ABSTRACT

Title of dissertation: GLOBAL NONLINEAR MODELING USING AUTOMATED LOCAL MODEL NETWORKS IN REAL TIME

Rose Weinstein
Doctor of Philosophy, 2021

Dissertation directed by: Professor James E. Hubbard Jr.
Department of Aerospace Engineering

Global nonlinear modeling is a challenging task that spans multiple disciplines. When it is necessary to develop a model across the global input space, and a single linear model is insufficient, nonlinear modeling methods are required. If the model is constrained to be developed autonomously in real time, the modeling problem is more difficult, and there are fewer available resources, tools, and techniques for efficient and effective model development. This scenario specifically arises in the context of the NASA Learn-to-Fly concept, which aims to develop tools for real-time aerodynamic modeling and control for new or modified flight vehicles, and which serves as the motivation for this research. This work aims to develop a modeling method that enables the model to be developed automatically in real time, with limited prior knowledge required, and that provides a model that is easily interpretable, allows physical insight into the system, and offers good global and local prediction capabilities. A novel method is developed and presented in this work for automated real-time global nonlinear modeling using local model networks, known

as Smoothed Partitioning with LocalIzed Trees in Real time (SPLITR). The global nonlinear system behavior is partitioned into several local regions known as cells, with the dimension, location, and timing of each partition automatically selected based on a new residual characterization procedure, under the constraints of real-time operation. Regression trees represent the successive partitioning of the global input space and describe the evolution of the cell structure. Recursive equation-error least-squares parameter estimation in the time domain is used to estimate a model that represents the local system behavior in each region so that the model can be updated independently with data in the explanatory variable ranges of each cell, even if the data are not contiguous in time. A weighted superposition of these piecewise local models across the input space forms a global nonlinear model that also accurately captures the local behavior. The SPLITR approach was tested and validated using both simplified simulated test data, as well as experimental flight test data, and the results were analyzed in terms of model predictive capabilities and interpretability. The results show that SPLITR can be used to automatically partition complex nonlinear behavior in real time, produce an accurate model, and provide valuable physical insight into the local and global system behavior.

GLOBAL NONLINEAR MODELING USING AUTOMATED LOCAL MODEL NETWORKS IN REAL TIME

by

Rose Weinstein

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2021

Advisory Committee:
Professor James E. Hubbard Jr., Chair/Advisor
Professor Amr Baz
Professor Robert M. Sanner
Professor Allen Winkelmann
Professor Huan Xu

Preface

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE 1840340, and by the NASA Aeronautics Research Mission Directorate (ARMD) Transformational Tools and Technologies (TTT) project.

Dedication

To my parents, family, friends, and all the mentors and role models whose dedication to me is what brought me here.

In memory of Eugene H. D. Heim,

an exemplary engineer, mentor, and colleague who inspired many with his optimism, energy, determination, and success.

Acknowledgments

"*First, think. Second, dream. Third, believe. And finally, dare.*" — Walt Disney

I joined the University of Maryland's Morpheus Lab in my sophomore year, and emerge 8 years later a product of the lab of "dreams," academically, professionally, and personally. Dr. Hubbard's lab is a unique cultivating environment in which we are given the utmost support, respect, and flexibility, and where we are trained to pursue technical excellence and rigor. We are taught that success is not an entitlement, and that hard work, determination, and of course, dreams, are key.

I find the Morpheus Lab experience to align well with Walt Disney's quote. First, we are equipped with the skills and technical expertise to think independently, ambitiously, and rigorously. Second, we are encouraged and required to funnel our thoughts into dreams that may seem "hopelessly naive" to others, but that feel, and are, within reach to us. Third, our foundation, collective support, and filtering of negativity enable us to truly believe in the power of our dreams. And finally, