}',1);
xlabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
ylabel('Gas phase mole fraction, y_{O2} and y_{N2}','color','k',...........
    'fontsize',10,'fontweight','b')
title('Gas phase mole fraction vs. time at L/2','color','k','fontsize',....
    12,'fontweight','b')
axis([0 tfeed 0 1])
grid on
%..........................................................................
% Amount adsorbed on adsorbent, n1 and n2, vs. time at L/2
%..........................................................................
figure(3)
plot(t,(Qfeed/(k1*rhob*L)).*n1bar(:,(n+1)/2),'-r>',t,(Qfeed/(k2*rhob*......
    L)).*n2bar(:,(n+1)/2),'-bs','LineWidth',2,'MarkerSize',3)
h = legend('n_{O2}','n_{N2}',1);
xlabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
ylabel('Amount adsorbed on adsorbent, n_{O2} and n_{N2} (mmol/g
adsorbent)','color','k','fontsize',10,'fontweight','b')
```

```
title('Amount adsorbed on adsorbent vs. time at
L/2','color','k','fontsize',12,'fontweight','b')
grid on
%...........................................................................
% Temperature, T, vs. time at L/2
%...........................................................................
figure(4)
plot(t,(Tref+(Tmax-Tmin).*thetabar(:,(n+1)/2)),'LineWidth',2)
xlabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
ylabel('Temperature, T (K)','color','k','fontsize',10,'fontweight','b')
title('Temperature vs. time at
L/2','color','k','fontsize',12,'fontweight','b')
grid on
%...........................................................................
% Pressure, P, vs. bed axial position at t=tfinal
%...........................................................................
figure(5)
plot(z,((Pmax-Pmin)/1013250).*Pbar(nout,:),'LineWidth',2)
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Pressure, P (atm)','color','k','fontsize',10,'fontweight','b')
title('Pressure vs. bed axial position at
tfinal','color','k','fontsize',12,'fontweight','b')
grid on
%...........................................................................
% Gas phase concentration, rhog, vs. bed axial position at t=tfinal
%...........................................................................
figure(6)
plot(z,(Qfeed*tfeed/L).*rhogbar(nout,:),'LineWidth',2)
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Gas phase concentration, \rho_g
(mmol/cm^3)','color','k','fontsize',10,'fontweight','b')
title('Gas phase concentration vs. bed axial position at
tfinal','color','k','fontsize',12,'fontweight','b')
grid on
%...........................................................................
% Mass flux, Q, vs. bed axial position at t=tfinal
%...........................................................................
figure(7)
plot(z,Qfeed.*Qbar(nout,:),'LineWidth',2)
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Mass flux, Q
(mmol/cm^2/s)','color','k','fontsize',10,'fontweight','b')
title('Mass flux vs. bed axial position at
tfinal','color','k','fontsize',12,'fontweight','b')
grid on
%...........................................................................
% Superficial flow rate, us, vs. bed axial position at t=tfinal
%...........................................................................
figure(8)
plot(z,(L/tfeed).*usbar(nout,:),'LineWidth',2)
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Superficial velocity, u_s
(cm/s)','color','k','fontsize',10,'fontweight','b')
title('Superficial velocity vs. bed axial position at
tfinal','color','k','fontsize',12,'fontweight','b')
grid on
```

```
end % end of choose == 2
%.........................................................................
if choose == 4 % start of choose == 3 : Surface plots
%.........................................................................
% Gas phase mole fraction, y1
%.........................................................................
figure(9)
% mesh(z,t,y1bar,'EdgeColor','black')
surf(z,t,y1bar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Gas phase mole fraction,
y_{O2}','color','k','fontsize',10,'fontweight','b')
axis tight
%.........................................................................
% Gas phase mole fraction, y2
%.........................................................................
figure(10)
% mesh(z,t,(1-y1bar),'EdgeColor','black')
surf(z,t,(1-y1bar))
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Gas phase mole fraction,
y_{N2}','color','k','fontsize',10,'fontweight','b')
axis tight
%.........................................................................
% Amount adsorbed on adsorbent, n1
%.........................................................................
figure(11)
% mesh(z,t,(Qfeed/(k1*rhob*L)).*n1bar,'EdgeColor','black')
surf(z,t,(Qfeed/(k1*rhob*L)).*n1bar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Amount adsorbed on adsorbent, n_{O2} (mmol/g
adsorbent)','color','k','fontsize',10,'fontweight','b')
axis tight
%.........................................................................
% Amount adsorbed on adsorbent, n2
%.........................................................................
figure(12)
% mesh(z,t,(Qfeed/(k2*rhob*L)).*n2bar,'EdgeColor','black')
surf(z,t,(Qfeed/(k2*rhob*L)).*n2bar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Amount adsorbed on adsorbent, n_{N2} (mmol/g
adsorbent)','color','k','fontsize',10,'fontweight','b')
axis tight
%.........................................................................
% Temperature, Tg
%.........................................................................
figure(13)
% mesh(z,t,(Tmin+(Tmax-Tmin).*thetabar),'EdgeColor','black')
```

```
surf(z,t,(Tref+(Tmax-Tmin).*thetabarg))
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Gas Phase Temperature, T
(K)','color','k','fontsize',10,'fontweight','b')
axis tight
%..........................................................................
% Temperature, Ts
%..........................................................................
figure(14)
% mesh(z,t,(Tmin+(Tmax-Tmin).*thetabar),'EdgeColor','black')
surf(z,t,(Tref+(Tmax-Tmin).*thetabars))
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Solid Phase Temperature, T
(K)','color','k','fontsize',10,'fontweight','b')
axis tight
%..........................................................................
% Pressure, P
%..........................................................................
figure(15)
% mesh(z,t,((Pmax-Pmin)/1013250).*Pbar),'EdgeColor','black')
surf(z,t,((Pmax-Pmin)/1013250).*Pbar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Pressure, P (atm)','color','k','fontsize',10,'fontweight','b')
axis tight
%..........................................................................
% Gas phase concentration, rhog
%..........................................................................
figure(16)
% mesh(z,t,(Qfeed*tfeed/L).*rhogbar),'EdgeColor','black')
surf(z,t,(Qfeed*tfeed/L).*rhogbar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Gas phase concentration, \rho_g
(mmol/cm^3)','color','k','fontsize',10,'fontweight','b')
axis tight
%..........................................................................
% Mass flux, Q
%..........................................................................
figure(17)
% mesh(z,t,Qfeed.*Qbar),'EdgeColor','black')
surf(z,t,Qfeed.*Qbar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Mass flux, Q
(mmol/cm^2/s)','color','k','fontsize',10,'fontweight','b')
axis tight
%..........................................................................
```

```
% Superficial flow rate, us
%...........................................................................
figure(18)
% mesh(z,t,(L/tfeed).*usbar),'EdgeColor','black')
surf(z,t,(L/tfeed).*usbar)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Superficial velocity, u_s
(cm/s)','color','k','fontsize',10,'fontweight','b')
axis tight
%...........................................................................
% Volumetric gas-solid heat transfer coefficient, ha, cal/cm3/sec/K
%...........................................................................
figure(19)
surf(z,t,ha)
colormap hsv
xlabel('Bed axial position, z
(cm)','color','k','fontsize',10,'fontweight','b')
ylabel('Time, t (sec)','color','k','fontsize',10,'fontweight','b')
zlabel('Volumetric gas-solid heat transfer coefficient, ha
(cal/cm^3/sec/K)','color','k','fontsize',10,'fontweight','b')
axis tight
%end % end of choose == 3
%...........................................................................
%if choose == 4 % start of choose = 4 : The 2D plots that Dr. Sircar is
interested in
%...........................................................................
% Among nout points, take 5 time points spreading evenly between [0,tfeed]
%...........................................................................
for m=1:5
    it(m) = 1+ (m-1)*(nout-1)/4;
end
% If nout = 101 ==> it = 1    26    51    76   101

for m=1:5
    time((m)) = t(it(m));
end
% Resultstime = time          % Evenly spaced time for displayed results
%...........................................................................
% Now I want to take 5 time points between [0,tfeed] that display the
% results beautifully
%...........................................................................
% Resultstime = [t(1); t(11); t(21); t(31); t(41); t(51); t(101)]
%...........................................................................
% Adsorption / Desorption Profiles
%...........................................................................
% particle diameter in micrometer
dp = dp *10000;

if adsorptionORpurge == 111
% O2 initially present in the bed
O2ingas = epsilon*A*L*(y10)*rhog0;      % mmoles O2
O2insolid = A*L*rhob*n10;               % mmoles O2
TotalO2inbed0_mmolesO2 = O2ingas + O2insolid    % mmoles O2

% O2 remaining at every it, averaged over the whole bed i=1:n
for it = 1:nout
    y1acc(it) = 0;
    rhogbaracc(it) = 0;
    n1baracc(it) = 0;
    for i = 1:n
        y1acc(it) = y1bar(it,i) + y1acc(it);
        rhogbaracc(it) = rhogbar(it,i) + rhogbaracc(it);
        n1baracc(it) = n1bar(it,i) + n1baracc(it);
    end
    y1ave(it) = y1acc(it)/n;
    rhogave(it) = (Qfeed*tfeed/L)*rhogbaracc(it)/n;
    n1ave(it) = (Qfeed/(k1*rhob*L))*n1baracc(it)/n;
end

for it = 1:nout
O2remainingas(it) = epsilon*A*L*y1ave(it)*rhogave(it);    % mmoles O2
O2remaininsolid(it) = A*L*rhob*n1ave(it);                 % mmoles O2
TotalO2remaininbed_mmolesO2(it) = O2remainingas(it) + O2remaininsolid(it);
% mmoles O2
frac_O2_desorbed(it) = (TotalO2inbed0_mmolesO2 - ..........................
    TotalO2remaininbed_mmolesO2(it))/TotalO2inbed0_mmolesO2;
end

frac_O2_desorbedv = frac_O2_desorbed';
end


if adsorptionORpurge == 222
% N2 initially present in the bed
N2ingas = epsilon*A*L*(1-y10)*rhog0;    % mmoles N2
N2insolid = A*L*rhob*n20;               % mmoles N2
TotalN2inbed0_mmolesN2 = N2ingas + N2insolid    % mmoles N2

% N2 remaining at every it, averaged over the whole bed i=1:n
for it = 1:nout
    y2acc(it) = 0;
    rhogbaracc(it) = 0;
    n2baracc(it) = 0;
    for i = 1:n
        y2acc(it) = (1-y1bar(it,i)) + y2acc(it);
        rhogbaracc(it) = rhogbar(it,i) + rhogbaracc(it);
        n2baracc(it) = n2bar(it,i) + n2baracc(it);
    end
    y2ave(it) = y2acc(it)/n;
    rhogave(it) = (Qfeed*tfeed/L)*rhogbaracc(it)/n;
    n2ave(it) = (Qfeed/(k2*rhob*L))*n2baracc(it)/n;
end

for it = 1:nout
N2remainingas(it) = epsilon*A*L*y2ave(it)*rhogave(it);    % mmoles N2
N2remaininsolid(it) = A*L*rhob*n2ave(it);                 % mmoles N2
TotalN2remaininbed_mmolesN2(it) = N2remainingas(it) + N2remaininsolid(it);
% mmoles N2
frac_N2_desorbed(it) = (TotalN2inbed0_mmolesN2 - ..........................
    TotalN2remaininbed_mmolesN2(it))/TotalN2inbed0_mmolesN2;
end

frac_N2_desorbedv = frac_N2_desorbed';
end
%..........................................................................
figure(20)
plot(zbar,(y1bar(1,:)),zbar,(y1bar((nout-1)*0.2+1,:)),zbar,(y1bar.........
    ((nout-1)*0.4+1,:)),zbar,(y1bar((nout-1)*0.6+1,:)),zbar,(y1bar.........
    ((nout-1)*0.8+1,:)),zbar,(y1bar(nout,:)),'LineWidth',2,................
    'MarkerSize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
text(.02,1.05,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('y_{O2}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.1])


figure(21)
plot(zbar,(1-y1bar(1,:)),zbar,(1-y1bar((nout-1)*0.2+1,:)),zbar,(1-.........
    y1bar((nout-1)*0.4+1,:)),zbar,(1-y1bar((nout-1)*0.6+1,:)),zbar,(1-.....
    y1bar((nout-1)*0.8+1,:)),zbar,(1-y1bar(nout,:)),'LineWidth',2,.........
    'MarkerSize',3); %axis tight
h = legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8',.........
    'tbar=1.0',1);
text(.02,1.05,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('y_{N2}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.1])


figure(22)
plot(zbar,((Qfeed/(k1*rhob*L)).*n1bar(1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar(
(nout-1)*0.2+1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar((nout-
1)*0.4+1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar((nout-
1)*0.6+1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar((nout-
1)*0.8+1,:)),zbar,((Qfeed/(k1*rhob*L)).*n1bar(nout,:)),'LineWidth',2,'MarkerS
ize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
```

```
ylabel('n_{O2} (mmoles O_2/g)','color','k','fontsize',12,'fontweight','b')


figure(23)
plot(zbar,((Qfeed/(k2*rhob*L)).*n2bar(1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar(
(nout-1)*0.2+1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar((nout-
1)*0.4+1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar((nout-
1)*0.6+1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar((nout-
1)*0.8+1,:)),zbar,((Qfeed/(k2*rhob*L)).*n2bar(nout,:)),'LineWidth',2,'MarkerS
ize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{N2} (mmoles N_2/g)','color','k','fontsize',12,'fontweight','b')


figure(24)
plot(zbar,((Tmax-Tmin).*thetabarg(1,:)+Tref),'-k',zbar,((Tmax-
Tmin).*thetabars(1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabarg((nout-
1)*0.2+1,:)+Tref),'-r<',zbar,((Tmax-Tmin).*thetabars((nout-
1)*0.2+1,:)+Tref),'-s','LineWidth',1,'MarkerSize',3); %axis tight
h = legend('Tg_0','Ts_0','Tg_{0.2}','Ts_{0.2}',1);
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('T_g and T_s (K) at tbar = 0,
0.2','color','k','fontsize',12,'fontweight','b')
%axis([0 1 282 300])


figure(25)
plot(zbar,((Tmax-Tmin).*thetabarg(1,:)+Tref),'-k',zbar,((Tmax-
Tmin).*thetabars(1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabarg((nout-
1)*0.2+1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabars((nout-
1)*0.2+1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabarg((nout-
1)*0.4+1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabars((nout-
1)*0.4+1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabarg((nout-
1)*0.6+1,:)+Tref),'-r<',zbar,((Tmax-Tmin).*thetabars((nout-
1)*0.6+1,:)+Tref),'-s','LineWidth',1,'MarkerSize',3); %axis tight
h =
legend('Tg_0','Ts_0','Tg_{0.2}','Ts_{0.2}','Tg_{0.4}','Ts_{0.4}','Tg_{0.6}','
Ts_{0.6}',1);
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
```

```
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('T_g and T_s (K) from tbar = 0 to
0.6','color','k','fontsize',12,'fontweight','b')
%axis([0 1 282 300])


figure(26)
plot(zbar,((Tmax-Tmin).*thetabarg(1,:)+Tref),'-k',zbar,((Tmax-
Tmin).*thetabars(1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabarg((nout-
1)*0.2+1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabars((nout-
1)*0.2+1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabarg((nout-
1)*0.4+1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabars((nout-
1)*0.4+1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabarg((nout-
1)*0.6+1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabars((nout-
1)*0.6+1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabarg((nout-
1)*0.8+1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabars((nout-
1)*0.8+1,:)+Tref),'-k',zbar,((Tmax-Tmin).*thetabarg((nout-
1)*1.0+1,:)+Tref),'-r<',zbar,((Tmax-Tmin).*thetabars((nout-
1)*1.0+1,:)+Tref),'-s','LineWidth',1,'MarkerSize',3); %axis tight
h =
legend('Tg_0','Ts_0','Tg_{0.2}','Ts_{0.2}','Tg_{0.4}','Ts_{0.4}','Tg_{0.6}','
Ts_{0.6}','Tg_{0.8}','Ts_{0.8}','Tg_{1.0}','Ts_{1.0}',1);
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('T_g and T_s (K) from tbar = 0 to
1','color','k','fontsize',12,'fontweight','b')
%axis([0 1 282 300])


figure(27)
plot(zbar,(Pmax-Pmin).*Pbar(1,:)*14.69595/1013250,zbar,(Pmax-
Pmin).*Pbar((nout-1)*0.2+1,:)*14.69595/1013250,zbar,(Pmax-Pmin).*Pbar((nout-
1)*0.4+1,:)*14.69595/1013250,zbar,(Pmax-Pmin).*Pbar((nout-
1)*0.6+1,:)*14.69595/1013250,zbar,(Pmax-Pmin).*Pbar((nout-
1)*0.8+1,:)*14.69595/1013250,zbar,(Pmax-Pmin).*Pbar((nout-
1)*1.0+1,:)*14.69595/1013250,'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
%if adsorptionORpurge == 111
%    title('Bed initially saturated with 100% O_2 adsorbed by 79%
N_2','color','k','fontsize',12,'fontweight','b');
%end
%if adsorptionORpurge == 222
%    title('Bed initially saturated with 79% N_2 purged by 100%
O_2','color','k','fontsize',12,'fontweight','b');
%end
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('Pressure (psi)','color','k','fontsize',12,'fontweight','b')
```

```
if adsorptionORpurge == 222
%..........................................................................
% Search the tbar that produces fracdes = 0.3, 0.5, 0.7
% Then plot the profiles for yN2, nN2, Tg, Ts at these tbar
%..........................................................................
minDF_3 = 1; % minimum difference between fraction desorbed and 0.3
for it=1:nout
    F_3 = abs(frac_N2_desorbed(1,it) - 0.3);
    if minDF_3 > F_3
        minDF_3 = F_3;
        it_3 = it;
    end
end
tbar_3 = tbarv(it_3,1)
minDF_3

minDF_5 = 1; % minimum difference between fraction desorbed and 0.5
for it=1:nout
    F_5 = abs(frac_N2_desorbed(1,it) - 0.5);
    if minDF_5 > F_5
        minDF_5 = F_5;
        it_5 = it;
    end
end
tbar_5 = tbarv(it_5,1)
minDF_5

minDF_7 = 1; % minimum difference between fraction desorbed and 0.7
for it=1:nout
    F_7 = abs(frac_N2_desorbed(1,it) - 0.7);
    if minDF_7 > F_7
        minDF_7 = F_7;
        it_7 = it;
    end
end
tbar_7 = tbarv(it_7,1)
minDF_7

minDF_8 = 1; % minimum difference between fraction desorbed and 0.8
for it=1:nout
    F_8 = abs(frac_N2_desorbed(1,it) - 0.8);
    if minDF_8 > F_8
        minDF_8 = F_8;
        it_8 = it;
    end
end
tbar_8 = tbarv(it_8,1)
minDF_8

minDF_85 = 1; % minimum difference between fraction desorbed and 0.85
for it=1:nout
    F_85 = abs(frac_N2_desorbed(1,it) - 0.85);
    if minDF_85 > F_85
        minDF_85 = F_85;
        it_85 = it;
    end
end
tbar_85 = tbarv(it_85,1)
minDF_85
```

```
minDF_9 = 1; % minimum difference between fraction desorbed and 0.9
for it=1:nout
    F_9 = abs(frac_N2_desorbed(1,it) - 0.9);
    if minDF_9 > F_9
        minDF_9 = F_9;
        it_9 = it;
    end
end
tbar_9 = tbarv(it_9,1)
minDF_9

minDF_95 = 1; % minimum difference between fraction desorbed and 0.95
for it=1:nout
    F_95 = abs(frac_N2_desorbed(1,it) - 0.95);
    if minDF_95 > F_95
        minDF_95 = F_95;
        it_95 = it;
    end
end
tbar_95 = tbarv(it_95,1)
minDF_95
%...........................................................................
for i = 1:n
    Nn1f357(1,i)=(Qfeed/(k1*rhob*L))*n1bar(1    ,i)/n10; % normalized with n10
    Nn1f357(2,i)=(Qfeed/(k1*rhob*L))*n1bar(it_3,i)/n10;
    Nn1f357(3,i)=(Qfeed/(k1*rhob*L))*n1bar(it_5,i)/n10;
    Nn1f357(4,i)=(Qfeed/(k1*rhob*L))*n1bar(it_7,i)/n10;
    Nn1f357(5,i)=(Qfeed/(k1*rhob*L))*n1bar(it_8,i)/n10;
    Nn1f357(6,i)=(Qfeed/(k1*rhob*L))*n1bar(it_85,i)/n10;
    Nn1f357(7,i)=(Qfeed/(k1*rhob*L))*n1bar(it_9,i)/n10;
    Nn1f357(8,i)=(Qfeed/(k1*rhob*L))*n1bar(it_95,i)/n10;

    n1f357(1,i)=(Qfeed/(k1*rhob*L))*n1bar(1    ,i); % mmol O2/g
    n1f357(2,i)=(Qfeed/(k1*rhob*L))*n1bar(it_3,i);
    n1f357(3,i)=(Qfeed/(k1*rhob*L))*n1bar(it_5,i);
    n1f357(4,i)=(Qfeed/(k1*rhob*L))*n1bar(it_7,i);
    n1f357(5,i)=(Qfeed/(k1*rhob*L))*n1bar(it_8,i);
    n1f357(6,i)=(Qfeed/(k1*rhob*L))*n1bar(it_85,i);
    n1f357(7,i)=(Qfeed/(k1*rhob*L))*n1bar(it_9,i);
    n1f357(8,i)=(Qfeed/(k1*rhob*L))*n1bar(it_95,i);

    Nn2f357(1,i)=(Qfeed/(k2*rhob*L))*n2bar(1    ,i)/n20; % normalized with n20
    Nn2f357(2,i)=(Qfeed/(k2*rhob*L))*n2bar(it_3,i)/n20;
    Nn2f357(3,i)=(Qfeed/(k2*rhob*L))*n2bar(it_5,i)/n20;
    Nn2f357(4,i)=(Qfeed/(k2*rhob*L))*n2bar(it_7,i)/n20;
    Nn2f357(5,i)=(Qfeed/(k2*rhob*L))*n2bar(it_8,i)/n20;
    Nn2f357(6,i)=(Qfeed/(k2*rhob*L))*n2bar(it_85,i)/n20;
    Nn2f357(7,i)=(Qfeed/(k2*rhob*L))*n2bar(it_9,i)/n20;
    Nn2f357(8,i)=(Qfeed/(k2*rhob*L))*n2bar(it_95,i)/n20;

    n2f357(1,i)=(Qfeed/(k2*rhob*L))*n2bar(1    ,i); % mmol N2/g
    n2f357(2,i)=(Qfeed/(k2*rhob*L))*n2bar(it_3,i);
    n2f357(3,i)=(Qfeed/(k2*rhob*L))*n2bar(it_5,i);
    n2f357(4,i)=(Qfeed/(k2*rhob*L))*n2bar(it_7,i);
    n2f357(5,i)=(Qfeed/(k2*rhob*L))*n2bar(it_8,i);
    n2f357(6,i)=(Qfeed/(k2*rhob*L))*n2bar(it_85,i);
    n2f357(7,i)=(Qfeed/(k2*rhob*L))*n2bar(it_9,i);
    n2f357(8,i)=(Qfeed/(k2*rhob*L))*n2bar(it_95,i);
    totn10=(A*L*rhob)*n10 + epsilon*A*L*(y10)*rhog0;   % total initial mmol
O2 in solid and gas phases
    totn20=(A*L*rhob)*n20 + epsilon*A*L*(1-y10)*rhog0; % total initial mmol
N2 in solid and gas phases

    totn1f357(1,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(1,i) +
epsilon*A*L*(y1bar(1,i))*rhogbar(1,i)*(Qfeed*tfeed/L); % total mmol O2 in gas
and solid phases
    totn1f357(2,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_3,i) +
epsilon*A*L*(y1bar(it_3,i))*rhogbar(it_3,i)*(Qfeed*tfeed/L);
    totn1f357(3,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_5,i) +
epsilon*A*L*(y1bar(it_5,i))*rhogbar(it_5,i)*(Qfeed*tfeed/L);
    totn1f357(4,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_7,i) +
epsilon*A*L*(y1bar(it_7,i))*rhogbar(it_7,i)*(Qfeed*tfeed/L);
    totn1f357(5,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_8,i) +
epsilon*A*L*(y1bar(it_8,i))*rhogbar(it_8,i)*(Qfeed*tfeed/L);
    totn1f357(6,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_85,i) +
epsilon*A*L*(y1bar(it_85,i))*rhogbar(it_85,i)*(Qfeed*tfeed/L);
    totn1f357(7,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_9,i) +
epsilon*A*L*(y1bar(it_9,i))*rhogbar(it_9,i)*(Qfeed*tfeed/L);
    totn1f357(8,i)=(A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_95,i) +
epsilon*A*L*(y1bar(it_95,i))*rhogbar(it_95,i)*(Qfeed*tfeed/L);

    Ntotn1f357(1,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(1,i) +
epsilon*A*L*(y1bar(1,i))*rhogbar(1,i)*(Qfeed*tfeed/L))/totn10; % normalized
with totn10
    Ntotn1f357(2,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_3,i) +
epsilon*A*L*(y1bar(it_3,i))*rhogbar(it_3,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(3,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_5,i) +
epsilon*A*L*(y1bar(it_5,i))*rhogbar(it_5,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(4,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_7,i) +
epsilon*A*L*(y1bar(it_7,i))*rhogbar(it_7,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(5,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_8,i) +
epsilon*A*L*(y1bar(it_8,i))*rhogbar(it_8,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(6,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_85,i) +
epsilon*A*L*(y1bar(it_85,i))*rhogbar(it_85,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(7,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_9,i) +
epsilon*A*L*(y1bar(it_9,i))*rhogbar(it_9,i)*(Qfeed*tfeed/L))/totn10;
    Ntotn1f357(8,i)=((A*L*rhob)*(Qfeed/(k1*rhob*L))*n1bar(it_95,i) +
epsilon*A*L*(y1bar(it_95,i))*rhogbar(it_95,i)*(Qfeed*tfeed/L))/totn10;

    totn2f357(1,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(1,i) +
epsilon*A*L*(1-y1bar(1,i))*rhogbar(1,i)*(Qfeed*tfeed/L); % total mmol N2 in
gas and solid phases
    totn2f357(2,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_3,i) +
epsilon*A*L*(1-y1bar(it_3,i))*rhogbar(it_3,i)*(Qfeed*tfeed/L);
    totn2f357(3,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_5,i) +
epsilon*A*L*(1-y1bar(it_5,i))*rhogbar(it_5,i)*(Qfeed*tfeed/L);
    totn2f357(4,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_7,i) +
epsilon*A*L*(1-y1bar(it_7,i))*rhogbar(it_7,i)*(Qfeed*tfeed/L);
    totn2f357(5,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_8,i) +
epsilon*A*L*(1-y1bar(it_8,i))*rhogbar(it_8,i)*(Qfeed*tfeed/L);
    totn2f357(6,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_85,i) +
epsilon*A*L*(1-y1bar(it_85,i))*rhogbar(it_85,i)*(Qfeed*tfeed/L);
    totn2f357(7,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_9,i) +
epsilon*A*L*(1-y1bar(it_9,i))*rhogbar(it_9,i)*(Qfeed*tfeed/L);
    totn2f357(8,i)=(A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_95,i) +
epsilon*A*L*(1-y1bar(it_95,i))*rhogbar(it_95,i)*(Qfeed*tfeed/L);
```

```
    Ntotn2f357(1,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(1,i) +
epsilon*A*L*(1-y1bar(1,i))*rhogbar(1,i)*(Qfeed*tfeed/L))/totn20; % normalized
with totn20
    Ntotn2f357(2,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_3,i) +
epsilon*A*L*(1-y1bar(it_3,i))*rhogbar(it_3,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(3,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_5,i) +
epsilon*A*L*(1-y1bar(it_5,i))*rhogbar(it_5,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(4,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_7,i) +
epsilon*A*L*(1-y1bar(it_7,i))*rhogbar(it_7,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(5,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_8,i) +
epsilon*A*L*(1-y1bar(it_8,i))*rhogbar(it_8,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(6,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_85,i) +
epsilon*A*L*(1-y1bar(it_85,i))*rhogbar(it_85,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(7,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_9,i) +
epsilon*A*L*(1-y1bar(it_9,i))*rhogbar(it_9,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2f357(8,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar(it_95,i) +
epsilon*A*L*(1-y1bar(it_95,i))*rhogbar(it_95,i)*(Qfeed*tfeed/L))/totn20;

    Remain_N2_inbed_percent(1,i) = 100*totn2f357(1,i)/(totn1f357(1,i) +
totn2f357(1,i));
    Remain_N2_inbed_percent(2,i) = 100*totn2f357(2,i)/(totn1f357(2,i) +
totn2f357(2,i));
    Remain_N2_inbed_percent(3,i) = 100*totn2f357(3,i)/(totn1f357(3,i) +
totn2f357(3,i));
    Remain_N2_inbed_percent(4,i) = 100*totn2f357(4,i)/(totn1f357(4,i) +
totn2f357(4,i));
    Remain_N2_inbed_percent(5,i) = 100*totn2f357(5,i)/(totn1f357(5,i) +
totn2f357(5,i));
    Remain_N2_inbed_percent(6,i) = 100*totn2f357(6,i)/(totn1f357(6,i) +
totn2f357(6,i));
    Remain_N2_inbed_percent(7,i) = 100*totn2f357(7,i)/(totn1f357(7,i) +
totn2f357(7,i));
    Remain_N2_inbed_percent(8,i) = 100*totn2f357(8,i)/(totn1f357(8,i) +
totn2f357(8,i));

    y1barf357(1,i)=y1bar(1   ,i);
    y1barf357(2,i)=y1bar(it_3,i);
    y1barf357(3,i)=y1bar(it_5,i);
    y1barf357(4,i)=y1bar(it_7,i);
    y1barf357(5,i)=y1bar(it_8,i);
    y1barf357(6,i)=y1bar(it_85,i);
    y1barf357(7,i)=y1bar(it_9,i);
    y1barf357(8,i)=y1bar(it_95,i);

    y2barf357(1,i)=1-y1bar(1   ,i);
    y2barf357(2,i)=1-y1bar(it_3,i);
    y2barf357(3,i)=1-y1bar(it_5,i);
    y2barf357(4,i)=1-y1bar(it_7,i);
    y2barf357(5,i)=1-y1bar(it_8,i);
    y2barf357(6,i)=1-y1bar(it_85,i);
    y2barf357(7,i)=1-y1bar(it_9,i);
    y2barf357(8,i)=1-y1bar(it_95,i);

    Tgf357(1,i)=(Tmax-Tmin)*thetabarg(1   ,i)+Tref; % gas phase temperature,
K
    Tgf357(2,i)=(Tmax-Tmin)*thetabarg(it_3,i)+Tref;
    Tgf357(3,i)=(Tmax-Tmin)*thetabarg(it_5,i)+Tref;
    Tgf357(4,i)=(Tmax-Tmin)*thetabarg(it_7,i)+Tref;
    Tgf357(5,i)=(Tmax-Tmin)*thetabarg(it_8,i)+Tref;
    Tgf357(6,i)=(Tmax-Tmin)*thetabarg(it_85,i)+Tref;
    Tgf357(7,i)=(Tmax-Tmin)*thetabarg(it_9,i)+Tref;
    Tgf357(8,i)=(Tmax-Tmin)*thetabarg(it_95,i)+Tref;

    TgTof357(1,i)=(Tmax-Tmin)*thetabarg(1    ,i)+Tref  - T0; % Tg - To, K
    TgTof357(2,i)=(Tmax-Tmin)*thetabarg(it_3,i)+Tref - T0;
    TgTof357(3,i)=(Tmax-Tmin)*thetabarg(it_5,i)+Tref - T0;
    TgTof357(4,i)=(Tmax-Tmin)*thetabarg(it_7,i)+Tref - T0;
    TgTof357(5,i)=(Tmax-Tmin)*thetabarg(it_8,i)+Tref - T0;
    TgTof357(6,i)=(Tmax-Tmin)*thetabarg(it_85,i)+Tref - T0;
    TgTof357(7,i)=(Tmax-Tmin)*thetabarg(it_9,i)+Tref - T0;
    TgTof357(8,i)=(Tmax-Tmin)*thetabarg(it_95,i)+Tref - T0;

    thetabargf357(1,i)=thetabarg(1    ,i);
    thetabargf357(2,i)=thetabarg(it_3,i);
    thetabargf357(3,i)=thetabarg(it_5,i);
    thetabargf357(4,i)=thetabarg(it_7,i);
    thetabargf357(5,i)=thetabarg(it_8,i);
    thetabargf357(6,i)=thetabarg(it_85,i);
    thetabargf357(7,i)=thetabarg(it_9,i);
    thetabargf357(8,i)=thetabarg(it_95,i);

    Tsf357(1,i)=(Tmax-Tmin)*thetabars(1    ,i)+Tref; % solid phase
temperature, K
    Tsf357(2,i)=(Tmax-Tmin)*thetabars(it_3,i)+Tref;
    Tsf357(3,i)=(Tmax-Tmin)*thetabars(it_5,i)+Tref;
    Tsf357(4,i)=(Tmax-Tmin)*thetabars(it_7,i)+Tref;
    Tsf357(5,i)=(Tmax-Tmin)*thetabars(it_8,i)+Tref;
    Tsf357(6,i)=(Tmax-Tmin)*thetabars(it_85,i)+Tref;
    Tsf357(7,i)=(Tmax-Tmin)*thetabars(it_9,i)+Tref;
    Tsf357(8,i)=(Tmax-Tmin)*thetabars(it_95,i)+Tref;

    TsTof357(1,i)=(Tmax-Tmin)*thetabars(1    ,i)+Tref  - T0; % Ts - To, K
    TsTof357(2,i)=(Tmax-Tmin)*thetabars(it_3,i)+Tref - T0;
    TsTof357(3,i)=(Tmax-Tmin)*thetabars(it_5,i)+Tref - T0;
    TsTof357(4,i)=(Tmax-Tmin)*thetabars(it_7,i)+Tref - T0;
    TsTof357(5,i)=(Tmax-Tmin)*thetabars(it_8,i)+Tref - T0;
    TsTof357(6,i)=(Tmax-Tmin)*thetabars(it_85,i)+Tref - T0;
    TsTof357(7,i)=(Tmax-Tmin)*thetabars(it_9,i)+Tref - T0;
    TsTof357(8,i)=(Tmax-Tmin)*thetabars(it_95,i)+Tref - T0;

    thetabarsf357(1,i)=thetabars(1    ,i);
    thetabarsf357(2,i)=thetabars(it_3,i);
    thetabarsf357(3,i)=thetabars(it_5,i);
    thetabarsf357(4,i)=thetabars(it_7,i);
    thetabarsf357(5,i)=thetabars(it_8,i);
    thetabarsf357(6,i)=thetabars(it_85,i);
    thetabarsf357(7,i)=thetabars(it_9,i);
    thetabarsf357(8,i)=thetabars(it_95,i);

    Pbarf357(1,i)=Pbar(1    ,i); % pressure in dimensionless
    Pbarf357(2,i)=Pbar(it_3,i);
    Pbarf357(3,i)=Pbar(it_5,i);
    Pbarf357(4,i)=Pbar(it_7,i);
    Pbarf357(5,i)=Pbar(it_8,i);
    Pbarf357(6,i)=Pbar(it_85,i);
    Pbarf357(7,i)=Pbar(it_9,i);
    Pbarf357(8,i)=Pbar(it_95,i);

    %h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
```

```
    Ntotn2tbar(1,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar((nout-1)*0.0+1,i) +
epsilon*A*L*(1-y1bar((nout-1)*0.0+1,i))*rhogbar((nout-
1)*0.0+1,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2tbar(2,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar((nout-1)*0.2+1,i) +
epsilon*A*L*(1-y1bar((nout-1)*0.2+1,i))*rhogbar((nout-
1)*0.2+1,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2tbar(3,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar((nout-1)*0.4+1,i) +
epsilon*A*L*(1-y1bar((nout-1)*0.4+1,i))*rhogbar((nout-
1)*0.4+1,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2tbar(4,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar((nout-1)*0.6+1,i) +
epsilon*A*L*(1-y1bar((nout-1)*0.6+1,i))*rhogbar((nout-
1)*0.6+1,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2tbar(5,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar((nout-1)*0.8+1,i) +
epsilon*A*L*(1-y1bar((nout-1)*0.8+1,i))*rhogbar((nout-
1)*0.8+1,i)*(Qfeed*tfeed/L))/totn20;
    Ntotn2tbar(6,i)=((A*L*rhob)*(Qfeed/(k2*rhob*L))*n2bar((nout-1)*1.0+1,i) +
epsilon*A*L*(1-y1bar((nout-1)*1.0+1,i))*rhogbar((nout-
1)*1.0+1,i)*(Qfeed*tfeed/L))/totn20;
    %h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
    TgTotbar(1,i)=(Tmax-Tmin)*thetabarg((nout-1)*0.0+1,i)+Tref - T0; % Tg-T0,
K
    TgTotbar(2,i)=(Tmax-Tmin)*thetabarg((nout-1)*0.2+1,i)+Tref - T0; % Tg-T0,
K
    TgTotbar(3,i)=(Tmax-Tmin)*thetabarg((nout-1)*0.4+1,i)+Tref - T0; % Tg-T0,
K
    TgTotbar(4,i)=(Tmax-Tmin)*thetabarg((nout-1)*0.6+1,i)+Tref - T0; % Tg-T0,
K
    TgTotbar(5,i)=(Tmax-Tmin)*thetabarg((nout-1)*0.8+1,i)+Tref - T0; % Tg-T0,
K
    TgTotbar(6,i)=(Tmax-Tmin)*thetabarg((nout-1)*1.0+1,i)+Tref - T0; % Tg-T0,
K
    %h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
    TsTotbar(1,i)=(Tmax-Tmin)*thetabars((nout-1)*0.0+1,i)+Tref - T0; % Ts-T0,
K
    TsTotbar(2,i)=(Tmax-Tmin)*thetabars((nout-1)*0.2+1,i)+Tref - T0; % Ts-T0,
K
    TsTotbar(3,i)=(Tmax-Tmin)*thetabars((nout-1)*0.4+1,i)+Tref - T0; % Ts-T0,
K
    TsTotbar(4,i)=(Tmax-Tmin)*thetabars((nout-1)*0.6+1,i)+Tref - T0; % Ts-T0,
K
    TsTotbar(5,i)=(Tmax-Tmin)*thetabars((nout-1)*0.8+1,i)+Tref - T0; % Ts-T0,
K
    TsTotbar(6,i)=(Tmax-Tmin)*thetabars((nout-1)*1.0+1,i)+Tref - T0; % Ts-T0,
K
    %h =
legend('tbar=0.0','tbar=0.2','tbar=0.4','tbar=0.6','tbar=0.8','tbar=1.0',1);
    y1bartbar(1,i)=y1bar((nout-1)*0.0+1,i);
    y1bartbar(2,i)=y1bar((nout-1)*0.2+1,i);
    y1bartbar(3,i)=y1bar((nout-1)*0.4+1,i);
    y1bartbar(4,i)=y1bar((nout-1)*0.6+1,i);
    y1bartbar(5,i)=y1bar((nout-1)*0.8+1,i);
    y1bartbar(6,i)=y1bar((nout-1)*1.0+1,i);
end
%.......................................................................
figure(28)
plot(zbar,y1barf357(1,:), zbar,y1barf357(2,:), zbar,y1barf357(3,:),
zbar,y1barf357(4,:), zbar,y1barf357(5,:), zbar,y1barf357(6,:) ,
zbar,y1barf357(7,:), zbar,y1barf357(8,:),'LineWidth',2,'MarkerSize',3); %axis
tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
text(.02,1.25,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
text(.02,1.15,['\fontsize{9}(f=0.30 at tbar=',num2str(tbar_3),');
\fontsize{9}(f=0.50 at tbar=',num2str(tbar_5) ,');  \fontsize{9}(f=0.70 at
tbar=',num2str(tbar_7),');'])
text(.02,1.07,['\fontsize{9}(f=0.80 at tbar=',num2str(tbar_8),');
\fontsize{9}(f=0.85 at tbar=',num2str(tbar_85),');  \fontsize{9}(f=0.90 at
tbar=',num2str(tbar_9),');  \fontsize{9}(f=0.95 at
tbar=',num2str(tbar_95),');'])
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('y_{O2}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.3])

figure(29)
plot(zbar,y2barf357(1,:), zbar,y2barf357(2,:), zbar,y2barf357(3,:),
zbar,y2barf357(4,:), zbar,y2barf357(5,:), zbar,y2barf357(6,:) ,
zbar,y2barf357(7,:), zbar,y2barf357(8,:),'LineWidth',2,'MarkerSize',3); %axis
tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
text(.02,1.25,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
text(.02,1.15,['\fontsize{9}(f=0.30 at tbar=',num2str(tbar_3),');
\fontsize{9}(f=0.50 at tbar=',num2str(tbar_5) ,');  \fontsize{9}(f=0.70 at
tbar=',num2str(tbar_7),');'])
text(.02,1.07,['\fontsize{9}(f=0.80 at tbar=',num2str(tbar_8),');
\fontsize{9}(f=0.85 at tbar=',num2str(tbar_85),');  \fontsize{9}(f=0.90 at
tbar=',num2str(tbar_9),');  \fontsize{9}(f=0.95 at
tbar=',num2str(tbar_95),');'])
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('y_{N2}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.3])

figure(30)
plot(zbar,n1f357(1,:), zbar,n1f357(2,:), zbar,n1f357(3,:), zbar,n1f357(4,:),
zbar,n1f357(5,:), zbar,n1f357(6,:), zbar,n1f357(7,:), zbar,n1f357(8,:)
,'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{O2} (mmoles O_2/g)','color','k','fontsize',12,'fontweight','b')

figure(31)
plot(zbar,n2f357(1,:), zbar,n2f357(2,:), zbar,n2f357(3,:), zbar,n2f357(4,:),
zbar,n2f357(5,:), zbar,n2f357(6,:), zbar,n2f357(7,:), zbar,n2f357(8,:)
,'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{N2} (mmoles N_2/g)','color','k','fontsize',12,'fontweight','b')

figure(32)
plot(zbar,Nn1f357(1,:), zbar,Nn1f357(2,:), zbar,Nn1f357(3,:),
zbar,Nn1f357(4,:), zbar,Nn1f357(5,:), zbar,Nn1f357(6,:), zbar,Nn1f357(7,:),
zbar,Nn1f357(8,:) ,'LineWidth',2,'MarkerSize',3); %axis tight
```

```
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{O2}/n_{O2_0}','color','k','fontsize',12,'fontweight','b')

figure(33)
plot(zbar,Nn2f357(1,:), zbar,Nn2f357(2,:), zbar,Nn2f357(3,:),
zbar,Nn2f357(4,:), zbar,Nn2f357(5,:), zbar,Nn2f357(6,:), zbar,Nn2f357(7,:),
zbar,Nn2f357(8,:) ,'LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
text(.02,1.25,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
text(.02,1.15,['\fontsize{9}(f=0.30 at tbar=',num2str(tbar_3),');
\fontsize{9}(f=0.50 at tbar=',num2str(tbar_5) ,');  \fontsize{9}(f=0.70 at
tbar=',num2str(tbar_7),');'])
text(.02,1.07,['\fontsize{9}(f=0.80 at tbar=',num2str(tbar_8),');
\fontsize{9}(f=0.85 at tbar=',num2str(tbar_85),');  \fontsize{9}(f=0.90 at
tbar=',num2str(tbar_9),');  \fontsize{9}(f=0.95 at
tbar=',num2str(tbar_95),');'])
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('n_{N2}/n_{N2_0}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.3])

figure(34)
plot(zbar,totn2f357(1,:), zbar,totn2f357(2,:), zbar,totn2f357(3,:),
zbar,totn2f357(4,:), zbar,totn2f357(5,:), zbar,totn2f357(6,:),
zbar,totn2f357(7,:), zbar,totn2f357(8,:) ,'LineWidth',2,'MarkerSize',3);
%axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('total N_2 in gas and solid phases, totn_{N2}, (mmoles
N_2)','color','k','fontsize',12,'fontweight','b')

figure(35)
plot(zbar,Ntotn2f357(1,:), zbar,Ntotn2f357(2,:), zbar,Ntotn2f357(3,:),
zbar,Ntotn2f357(4,:), zbar,Ntotn2f357(5,:), zbar,Ntotn2f357(6,:),
zbar,Ntotn2f357(7,:), zbar,Ntotn2f357(8,:) ,'LineWidth',2,'MarkerSize',3);
%axis tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
text(.02,1.25,['\fontsize{11}d_p = ',num2str(dp),' \mum ; \fontsize{11}L =
',num2str(L),' cm ; \fontsize{11}D = ',num2str(D),' cm'])
text(.02,1.15,['\fontsize{9}(f=0.30 at tbar=',num2str(tbar_3),');
\fontsize{9}(f=0.50 at tbar=',num2str(tbar_5) ,');  \fontsize{9}(f=0.70 at
tbar=',num2str(tbar_7),');'])
text(.02,1.07,['\fontsize{9}(f=0.80 at tbar=',num2str(tbar_8),');
\fontsize{9}(f=0.85 at tbar=',num2str(tbar_85),');  \fontsize{9}(f=0.90 at
tbar=',num2str(tbar_9),');  \fontsize{9}(f=0.95 at
tbar=',num2str(tbar_95),');'])
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('totn_{N2}/totn_{N2_0}','color','k','fontsize',12,'fontweight','b')
axis([0 1 0 1.3])

figure(36)
plot(zbar,(Pmax-Pmin).*Pbarf357(1,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(2,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(3,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(4,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(5,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(6,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(7,:)*14.69595/1013250, zbar,(Pmax-
Pmin).*Pbarf357(8,:)*14.69595/1013250,'LineWidth',2,'MarkerSize',3); %axis
tight
h =
legend('f=0.0','f=0.3','f=0.5','f=0.7','f=0.8','f=0.85','f=0.9','f=0.95',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('Pressure (psi)','color','k','fontsize',12,'fontweight','b')

figure(37)
plot(zbar,Tgf357(1,:),'--k',zbar,Tsf357(1,:),'-k',zbar,Tgf357(2,:),'--
m',zbar,Tsf357(2,:),'-m', zbar,Tgf357(3,:),'--b',zbar,Tsf357(3,:),'-b',
zbar,Tgf357(4,:),'--r',zbar,Tsf357(4,:),'-r', zbar,Tgf357(5,:),'--
g',zbar,Tsf357(5,:),'-g', zbar,Tgf357(6,:),'--y', zbar,Tsf357(6,:),'-y',
zbar,Tgf357(7,:),'--c',zbar,Tsf357(7,:),'-c', zbar,Tgf357(8,:),'--p',
zbar,Tsf357(8,:),'-p','LineWidth',2,'MarkerSize',3); %axis tight
h =
legend('Tg_{f=0.0}','Ts_{f=0.0}','Tg_{f=0.3}','Ts_{f=0.3}','Tg_{f=0.5}','Ts_{
f=0.5}','Tg_{f=0.7}','Ts_{f=0.7}','Tg_{f=0.8}','Ts_{f=0.8}','Tg_{f=0.85}','Ts
_{f=0.85}','Tg_{f=0.9}','Ts_{f=0.9}','Tg_{f=0.95}','Ts_{f=0.95}',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('T_g and T_s (K)','color','k','fontsize',12,'fontweight','b')

figure(38)
plot(zbar,Tgf357(1,:)-Tref,'--k',zbar,Tsf357(1,:)-Tref,'-k',zbar,Tgf357(2,:)-
Tref,'--m',zbar,Tsf357(2,:)-Tref,'-m', zbar,Tgf357(3,:)-Tref,'--
b',zbar,Tsf357(3,:)-Tref,'-b', zbar,Tgf357(4,:)-Tref,'--r',zbar,Tsf357(4,:)-
Tref,'-r', zbar,Tgf357(5,:)-Tref,'--g',zbar,Tsf357(5,:)-Tref,'-g',
zbar,Tgf357(6,:)-Tref,'--y', zbar,Tsf357(6,:)-Tref,'-y', zbar,Tgf357(7,:)-
Tref,'--c',zbar,Tsf357(7,:)-Tref,'-c', zbar,Tgf357(8,:)-Tref,'--p',
zbar,Tsf357(8,:)-Tref,'-p', 'LineWidth',2,'MarkerSize',3); %axis tight
h = legend('Tg-T0_{f=0.0}','Ts-T0_{f=0.0}','Tg-T0_{f=0.3}','Ts-
T0_{f=0.3}','Tg-T0_{f=0.5}','Ts-T0_{f=0.5}','Tg-T0_{f=0.7}','Ts-
T0_{f=0.7}','Tg-T0_{f=0.8}','Ts-T0_{f=0.8}','Tg-T0_{f=0.85}','Ts-
T0_{f=0.85}','Tg-T0_{f=0.9}','Ts-T0_{f=0.9}','Tg-T0_{f=0.95}','Ts-
T0_{f=0.95}',1);
xlabel('Bed axial position, zbar','color','k','fontsize',12,'fontweight','b')
ylabel('(T_g - T_0) and (T_s - T_0)
(K)','color','k','fontsize',12,'fontweight','b')
end
end % end of choose == 4
%.........................................................................
```

Appendix 5

Models E, F (with or without gas thermal axial dispersion depending on the Nusselt correlation used)

MOL Routine

```
%..........................................................................
% File: psa_1_sequence_model_EF
%..........................................................................
% Chai Siew Wah, Chemical Engineering, Lehigh University, May 2011
%..........................................................................
% MATHEMATICAL MODEL OF PRESSURE SWING ADSORPTION SYSTEM FOR ONE STEP :
% ADSORPTION OR PURGE
%..........................................................................
%
%                  :......................................:
%   ADSORPTION --> :                                      : --> PRODUCT
%   ADSORPTION --> :                                      : --> PRODUCT
%   ADSORPTION --> :                                      : --> PRODUCT
%                  :......................................:
%                 i=1                                    i=n
%
%                  :......................................:
%     EXHAUST <--  :                                      : <-- PURGE
%     EXHAUST <--  :                                      : <-- PURGE
%     EXHAUST <--  :                                      : <-- PURGE
%                  :......................................:
%                 i=1                                    i=n
%..........................................................................
% Assumptions
%..........................................................................
% (1) Ideal gas law
% (2) Empirical Langmuirian type binary isotherms - from Rege and Yang 1997
% (3) Nonisobaric - Ergun equation solved analytically
% (4) Nonisothermal - adiabatic
% (5) Linear driving force model for mass transfer kinetic
% (6) Mass axial dispersion in gas phase
% (7) Wakao correlation for gas-solid heat transfer kinetic +
%     gas phase thermal axial dispersion
% (8) KuniiSuzuki correlation for gas-solid heat transfer kinetic without
%     gas phase thermal axial dispersion
% (9)  Absence of solid phase thermal axial conduction
% (10) Absence of radial distribution of mass and heat
% (11) Absence of gas mal-distribution or particle agglomeration
%..........................................................................
% 6 ordinary differential equations : y1, y2, n1, n2, Tg, Ts
% 1 direct calculation for          : rhogbar (= Y1bar + Y2bar)
% 1 algebral equation for           : Q (= Ergun equation)
% 1 algebral equation for           : P (= Ideal gas law)
% 1 calculated for                  : us (= Q/rhog) - but reduce its use
% Solve for y1, y2, n1, n2, Tg, Ts, P, rhog, Q, us
%..........................................................................
% (1) Solve Ergun equation for Qbar and properly dealing its positive or
% negative flow direction

% (2) Use Schiesser's boundary condition for mass and heat balances that
% have axial dispersion terms (retain the whole equation except the 2nd
% derivative of space) instead of Danwerts' boundary condition of dY/dz = 0
% at exit

% (3) Eliminate overall mass balance, use 2nd component mass balance, so
% that I do not need a boundary condition for rhog (there is no defined
% rhog at inlet or exit), and I can use flux conservation formulation of
% Q throughout, and rhogbar = Y1bar + Y2bar.

% (4) Gas compositions y1feed y2feed are specified, rhogbar(feed end) =
% Y1bar + Y2bar is not. But I need the right BCs for Y1bar, Y2bar,
% Thetabarg, which depend on rhogbar(feed end).

% (5) Normally Pbar(i) is calculated from rhogbar(i) using ideal gas law.
% But at feed end, rhogbar(feed end) is undetermined. So, Pressure comes
% in to help by using its previous dP/dz and Pbar to calculate
% Pbar(feed end), then calculate rhogbar(feed end) from ideal gas law,
% then calculate the right BCs for Y1bar, Y2bar, Thetabarg.

% (6) Correct use of: epsilon (gas phase accumulation), epsilonbar(Ergun,
% axial dispersions), mass and thermal axial dispersions, gas-solid heat
% transfer (Kunii/Wakao) from literature values (Rege and Yang 1997
% provides Cs, Cg, q1, q2, isotherms), adsorbent density from bed
% dimensions and 160 g LiX in Invacare test unit.

% (7) Only allow average velocity,uz = us/epsilon, and mass flux, Q, to
% acquire positive or negative value, all the rest, y1, y2, n1, n2, Tg, Ts,
% P, rhog, must use positive values
%...........................................................................
function yt=psa_1_sequence_model_EF(t,Y)
%...........................................................................
% Parameters shared with the ODE routine
%...........................................................................
global  Mg1 Mg2 adsorptionORpurge dYbardZbarDifferentialMethod ......
    ncall n dzbar dtbar miu epsilon epsilonbar k1 k2 q1 q2 Cs Cg rhob R....
    dp L tfeed y1feed y2feed Qin Qfeed P0 Pout Tfeed Pmax Pmin Tmax Tmin...
    Tref Q0 kg MassAxial HeatAxial HeateqmORnoneqm Dem4 Dem1
%...........................................................................
% One vector to six vectors
%...........................................................................
for i = 1:n
    Y1bar(i)    = Y(0*n+i);
    n1bar(i)    = Y(1*n+i);
    n2bar(i)    = Y(2*n+i);
    Thetabarg(i)= Y(3*n+i);
    Thetabars(i)= Y(4*n+i);
    Y2bar(i)    = Y(5*n+i);
end

for i = 1:n
    y1bar(i)    = Y1bar(i)/(Y1bar(i)+Y2bar(i));
    y2bar(i)    = Y2bar(i)/(Y1bar(i)+Y2bar(i));
    rhogbar(i)  = Y1bar(i) + Y2bar(i);
    Mg(i)       = Mg1*y1bar(i) + Mg2*(1-y1bar(i));
    thetabarg(i)= Thetabarg(i)/rhogbar(i);
    thetabars(i)= Thetabars(i);
end
%...........................................................................
% Boundary conditions after initial conditions
%...........................................................................
```

```
if (ncall~=0)       % not initial condition

if adsorptionORpurge == 111
   i = 1;           % Only use boundary conditions at the adsorption inlet
end
if adsorptionORpurge == 222
   i = n;           % Only use boundary conditions at the purge inlet
end

y1bar(i)         = y1feed;
y2bar(i)         = y2feed;
Y1bar(i)         = y1feed*rhogbar(i);
Y2bar(i)         = y2feed*rhogbar(i);
thetabarg(i)     = (Tfeed-Tref)/(Tmax-Tmin);
Thetabarg(i)     = ((Tfeed-Tref)/(Tmax-Tmin))*rhogbar(i);
%thetabars(i)     = (Tfeed-Tref)/(Tmax-Tmin);
%Thetabars(i)     = (Tfeed-Tref)/(Tmax-Tmin);
% rhogbar(feed end) needs to be taken care of by Pbar(feed end)
end
%..........................................................................
% Formulate pre-parameters for ODEs
%..........................................................................
% Formulate Pbar(i) using ideal gas law except that at inlet point
%..........................................................................
if (ncall==0)
    for i = 1:n
        Pbar(i) = P0/(Pmax-Pmin);
    end
end

if (ncall~=0)       % not initial condition
%..........................................................................
    if adsorptionORpurge == 111
        i = n;          % BC for adsorption step
        Pbar(i) = Pout/(Pmax-Pmin);

        for i = n-1:-1:2
            Pbar(i) = rhogbar(i)*(thetabarg(i)*(Tmax-Tmin)+Tref)*(R*.......
                Qfeed*tfeed)/((Pmax-Pmin)*L);
            dPbardzbar(i) = (Pbar(i+1)-Pbar(i))/dzbar;
        end

        % I cannot depend on rhogbar(feed end). dP/dz comes in to help
        for i = 1
            Pbar(i)       = Pbar(i+1) - dPbardzbar(i+1)*dzbar;
            rhogbar(i)    = Pbar(i)*(Pmax-Pmin)*L/((thetabarg(i)*(Tmax-....
                Tmin)+Tref)*R*Qfeed*tfeed);
            dPbardzbar(i) = (Pbar(i+1)-Pbar(i))/dzbar;
            Y1bar(i)      = y1feed*rhogbar(i);
            Y2bar(i)      = y2feed*rhogbar(i);
            Thetabarg(i)  = ((Tfeed-Tref)/(Tmax-Tmin))*rhogbar(i);
        end

        for i = n           % special upwinding
            dPbardzbar(i) = (Pbar(i)-Pbar(i-1))/dzbar;
        end
    end
%..........................................................................
    if adsorptionORpurge == 222
        i = 1;          % BC for purge step
        Pbar(i) = Pout/(Pmax-Pmin);

        for i = 2:n-1
            Pbar(i) = rhogbar(i)*(thetabarg(i)*(Tmax-Tmin)+Tref)*(R*.......
                Qfeed*tfeed)/((Pmax-Pmin)*L);
            dPbardzbar(i) = (Pbar(i)-Pbar(i-1))/dzbar;
        end

        % I cannot depend on rhogbar(feed end). dP/dz comes in to help
        for i = n
            Pbar(i)        = dPbardzbar(i-1)*dzbar + Pbar(i-1);
            rhogbar(i)     = Pbar(i)*(Pmax-Pmin)*L/((thetabarg(i)*(Tmax-....
                Tmin)+Tref)*R*Qfeed*tfeed);
            dPbardzbar(i)  = (Pbar(i)-Pbar(i-1))/dzbar;
            Y1bar(i)       = y1feed*rhogbar(i);
            Y2bar(i)       = y2feed*rhogbar(i);
            Thetabarg(i)   = ((Tfeed-Tref)/(Tmax-Tmin))*rhogbar(i);
        end

        for i = 1            % special upwinding
            dPbardzbar(i) = (Pbar(i+1)-Pbar(i))/dzbar;
        end
    end
%.............................................................................
end
%.............................................................................
% Solving for usbar(i) in Ergun equation using quadratic formula - failed
% Solving for Qbar(i) in Ergun equation using quadratic formula
%.............................................................................
if (ncall==0)        % initial condition
    for i = 1:n
        Qbar(i) = Q0/Qfeed;
    end
end

if (ncall~=0)
    if adsorptionORpurge == 111
        Qbar(1) = Qin/Qfeed;
        for i = 2:n
            a = 1.75*Mg(i)*L*(1-epsilonbar)/(dp*rhogbar(i)*epsilonbar^2);
            b = 150*miu*L*(1-epsilonbar)^2/((dp^2)*Qfeed*rhogbar(i)*.......
                epsilonbar^2);
            c = epsilonbar*tfeed*(Pmax-Pmin)*dPbardzbar(i)/(Qfeed*L);
            solQbar1 = (-b + sqrt(b^2-4*a*c))/(2*a);
            solQbar2 = (-b - sqrt(b^2-4*a*c))/(2*a);
            Qbar(i) = max(solQbar1,solQbar2);
        end
    end

    if adsorptionORpurge == 222
        Qbar(n) = Qin/Qfeed;
        for i = 1:n-1
            a = -1.75*Mg(i)*L*(1-epsilonbar)/(dp*rhogbar(i)*epsilonbar^2);
            b = 150*miu*L*(1-epsilonbar)^2/((dp^2)*Qfeed*rhogbar(i)*.......
                epsilonbar^2);
            c = epsilonbar*tfeed*(Pmax-Pmin)*dPbardzbar(i)/(Qfeed*L);
            solQbar1 = (-b + sqrt(b^2-4*a*c))/(2*a);
            solQbar2 = (-b - sqrt(b^2-4*a*c))/(2*a);
            Qbar(i) = min(solQbar1,solQbar2);
        end
    end
end
%..........................................................................
% Formulate usbar(i) = Qbar(i)/rhogbar(i)
%..........................................................................
if (ncall==0)       % initial condition
    for i = 1:n     % position
        usbar(i) = Q0/Qfeed/rhogbar(i);
    end
end

if (ncall~=0)
    if adsorptionORpurge == 111
        i = 1;          % BC for adsorption step
        usbar(i) = Qin/Qfeed/rhogbar(i);

        for i = 2:n
            usbar(i) = Qbar(i)/rhogbar(i);
        end
    end

    if adsorptionORpurge == 222
        i = n;          % BC for purge step
        usbar(i) = Qin/Qfeed/rhogbar(i);

        for i = n-1:-1:1
            usbar(i) = Qbar(i)/rhogbar(i);
        end
    end
end
%..........................................................................
% Formulate gas phase mass axial dispersion coefficient in mass balance,
% DL, cm2/sec
%..........................................................................
if MassAxial == 881
    for i = 1:n
        DL(i) = 0;
    end
end

if MassAxial == 882
    if adsorptionORpurge == 111
        DM = Dem4;
    end

    if adsorptionORpurge == 222
        DM = Dem1;
    end

    for i = 1:n
        DL(i) = 0.7*DM + 0.5*dp*(abs(Qbar(i))*Qfeed)/(rhogbar(i)*..........
            (Qfeed*tfeed/L)*epsilonbar);
    end
end
%..........................................................................
% Formulate volumetric gas-solid heat transfer coefficient, ha
%..........................................................................
if HeatAxial == 991  % Kunni and Suzuki
for i = 1:n
    if HeateqmORnoneqm == 993
```

```
        % Kunii and Suzuki 1967, for packed bed, no need to use thermal
        % axial dispersion, cal/cm2/sec/K
        h = (0.032*((Cg*miu/(Mg(i)*kg))*(Qfeed*abs(Qbar(i))*dp*Mg(i)/......
            miu))^(1.5))*(kg/dp);

        % ha, cal/cm3/sec/K, assuming sphere particle, a = available
        % surface area/bed volume
        ha(i) = 6*(1-epsilonbar)*h/dp;
    end

    if HeateqmORnoneqm == 994
        ha(i) = 2000; % for equilibrium gas-solid heat transfer
    end
%...........................................................................
% Formulate thermal axial dispersion coefficient, Dg, Ds, cm2/sec
%...........................................................................
        Dg(i) = 0; % Gas phase thermal diffusivity, cm2/sec

        % Ds = ks/(rhop*Cs);  % Solid phase thermal diffusivity, cm2/sec
        Ds = 0;               % Absence of solid phase thermal diffusivity
%...........................................................................
end
end
%...........................................................................
if HeatAxial == 992  % Wakao
for i = 1:n
    if HeateqmORnoneqm == 993
        % Ranz & Marshall 1952, for single particle, need to use thermal
        % axial dispersion, cal/cm2/sec/K
        % h = (2+0.6*((Cg*miu/(Mg(i)*kg))^(1/3))*((Qfeed*abs(Qbar(i))*.....
        % dp*Mg(i)/(epsilonbar*miu))^(0.5)))*kg/dp;

        % Wakao 1979, for packed bed, need to use thermal axial dispersion,
        % cal/cm2/sec/K
        h = (2+1.1*((Cg*miu/(Mg(i)*kg))^(1/3))*((Qfeed*abs(Qbar(i))*.......
            dp*Mg(i)/miu)^(0.6)))*kg/dp;

        % ha, cal/cm3/sec/K, assuming sphere particle, a = available
        % surface area/bed volume
        ha(i) = 6*(1-epsilonbar)*h/dp;
%...........................................................................
% Formulate thermal axial dispersion coefficient, Dg, Ds, cm2/sec
%...........................................................................
        % Gas phase thermal diffusivity, cm2/sec
        Dg(i) = kg/(rhogbar(i)*(Qfeed*tfeed/L)*Cg);

        % Ds = ks/(rhop*Cs);  % Solid phase thermal diffusivity, cm2/sec
        Ds = 0;               % Absence of solid phase thermal diffusivity
%...........................................................................
    end

    if HeateqmORnoneqm == 994
        ha(i) = 2000; % for equilibrium gas-solid heat transfer
%...........................................................................
% Formulate thermal axial dispersion coefficient, Dg, Ds, cm2/sec
%...........................................................................
        Dg(i) = 0; % Gas phase thermal diffusivity, cm2/sec

        % Ds = ks/(rhop*Cs);  % Solid phase thermal diffusivity, cm2/sec
        Ds = 0;               % Absence of solid phase thermal diffusivity
```

```
%...........................................................................
    end
%...........................................................................
end
end
%...........................................................................
% Formulate n1eqmbar(i), n2eqmbar(i) using isotherms onto LiX adsorbent
% from Rege and Yand 1997
%...........................................................................
for i = 1:n
      n1eqmbar(i) = (k1*rhob*L/Qfeed)*1.11*10^(-3)*exp(1593.0/.............
          ((Tmax-Tmin)*thetabars(i)+Tref))*(y1bar(i))   *((Pmax-Pmin)*......
          Pbar(i)/1013250)/(1 + 1.03*10^(-4)*exp(2061.9/((Tmax-Tmin)*......
          thetabars(i)+Tref))*(y1bar(i))*((Pmax-Pmin)*Pbar(i)/1013250).....
          + 2.07*10^(-4)*exp(2455.5/((Tmax-Tmin)*thetabars(i)+Tref))*......
          (1-y1bar(i))*((Pmax-Pmin)*Pbar(i)/1013250));
      n2eqmbar(i) = (k2*rhob*L/Qfeed)*1.25*10^(-3)*exp(2168.6/.............
          ((Tmax-Tmin)*thetabars(i)+Tref))*(1-y1bar(i))*((Pmax-Pmin)*......
          Pbar(i)/1013250)/(1 + 1.03*10^(-4)*exp(2061.9/((Tmax-Tmin)*......
          thetabars(i)+Tref))*(y1bar(i))*((Pmax-Pmin)*Pbar(i)/1013250).....
          + 2.07*10^(-4)*exp(2455.5/((Tmax-Tmin)*thetabars(i)+Tref))*......
          (1-y1bar(i))*((Pmax-Pmin)*Pbar(i)/1013250));
end
%...........................................................................
```

```
%...........................................................................
% Start of 1st order (2 points) upwind difference
if dYbardZbarDifferentialMethod == 333
%...........................................................................
% This Flux Limiter formulation is valid for any regularly or non-regularly
% spaced grid, and for any constant or varying advection velocity
% Dullemon, Numerical Fluid Dynamics - Lecture Notes, Chapter 4
%...........................................................................
% Using linear terms y1bar, y2bar, thetabarg, and mass flux Q
%...........................................................................
% Formulate convective terms for i = 3:n-2
%...........................................................................
for i = 3:n-2
    % Le = left edge of cell
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    % Re = right edge of cell
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;

        ry1barLe     = (y1bar(i-1)      - y1bar(i-2))    /(y1bar(i)      -...
            y1bar(i-1));
        ry2barLe     = (y2bar(i-1)      - y2bar(i-2))    /(y2bar(i)      -...
            y2bar(i-1));
        rthetabargLe = (thetabarg(i-1) - thetabarg(i-2))/(thetabarg(i) -...
            thetabarg(i-1));

    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
            y1bar(i-1));
        ry2barLe     = (y2bar(i+1)     - y2bar(i))    /(y2bar(i)     -.....
            y2bar(i-1));
        rthetabargLe = (thetabarg(i+1) - thetabarg(i))/(thetabarg(i) -.....
            thetabarg(i-1));
    end

    superbeey1barLe     = 0;
    superbeey2barLe     = 0;
    superbeethetabargLe = 0;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     +.............
        0.5*abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...........
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))     +.............
        0.5*abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...........
        superbeey2barLe*    (y2bar(i)     - y2bar(i-1));

    fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        thetabarg(i-1) + (1 - flowdirectionLe)*thetabarg(i)) +.............
        0.5*abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...........
        superbeethetabargLe*(thetabarg(i) - thetabarg(i-1));

    % Directional velocity has been taken care of in this flux formulation,
    % please do not change (Q) into abs(Q) anymore

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;

        ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
            y1bar(i));
        ry2barRe     = (y2bar(i)     - y2bar(i-1))    /(y2bar(i+1)     -...
            y2bar(i));
        rthetabargRe = (thetabarg(i) - thetabarg(i-1))/(thetabarg(i+1) -...
            thetabarg(i));

    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;

        ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /(y1bar(i+1).......
            - y1bar(i));
        ry2barRe     = (y2bar(i+2)     - y2bar(i+1))    /(y2bar(i+1).......
            - y2bar(i));
        rthetabargRe = (thetabarg(i+2) - thetabarg(i+1))/(thetabarg(i+1)...
            - thetabarg(i));
    end

    superbeey1barRe     = 0;
    superbeey2barRe     = 0;
    superbeethetabargRe = 0;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     +.............
        0.5*abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...........
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));
    fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))     +.............
        0.5*abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*..........
        superbeey2barRe*    (y2bar(i+1)     - y2bar(i));

    fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i+1)) +.............
        0.5*abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*..........
        superbeethetabargRe*(thetabarg(i+1) - thetabarg(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
    convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
end
%...........................................................................
% Formulate convective terms for i = 1, 2, n-1, n
%...........................................................................
for i = 1
    Qbar_eps_Le = (Qbar(i) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = 1;
        ry2barLe     = 1;
        rthetabargLe = 1;
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = 1;
        ry2barLe     = 1;
        rthetabargLe = 1;
    end

    superbeey1barLe     = 0;
    superbeey2barLe     = 0;
    superbeethetabargLe = 0;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*..........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i));

    fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y2bar(i)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*..........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey2barLe*    (y2bar(i)     - y2bar(i));

    fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        thetabarg(i) + (1 - flowdirectionLe)*thetabarg(i)) + 0.5*..........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeethetabargLe*(thetabarg(i) - thetabarg(i));

    % Directional velocity has been taken care of in this flux formulation,
    % please do not change (Q) into abs(Q) anymore

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = 0;
        ry2barRe     = 0;
        rthetabargRe = 0;
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /(y1bar(i+1).......
            - y1bar(i));
        ry2barRe     = (y2bar(i+2)     - y2bar(i+1))    /(y2bar(i+1).......
            - y2bar(i));
        rthetabargRe = (thetabarg(i+2) - thetabarg(i+1))/(thetabarg(i+1)...
            - thetabarg(i));
    end

    superbeey1barRe     = 0;
    superbeey2barRe     = 0;
    superbeethetabargRe = 0;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey2barRe*    (y2bar(i+1)     - y2bar(i));

    fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i+1)) + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeethetabargRe*(thetabarg(i+1) - thetabarg(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
    convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
end
%..........................................................................
for i = 2
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = 0;
        ry2barLe     = 0;
        rthetabargLe = 0;
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
            y1bar(i-1));
        ry2barLe     = (y2bar(i+1)     - y2bar(i))    /(y2bar(i)     -.....
            y2bar(i-1));
        rthetabargLe = (thetabarg(i+1) - thetabarg(i))/(thetabarg(i) -.....
            thetabarg(i-1));
    end

    superbeey1barLe     = 0;
    superbeey2barLe     = 0;
    superbeethetabargLe = 0;
```

```
fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
    y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
    superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
    y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*........
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
    superbeey2barLe*    (y2bar(i)     - y2bar(i-1));

fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
    thetabarg(i-1) + (1 - flowdirectionLe)*thetabarg(i)) + 0.5*........
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
    superbeethetabargLe*(thetabarg(i) - thetabarg(i-1));

% for right edge i+1/2
if Qbar_eps_Re >= 0
    flowdirectionRe = 1;

    ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
        y1bar(i));
    ry2barRe     = (y2bar(i)     - y2bar(i-1))    /(y2bar(i+1)     -...
        y2bar(i));
    rthetabargRe = (thetabarg(i) - thetabarg(i-1))/(thetabarg(i+1) -...
        thetabarg(i));

elseif Qbar_eps_Re <= 0
    flowdirectionRe = -1;

    ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /(y1bar(i+1).......
        - y1bar(i));
    ry2barRe     = (y2bar(i+2)     - y2bar(i+1))    /(y2bar(i+1).......
        - y2bar(i));
    rthetabargRe = (thetabarg(i+2) - thetabarg(i+1))/(thetabarg(i+1)...
        - thetabarg(i));
end

superbeey1barRe     = 0;
superbeey2barRe     = 0;
superbeethetabargRe = 0;

fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
    y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
    abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
    superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
    y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))     + 0.5*........
    abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
    superbeey2barRe*    (y2bar(i+1)     - y2bar(i));

fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
    thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i+1)) + 0.5*........
    abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
    superbeethetabargRe*(thetabarg(i+1) - thetabarg(i));

convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
```

```
end
%...........................................................................
for i = n-1
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));
        ry2barLe     = (y2bar(i-1)     - y2bar(i-2))    /(y2bar(i)     -...
            y2bar(i-1));
        rthetabargLe = (thetabarg(i-1) - thetabarg(i-2))/(thetabarg(i) -...
            thetabarg(i-1));
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
            y1bar(i-1));
        ry2barLe     = (y2bar(i+1)     - y2bar(i))    /(y2bar(i)     -.....
            y2bar(i-1));
        rthetabargLe = (thetabarg(i+1) - thetabarg(i))/(thetabarg(i) -.....
            thetabarg(i-1));
    end

    superbeey1barLe     = 0;
    superbeey2barLe     = 0;
    superbeethetabargLe = 0;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey2barLe*    (y2bar(i)     - y2bar(i-1));

    fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        thetabarg(i-1) + (1 - flowdirectionLe)*thetabarg(i)) + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeethetabargLe*(thetabarg(i) - thetabarg(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
            y1bar(i));
        ry2barRe     = (y2bar(i)     - y2bar(i-1))    /(y2bar(i+1)     -...
            y2bar(i));
        rthetabargRe = (thetabarg(i) - thetabarg(i-1))/(thetabarg(i+1) -...
            thetabarg(i));
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = 0;
        ry2barRe     = 0;
        rthetabargRe = 0;
    end
    superbeey1barRe     = 0;
    superbeey2barRe     = 0;
    superbeethetabargRe = 0;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey2barRe*    (y2bar(i+1)     - y2bar(i));

    fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i+1)) + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeethetabargRe*(thetabarg(i+1) - thetabarg(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
    convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
end
%..........................................................................
for i = n
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;

        ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));
        ry2barLe     = (y2bar(i-1)     - y2bar(i-2))    /(y2bar(i)     -...
            y2bar(i-1));
        rthetabargLe = (thetabarg(i-1) - thetabarg(i-2))/(thetabarg(i) -...
            thetabarg(i-1));

    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = 0;
        ry2barLe     = 0;
        rthetabargLe = 0;
    end

    superbeey1barLe     = 0;
    superbeey2barLe     = 0;
    superbeethetabargLe = 0;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey2barLe*    (y2bar(i)     - y2bar(i-1));
    fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        thetabarg(i-1) + (1 - flowdirectionLe)*thetabarg(i)) + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeethetabargLe*(thetabarg(i) - thetabarg(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = 1;
        ry2barRe     = 1;
        rthetabargRe = 1;
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = 1;
        ry2barRe     = 1;
        rthetabargRe = 1;
    end

    superbeey1barRe     = 0;
    superbeey2barRe     = 0;
    superbeethetabargRe = 0;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i))     + 0.5*..........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i)     - y1bar(i));

    fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y2bar(i)     + (1 - flowdirectionRe)*y2bar(i))     + 0.5*..........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey2barRe*    (y2bar(i)     - y2bar(i));

    fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i)) + 0.5*..........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeethetabargRe*(thetabarg(i) - thetabarg(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
    convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
end
%..........................................................................
end % end of dYbardZbarDifferentialMethod == 333
%..........................................................................
%..........................................................................
% Start of 2nd order (3 points) upwind difference
if dYbardZbarDifferentialMethod == 444
%..........................................................................
% This Flux Limiter formula is valid for any regularly or non-regularly
% spaced grid, and for any constant or varying advection velocity
% Dullemon, Numerical Fluid Dynamics - Lecture Notes, Chapter 4
%..........................................................................
% Formulate convective terms for i = 3:n-2
%..........................................................................
for i = 3:n-2

    % Le = left edge of cell
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    % Re = right edge of cell
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);
```

```
% for left edge i-1/2
if Qbar_eps_Le >= 0
    flowdirectionLe = 1;
    ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
        y1bar(i-1));
    ry2barLe     = (y2bar(i-1)     - y2bar(i-2))    /(y2bar(i)     -...
        y2bar(i-1));
    rthetabargLe = (thetabarg(i-1) - thetabarg(i-2))/(thetabarg(i) -...
        thetabarg(i-1));
elseif Qbar_eps_Le <= 0
    flowdirectionLe = -1;
    ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
        y1bar(i-1));
    ry2barLe     = (y2bar(i+1)     - y2bar(i))    /(y2bar(i)     -.....
        y2bar(i-1));
    rthetabargLe = (thetabarg(i+1) - thetabarg(i))/(thetabarg(i) -.....
        thetabarg(i-1));
end

superbeey1barLe     = 1;
superbeey2barLe     = 1;
superbeethetabargLe = 1;

fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
    y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*.......
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*..............
    superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
    y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*.......
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*..............
    superbeey2barLe*    (y2bar(i)     - y2bar(i-1));

fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
    thetabarg(i-1) + (1 - flowdirectionLe)*thetabarg(i)) + 0.5*.......
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*..............
    superbeethetabargLe*(thetabarg(i) - thetabarg(i-1));

% for right edge i+1/2
if Qbar_eps_Re >= 0
    flowdirectionRe = 1;
    ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
        y1bar(i));
    ry2barRe     = (y2bar(i)     - y2bar(i-1))    /(y2bar(i+1)     -...
        y2bar(i));
    rthetabargRe = (thetabarg(i) - thetabarg(i-1))/(thetabarg(i+1) -...
        thetabarg(i));
elseif Qbar_eps_Re <= 0
    flowdirectionRe = -1;
    ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /.................
        (y1bar(i+1)     - y1bar(i));
    ry2barRe     = (y2bar(i+2)     - y2bar(i+1))    /.................
        (y2bar(i+1)     - y2bar(i));
    rthetabargRe = (thetabarg(i+2) - thetabarg(i+1))/.................
        (thetabarg(i+1) - thetabarg(i));
end

superbeey1barRe     = 1;
superbeey2barRe     = 1;
```

```
    superbeethetabargRe = 1;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*.......
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))     + 0.5*.......
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey2barRe*    (y2bar(i+1)     - y2bar(i));

    fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i+1)) + 0.5*.......
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeethetabargRe*(thetabarg(i+1) - thetabarg(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
    convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
end
%..........................................................................
% Formulate convective terms for i = 1, 2, n-1, n
%..........................................................................
for i = 1
    Qbar_eps_Le = (Qbar(i) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = 1;
        ry2barLe     = 1;
        rthetabargLe = 1;
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = 1;
        ry2barLe     = 1;
        rthetabargLe = 1;
    end

    superbeey1barLe     = 1;
    superbeey2barLe     = 1;
    superbeethetabargLe = 1;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*..........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i));

    fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y2bar(i)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*..........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey2barLe*    (y2bar(i)     - y2bar(i));

    fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        thetabarg(i) + (1 - flowdirectionLe)*thetabarg(i)) + 0.5*..........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeethetabargLe*(thetabarg(i) - thetabarg(i));
    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = 0;
        ry2barRe     = 0;
        rthetabargRe = 0;
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /(y1bar(i+1).......
            - y1bar(i));
        ry2barRe     = (y2bar(i+2)     - y2bar(i+1))    /(y2bar(i+1).......
            - y2bar(i));
        rthetabargRe = (thetabarg(i+2) - thetabarg(i+1))/(thetabarg(i+1)...
            - thetabarg(i));
    end

    superbeey1barRe     = 1;
    superbeey2barRe     = 1;
    superbeethetabargRe = 1;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey2barRe*    (y2bar(i+1)     - y2bar(i));

    fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i+1)) + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeethetabargRe*(thetabarg(i+1) - thetabarg(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
    convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
end
%..........................................................................
for i = 2
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = 0;
        ry2barLe     = 0;
        rthetabargLe = 0;
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
            y1bar(i-1));
        ry2barLe     = (y2bar(i+1)     - y2bar(i))    /(y2bar(i)     -.....
            y2bar(i-1));
        rthetabargLe = (thetabarg(i+1) - thetabarg(i))/(thetabarg(i) -.....
            thetabarg(i-1));
    end
```

```
superbeey1barLe     = 1;
superbeey2barLe     = 1;
superbeethetabargLe = 1;

fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
    y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))      + 0.5*.......
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
    superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
    y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))      + 0.5*.......
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
    superbeey2barLe*    (y2bar(i)     - y2bar(i-1));

fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
    thetabarg(i-1) + (1 - flowdirectionLe)*thetabarg(i)) + 0.5*.......
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
    superbeethetabargLe*(thetabarg(i) - thetabarg(i-1));

% for right edge i+1/2
if Qbar_eps_Re >= 0
    flowdirectionRe = 1;
    ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
        y1bar(i));
    ry2barRe     = (y2bar(i)     - y2bar(i-1))    /(y2bar(i+1)     -...
        y2bar(i));
    rthetabargRe = (thetabarg(i) - thetabarg(i-1))/(thetabarg(i+1) -...
        thetabarg(i));

elseif Qbar_eps_Re <= 0
    flowdirectionRe = -1;
    ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /..................
        (y1bar(i+1)     - y1bar(i));
    ry2barRe     = (y2bar(i+2)     - y2bar(i+1))    /..................
        (y2bar(i+1)     - y2bar(i));
    rthetabargRe = (thetabarg(i+2) - thetabarg(i+1))/..................
        (thetabarg(i+1) - thetabarg(i));
end

superbeey1barRe     = 1;
superbeey2barRe     = 1;
superbeethetabargRe = 1;

fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
    y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*.......
    abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
    superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
    y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))     + 0.5*.......
    abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
    superbeey2barRe*    (y2bar(i+1)     - y2bar(i));

fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
    thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i+1)) + 0.5*.......
    abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
    superbeethetabargRe*(thetabarg(i+1) - thetabarg(i));

convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
```

```
    convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
end
%..........................................................................
for i = n-1
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));
        ry2barLe     = (y2bar(i-1)     - y2bar(i-2))    /(y2bar(i)     -...
            y2bar(i-1));
        rthetabargLe = (thetabarg(i-1) - thetabarg(i-2))/(thetabarg(i) -...
            thetabarg(i-1));
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
            y1bar(i-1));
        ry2barLe     = (y2bar(i+1)     - y2bar(i))    /(y2bar(i)     -.....
            y2bar(i-1));
        rthetabargLe = (thetabarg(i+1) - thetabarg(i))/(thetabarg(i) -.....
            thetabarg(i-1));
    end

    superbeey1barLe     = 1;
    superbeey2barLe     = 1;
    superbeethetabargLe = 1;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey2barLe*    (y2bar(i)     - y2bar(i-1));

    fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        thetabarg(i-1) + (1 - flowdirectionLe)*thetabarg(i)) + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeethetabargLe*(thetabarg(i) - thetabarg(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
            y1bar(i));
        ry2barRe     = (y2bar(i)     - y2bar(i-1))    /(y2bar(i+1)     -...
            y2bar(i));
        rthetabargRe = (thetabarg(i) - thetabarg(i-1))/(thetabarg(i+1) -...
            thetabarg(i));

    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = 0;
        ry2barRe     = 0;
        rthetabargRe = 0;
    end

    superbeey1barRe     = 1;
    superbeey2barRe     = 1;
    superbeethetabargRe = 1;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey2barRe*    (y2bar(i+1)     - y2bar(i));

    fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i+1)) + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeethetabargRe*(thetabarg(i+1) - thetabarg(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
    convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
end
%..........................................................................
for i = n
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
            y1bar(i-1));
        ry2barLe     = (y2bar(i-1)     - y2bar(i-2))    /(y2bar(i)     -...
            y2bar(i-1));
        rthetabargLe = (thetabarg(i-1) - thetabarg(i-2))/(thetabarg(i) -...
            thetabarg(i-1));
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = 0;
        ry2barLe     = 0;
        rthetabargLe = 0;
    end

    superbeey1barLe     = 1;
    superbeey2barLe     = 1;
    superbeethetabargLe = 1;

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

    fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey2barLe*    (y2bar(i)     - y2bar(i-1));
    fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        thetabarg(i-1) + (1 - flowdirectionLe)*thetabarg(i)) + 0.5*........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeethetabargLe*(thetabarg(i) - thetabarg(i-1));

    % for right edge i+1/2
    if Qbar_eps_Re >= 0
        flowdirectionRe = 1;
        ry1barRe     = 1;
        ry2barRe     = 1;
        rthetabargRe = 1;
    elseif Qbar_eps_Re <= 0
        flowdirectionRe = -1;
        ry1barRe     = 1;
        ry2barRe     = 1;
        rthetabargRe = 1;
    end

    superbeey1barRe     = 1;
    superbeey2barRe     = 1;
    superbeethetabargRe = 1;

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i))     + 0.5*..........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i)     - y1bar(i));

    fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y2bar(i)     + (1 - flowdirectionRe)*y2bar(i))     + 0.5*..........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey2barRe*    (y2bar(i)     - y2bar(i));

    fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i)) + 0.5*..........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeethetabargRe*(thetabarg(i) - thetabarg(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
    convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
end
%..........................................................................
end % end of dYbardZbarDifferentialMethod == 444
%..........................................................................
%..........................................................................
% Start of Superbee Flux Limiter
if dYbardZbarDifferentialMethod == 777
%..........................................................................
% This Flux Limiter formula is valid for any regularly or non-regularly
% spaced grid, and for any constant or varying advection velocity
% Dullemon, Numerical Fluid Dynamics - Lecture Notes, Chapter 4
%..........................................................................
% Formulate convective terms for i = 3:n-2
%..........................................................................
for i = 3:n-2
    % Le = left edge of cell
    Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
    % Re = right edge of cell
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);
```

```
% for left edge i-1/2
if Qbar_eps_Le >= 0
    flowdirectionLe = 1;
    ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
        y1bar(i-1));
    ry2barLe     = (y2bar(i-1)     - y2bar(i-2))    /(y2bar(i)     -...
        y2bar(i-1));
    rthetabargLe = (thetabarg(i-1) - thetabarg(i-2))/(thetabarg(i) -...
        thetabarg(i-1));
elseif Qbar_eps_Le <= 0
    flowdirectionLe = -1;
    ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
        y1bar(i-1));
    ry2barLe     = (y2bar(i+1)     - y2bar(i))    /(y2bar(i)     -.....
        y2bar(i-1));
    rthetabargLe = (thetabarg(i+1) - thetabarg(i))/(thetabarg(i) -.....
        thetabarg(i-1));
end

superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,     ............
    min(2,ry1barLe)));
superbeey2barLe     = max(0 , max(min(1,2*ry2barLe) ,     ............
    min(2,ry2barLe)));
superbeethetabargLe = max(0 , max(min(1,2*rthetabargLe) , ............
    min(2,rthetabargLe)));

fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*.............
    y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*.......
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*..............
    superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*.............
    y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*.......
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*..............
    superbeey2barLe*    (y2bar(i)     - y2bar(i-1));

fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*.............
    thetabarg(i-1) + (1 - flowdirectionLe)*thetabarg(i)) + 0.5*.......
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*..............
    superbeethetabargLe*(thetabarg(i) - thetabarg(i-1));

% for right edge i+1/2
if Qbar_eps_Re >= 0
    flowdirectionRe = 1;
    ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
        y1bar(i));
    ry2barRe     = (y2bar(i)     - y2bar(i-1))    /(y2bar(i+1)     -...
        y2bar(i));
    rthetabargRe = (thetabarg(i) - thetabarg(i-1))/(thetabarg(i+1) -...
        thetabarg(i));
elseif Qbar_eps_Re <= 0
    flowdirectionRe = -1;
    ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /.................
        (y1bar(i+1)     - y1bar(i));
    ry2barRe     = (y2bar(i+2)     - y2bar(i+1))    /.................
        (y2bar(i+1)     - y2bar(i));
    rthetabargRe = (thetabarg(i+2) - thetabarg(i+1))/.................
        (thetabarg(i+1) - thetabarg(i));
end
```

```
    superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,      .............
        min(2,ry1barRe)));
    superbeey2barRe     = max(0 , max(min(1,2*ry2barRe) ,      .............
        min(2,ry2barRe)));
    superbeethetabargRe = max(0 , max(min(1,2*rthetabargRe) , .............
        min(2,rthetabargRe)));

    fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

    fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))     + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeey2barRe*    (y2bar(i+1)     - y2bar(i));

    fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i+1)) + 0.5*........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeethetabargRe*(thetabarg(i+1) - thetabarg(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
    convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
end
%..........................................................................
% Formulate convective terms for i = 1, 2, n-1, n
%..........................................................................
for i = 1
    Qbar_eps_Le = (Qbar(i) + Qbar(i))/(2*epsilon);
    Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

    % for left edge i-1/2
    if Qbar_eps_Le >= 0
        flowdirectionLe = 1;
        ry1barLe     = 1;
        ry2barLe     = 1;
        rthetabargLe = 1;
    elseif Qbar_eps_Le <= 0
        flowdirectionLe = -1;
        ry1barLe     = 1;
        ry2barLe     = 1;
        rthetabargLe = 1;
    end

    superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,      .............
        min(2,ry1barLe)));
    superbeey2barLe     = max(0 , max(min(1,2*ry2barLe) ,      .............
        min(2,ry2barLe)));
    superbeethetabargLe = max(0 , max(min(1,2*rthetabargLe) , .............
        min(2,rthetabargLe)));

    fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y1bar(i)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*..........
        abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
        superbeey1barLe*    (y1bar(i)     - y1bar(i));

    fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
        y2bar(i)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*..........
```

```
            abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
            superbeey2barLe*    (y2bar(i)     - y2bar(i));

        fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
            thetabarg(i) + (1 - flowdirectionLe)*thetabarg(i)) + 0.5*..........
            abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
            superbeethetabargLe*(thetabarg(i) - thetabarg(i));

        % for right edge i+1/2
        if Qbar_eps_Re >= 0
            flowdirectionRe = 1;
            ry1barRe     = 0;
            ry2barRe     = 0;
            rthetabargRe = 0;
        elseif Qbar_eps_Re <= 0
            flowdirectionRe = -1;
            ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /..................
                (y1bar(i+1)     - y1bar(i));
            ry2barRe     = (y2bar(i+2)     - y2bar(i+1))    /..................
                (y2bar(i+1)     - y2bar(i));
            rthetabargRe = (thetabarg(i+2) - thetabarg(i+1))/..................
                (thetabarg(i+1) - thetabarg(i));
        end

        superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,     .............
            min(2,ry1barRe)));
        superbeey2barRe     = max(0 , max(min(1,2*ry2barRe) ,     .............
            min(2,ry2barRe)));
        superbeethetabargRe = max(0 , max(min(1,2*rthetabargRe) , .............
            min(2,rthetabargRe)));

        fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
            y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
            abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
            superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

        fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
            y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))     + 0.5*........
            abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
            superbeey2barRe*    (y2bar(i+1)     - y2bar(i));

        fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
            thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i+1)) + 0.5*........
            abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
            superbeethetabargRe*(thetabarg(i+1) - thetabarg(i));

        convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
        convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
        convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
    end
    %..........................................................................
    for i = 2
        Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
        Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

        % for left edge i-1/2
        if Qbar_eps_Le >= 0
            flowdirectionLe = 1;
            ry1barLe     = 0;
            ry2barLe     = 0;
            rthetabargLe = 0;
        elseif Qbar_eps_Le <= 0
            flowdirectionLe = -1;
            ry1barLe     = (y1bar(i+1)     - y1bar(i))    /(y1bar(i)     -.....
                y1bar(i-1));
            ry2barLe     = (y2bar(i+1)     - y2bar(i))    /(y2bar(i)     -.....
                y2bar(i-1));
            rthetabargLe = (thetabarg(i+1) - thetabarg(i))/(thetabarg(i) -.....
                thetabarg(i-1));
        end

        superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,     .............
            min(2,ry1barLe)));
        superbeey2barLe     = max(0 , max(min(1,2*ry2barLe) ,     .............
            min(2,ry2barLe)));
        superbeethetabargLe = max(0 , max(min(1,2*rthetabargLe) , .............
            min(2,rthetabargLe)));

        fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
            y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
            abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
            superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

        fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
            y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*........
            abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
            superbeey2barLe*    (y2bar(i)     - y2bar(i-1));

        fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
            thetabarg(i-1) + (1 - flowdirectionLe)*thetabarg(i)) + 0.5*........
            abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
            superbeethetabargLe*(thetabarg(i) - thetabarg(i-1));

        % for right edge i+1/2
        if Qbar_eps_Re >= 0
            flowdirectionRe = 1;
            ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
                y1bar(i));
            ry2barRe     = (y2bar(i)     - y2bar(i-1))    /(y2bar(i+1)     -...
                y2bar(i));
            rthetabargRe = (thetabarg(i) - thetabarg(i-1))/(thetabarg(i+1) -...
                thetabarg(i));
        elseif Qbar_eps_Re <= 0
            flowdirectionRe = -1;
            ry1barRe     = (y1bar(i+2)     - y1bar(i+1))    /(y1bar(i+1)    ...
                - y1bar(i));
            ry2barRe     = (y2bar(i+2)     - y2bar(i+1))    /(y2bar(i+1)    ...
                - y2bar(i));
            rthetabargRe = (thetabarg(i+2) - thetabarg(i+1))/(thetabarg(i+1)...
                - thetabarg(i));
        end

        superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,     .............
            min(2,ry1barRe)));
        superbeey2barRe     = max(0 , max(min(1,2*ry2barRe) ,     .............
            min(2,ry2barRe)));
        superbeethetabargRe = max(0 , max(min(1,2*rthetabargRe) , .............
            min(2,rthetabargRe)));

        fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
            y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))      + 0.5*........
            abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
            superbeey1barRe*    (y1bar(i+1)      - y1bar(i));

        fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
            y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))      + 0.5*........
            abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
            superbeey2barRe*    (y2bar(i+1)      - y2bar(i));

        fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
            thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i+1)) + 0.5*........
            abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
            superbeethetabargRe*(thetabarg(i+1) - thetabarg(i));

        convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
        convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
        convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
    end
    %..........................................................................
    for i = n-1
        Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
        Qbar_eps_Re = (Qbar(i) + Qbar(i+1))/(2*epsilon);

        % for left edge i-1/2
        if Qbar_eps_Le >= 0
            flowdirectionLe = 1;
            ry1barLe     = (y1bar(i-1)      - y1bar(i-2))    /(y1bar(i)     -...
                y1bar(i-1));
            ry2barLe     = (y2bar(i-1)      - y2bar(i-2))    /(y2bar(i)     -...
                y2bar(i-1));
            rthetabargLe = (thetabarg(i-1) - thetabarg(i-2))/(thetabarg(i) -...
                thetabarg(i-1));
        elseif Qbar_eps_Le <= 0
            flowdirectionLe = -1;
            ry1barLe     = (y1bar(i+1)      - y1bar(i))    /(y1bar(i)     -.....
                y1bar(i-1));
            ry2barLe     = (y2bar(i+1)      - y2bar(i))    /(y2bar(i)     -.....
                y2bar(i-1));
            rthetabargLe = (thetabarg(i+1) - thetabarg(i))/(thetabarg(i) -.....
                thetabarg(i-1));
        end

        superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,      .............
            min(2,ry1barLe)));
        superbeey2barLe     = max(0 , max(min(1,2*ry2barLe) ,      .............
            min(2,ry2barLe)));
        superbeethetabargLe = max(0 , max(min(1,2*rthetabargLe) , .............
            min(2,rthetabargLe)));

        fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
            y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))      + 0.5*........
            abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
            superbeey1barLe*    (y1bar(i)      - y1bar(i-1));

        fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
            y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))      + 0.5*........
            abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
            superbeey2barLe*    (y2bar(i)      - y2bar(i-1));

        fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
            thetabarg(i-1) + (1 - flowdirectionLe)*thetabarg(i)) + 0.5*........
            abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
            superbeethetabargLe*(thetabarg(i) - thetabarg(i-1));

        % for right edge i+1/2
        if Qbar_eps_Re >= 0
            flowdirectionRe = 1;
            ry1barRe     = (y1bar(i)     - y1bar(i-1))    /(y1bar(i+1)     -...
                y1bar(i));
            ry2barRe     = (y2bar(i)     - y2bar(i-1))    /(y2bar(i+1)     -...
                y2bar(i));
            rthetabargRe = (thetabarg(i) - thetabarg(i-1))/(thetabarg(i+1) -...
                thetabarg(i));
        elseif Qbar_eps_Re <= 0
            flowdirectionRe = -1;
            ry1barRe     = 0;
            ry2barRe     = 0;
            rthetabargRe = 0;
        end

        superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,     .............
            min(2,ry1barRe)));
        superbeey2barRe     = max(0 , max(min(1,2*ry2barRe) ,     .............
            min(2,ry2barRe)));
        superbeethetabargRe = max(0 , max(min(1,2*rthetabargRe) , .............
            min(2,rthetabargRe)));

        fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
            y1bar(i)     + (1 - flowdirectionRe)*y1bar(i+1))     + 0.5*........
            abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
            superbeey1barRe*    (y1bar(i+1)     - y1bar(i));

        fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
            y2bar(i)     + (1 - flowdirectionRe)*y2bar(i+1))     + 0.5*........
            abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
            superbeey2barRe*    (y2bar(i+1)     - y2bar(i));

        fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
            thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i+1)) + 0.5*........
            abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
            superbeethetabargRe*(thetabarg(i+1) - thetabarg(i));

        convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
        convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
        convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
    end
    %..........................................................................
    for i = n
        Qbar_eps_Le = (Qbar(i-1) + Qbar(i))/(2*epsilon);
        Qbar_eps_Re = (Qbar(i) + Qbar(i))/(2*epsilon);

        % for left edge i-1/2
        if Qbar_eps_Le >= 0
            flowdirectionLe = 1;
            ry1barLe     = (y1bar(i-1)     - y1bar(i-2))    /(y1bar(i)     -...
                y1bar(i-1));
            ry2barLe     = (y2bar(i-1)     - y2bar(i-2))    /(y2bar(i)     -...
                y2bar(i-1));
            rthetabargLe = (thetabarg(i-1) - thetabarg(i-2))/(thetabarg(i) -...
                thetabarg(i-1));
```

```
elseif Qbar_eps_Le <= 0
    flowdirectionLe = -1;
    ry1barLe      = 0;
    ry2barLe      = 0;
    rthetabargLe = 0;
end

superbeey1barLe     = max(0 , max(min(1,2*ry1barLe) ,     .............
    min(2,ry1barLe)));
superbeey2barLe     = max(0 , max(min(1,2*ry2barLe) ,     .............
    min(2,ry2barLe)));
superbeethetabargLe = max(0 , max(min(1,2*rthetabargLe) , .............
    min(2,rthetabargLe)));

fluxy1barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
    y1bar(i-1)     + (1 - flowdirectionLe)*y1bar(i))     + 0.5*........
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
    superbeey1barLe*    (y1bar(i)     - y1bar(i-1));

fluxy2barLe     = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
    y2bar(i-1)     + (1 - flowdirectionLe)*y2bar(i))     + 0.5*........
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
    superbeey2barLe*    (y2bar(i)     - y2bar(i-1));

fluxthetabargLe = 0.5*Qbar_eps_Le*((1 + flowdirectionLe)*..............
    thetabarg(i-1) + (1 - flowdirectionLe)*thetabarg(i)) + 0.5*........
    abs(Qbar_eps_Le)*(1 - abs(Qbar_eps_Le*dtbar/dzbar))*...............
    superbeethetabargLe*(thetabarg(i) - thetabarg(i-1));

% for right edge i+1/2
if Qbar_eps_Re >= 0
    flowdirectionRe = 1;
    ry1barRe      = 1;
    ry2barRe      = 1;
    rthetabargRe = 1;
elseif Qbar_eps_Re <= 0
    flowdirectionRe = -1;
    ry1barRe      = 1;
    ry2barRe      = 1;
    rthetabargRe = 1;
end

superbeey1barRe     = max(0 , max(min(1,2*ry1barRe) ,     .............
    min(2,ry1barRe)));
superbeey2barRe     = max(0 , max(min(1,2*ry2barRe) ,     .............
    min(2,ry2barRe)));
superbeethetabargRe = max(0 , max(min(1,2*rthetabargRe) , .............
    min(2,rthetabargRe)));

fluxy1barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
    y1bar(i)     + (1 - flowdirectionRe)*y1bar(i))     + 0.5*..........
    abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
    superbeey1barRe*    (y1bar(i)     - y1bar(i));

fluxy2barRe     = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
    y2bar(i)     + (1 - flowdirectionRe)*y2bar(i))     + 0.5*..........
    abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
    superbeey2barRe*    (y2bar(i)     - y2bar(i));

fluxthetabargRe = 0.5*Qbar_eps_Re*((1 + flowdirectionRe)*..............
        thetabarg(i) + (1 - flowdirectionRe)*thetabarg(i)) + 0.5*..........
        abs(Qbar_eps_Re)*(1 - abs(Qbar_eps_Re*dtbar/dzbar))*...............
        superbeethetabargRe*(thetabarg(i) - thetabarg(i));

    convecy1bar(i)     = (fluxy1barLe - fluxy1barRe)/dzbar;
    convecy2bar(i)     = (fluxy2barLe - fluxy2barRe)/dzbar;
    convecthetabarg(i) = (fluxthetabargLe - fluxthetabargRe)/dzbar;
end
%..........................................................................
end % end of dYbardZbarDifferentialMethod == 777
%..........................................................................
%..........................................................................
% Formulate ODEs - can be used for adsorption and purge - both directions
%..........................................................................
% Kunni & Suzuki - do not need thermal axial dispersions in energy balances
% Wakao - need gas phase thermal axial dispersion (need a second BC for Tg)
% Need 2nd boundary condition for 2nd order differences (axial dispersions)

for i=1
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon +.....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y1bar(i+2)-2*Y1bar(i+1)....
        +Y1bar(i))/(dzbar)^2;

    dY2bardtbar(i) = convecy2bar(i) - (n2eqmbar(i)-n2bar(i))/epsilon +.....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y2bar(i+2)-2*Y2bar(i+1)....
        +Y2bar(i))/(dzbar)^2;

    dThetabargdtbar(i) = convecthetabarg(i) + (ha(i)*L/(epsilon*Cg*........
        Qfeed))*(thetabars(i) - thetabarg(i)) + (epsilonbar*Dg(i)*tfeed....
        /(epsilon*L^2))*(Thetabarg(i+2)-2*Thetabarg(i+1)+Thetabarg(i))/....
        (dzbar)^2;

    dThetabarsdtbar(i) = (Qfeed*tfeed/(rhob*Cs*L*(Tmax-Tmin)))*(q1*........
        (n1eqmbar(i)-n1bar(i)) + q2*(n2eqmbar(i)-n2bar(i))) - (ha(i)*......
        tfeed/(rhob*Cs))*(thetabars(i) - thetabarg(i)) + (Ds*tfeed/L^2)....
        *(thetabars(i+2)-2*thetabars(i+1)+thetabars(i))/(dzbar)^2;
end
for i=2:n-1
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon +.....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y1bar(i+1)-2*Y1bar(i)+.....
        Y1bar(i-1))/(dzbar)^2;

    dY2bardtbar(i) = convecy2bar(i) - (n2eqmbar(i)-n2bar(i))/epsilon +.....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y2bar(i+1)-2*Y2bar(i)+.....
        Y2bar(i-1))/(dzbar)^2;

    dThetabargdtbar(i) = convecthetabarg(i) + (ha(i)*L/(epsilon*Cg*........
        Qfeed))*(thetabars(i) - thetabarg(i)) + (epsilonbar*Dg(i)*tfeed....
        /(epsilon*L^2))*(Thetabarg(i+1)-2*Thetabarg(i)+Thetabarg(i-1)).....
        /(dzbar)^2;

    dThetabarsdtbar(i) = (Qfeed*tfeed/(rhob*Cs*L*(Tmax-Tmin)))*(q1*........
        (n1eqmbar(i)-n1bar(i)) + q2*(n2eqmbar(i)-n2bar(i))) - (ha(i)*......
        tfeed/(rhob*Cs))*(thetabars(i) - thetabarg(i)) + (Ds*tfeed/L^2)....
        *(thetabars(i+1)-2*thetabars(i)+thetabars(i-1))/(dzbar)^2;
end
for i=n
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon +.....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y1bar(n)-2*Y1bar(n-1)+.....
        Y1bar(n-2))/(dzbar)^2;
    dY2bardtbar(i) = convecy2bar(i) - (n2eqmbar(i)-n2bar(i))/epsilon +.....
        (epsilonbar*DL(i)*tfeed/(epsilon*L^2))*(Y2bar(n)-2*Y2bar(n-1)+.....
        Y2bar(n-2))/(dzbar)^2;

    dThetabargdtbar(i) = convecthetabarg(i) + (ha(i)*L/(epsilon*Cg*........
        Qfeed))*(thetabars(i) - thetabarg(i)) + (epsilonbar*Dg(i)*tfeed/...
        (epsilon*L^2))*(Thetabarg(n)-2*Thetabarg(n-1)+Thetabarg(n-2))/.....
        (dzbar)^2;

    dThetabarsdtbar(i) = (Qfeed*tfeed/(rhob*Cs*L*(Tmax-Tmin)))*(q1*........
        (n1eqmbar(i)-n1bar(i)) + q2*(n2eqmbar(i)-n2bar(i))) - (ha(i)*......
        tfeed/(rhob*Cs))*(thetabars(i) - thetabarg(i)) + (Ds*tfeed/L^2)....
        *(thetabars(n)-2*thetabars(n-1)+thetabars(n-2))/(dzbar)^2;
end
%..........................................................................
if MassAxial == 882 % Turn on mass axial dispersion
if adsorptionORpurge == 111
    % 2nd BC for d2YdZ2 at zbar = 1 : dY/dz = 0 (wrong); just eliminate
    % d2YdZ2 from the complete equation (Schiesser)
    for i=n
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon + 0;
    dY2bardtbar(i) = convecy2bar(i) - (n2eqmbar(i)-n2bar(i))/epsilon + 0;
    end
end
if adsorptionORpurge == 222
    % 2nd BC for d2YdZ2 at zbar = 1 : dY/dz = 0 (wrong); just eliminate
    % d2YdZ2 from the complete equation (Schiesser)
    for i=1
    dY1bardtbar(i) = convecy1bar(i) - (n1eqmbar(i)-n1bar(i))/epsilon + 0;
    dY2bardtbar(i) = convecy2bar(i) - (n2eqmbar(i)-n2bar(i))/epsilon + 0;
    end
end
end
%..........................................................................
% Wakao 1979, for packed bed, need to use gas thermal axial dispersion
if HeatAxial == 992
if adsorptionORpurge == 111
    % 2nd BC for d2YdZ2 at zbar = 1 : dY/dz = 0 (wrong); just eliminate
    % d2YdZ2 from the complete equation (Schiesser)
    for i=n
    dThetabargdtbar(i) = convecthetabarg(i) + (ha(i)*L/(epsilon*Cg*........
        Qfeed))*(thetabars(i) - thetabarg(i)) + 0;
    end
end
if adsorptionORpurge == 222
    % 2nd BC for d2YdZ2 at zbar = 1 : dY/dz = 0 (wrong); just eliminate
    % d2YdZ2 from the complete equation (Schiesser)
    for i=1
    dThetabargdtbar(i) = convecthetabarg(i) + (ha(i)*L/(epsilon*Cg*........
        Qfeed))*(thetabars(i) - thetabarg(i)) + 0;
    end
end
end
%..........................................................................
for i=1:n
      dn1bardtbar(i) = k1*tfeed*(n1eqmbar(i)-n1bar(i));
      dn2bardtbar(i) = k2*tfeed*(n2eqmbar(i)-n2bar(i));
end
%..........................................................................
```

```
% Six vectors into one vector
%..........................................................................
for i = 1:n
    Yt(0*n+i) = dY1bardtbar(i);
    Yt(1*n+i) = dn1bardtbar(i);
    Yt(2*n+i) = dn2bardtbar(i);
    Yt(3*n+i) = dThetabargdtbar(i);
    Yt(4*n+i) = dThetabarsdtbar(i);
    Yt(5*n+i) = dY2bardtbar(i);
end

if adsorptionORpurge == 111
    i = 1;
    Yt(0*n+i) = 0;     % because Y1bar(1) = y1feed*rhog(1)
    Yt(3*n+i) = 0;     % because Thetabarg(1) = Tfeed*rhog(1)
    %Yt(4*n+i) = 0;    % because Thetabars(1) = Tfeed not
    Yt(5*n+i) = 0;     % because Y2bar(1) = y2feed*rhog(1)
end
if adsorptionORpurge == 222
    i = n;
    Yt(0*n+i) = 0;     % because Y1bar(n) = y1feed*rhog(n)
    Yt(3*n+i) = 0;     % because Thetabarg(n) = Tfeed*rhog(n)
    %Yt(4*n+i) = 0;    % because Thetabars(n) = Tfeed not
    Yt(5*n+i) = 0;     % because Y2bar(n) = y2feed*rhog(n)
end
%..........................................................................
  yt=Yt';

% Increment calls to psa_1
  ncall=ncall+1;
%..........................................................................
```

Biography

Siew Wah Chai was born in Kuala Lumpur, Malaysia on 23rd of March 1978. Her father, Chai Kin Cheon, now retired, was a salesman. Her mother, Wong Lye Yong, is a homemaker. Having three sisters and one brother, she is the 4th among five. Her close family includes aunt Mae Yip, brother and sister in law, nieces and nephew.

Siew Wah received all early educations in her hometown, Kepong. She earned B.Eng. and M.Eng. degrees in Chemical Engineering from, respectively, Universiti Teknologi Petronas (May 2002), and Lehigh University (December 2004). She obtains Ph.D. degree in Chemical Engineering from Lehigh University (September 2011), under the supervision and guidance of Prof. Mayuresh V. Kothare and Dr. Shivaji Sircar. The main publications during her doctoral pursuit are (1) Miniature Oxygen Concentrators and Methods. U.S. Patent Application 2010/0300285, (2) Rapid Pressure Swing Adsorption for Reduction of Bed Size Factor of A Medical Oxygen Concentrator, *Ind. Eng. Chem. Res.*, 2011, (3) Numerical Study of Nitrogen Desorption by Rapid Oxygen Purge for a Medical Oxygen Concentrator, *Adsorption*, 2011. Her research interests are adsorption, process modeling, simulation and control. She was a lecturer in Universiti Teknologi Petronas. Upon the completion of her Ph.D. program, she will join Praxair Inc. at Tonawanda, N.Y., as a development specialist.

Siew Wah's favorite activities are jungle trekking, jogging, sightseeing of natural views, playing guitar and singing, cooking and eating, making colorful slides, and watching inspiring cartoons. Above all, she loves reading GOD's Word, because this is how her old and ragged life is to be transformed completely.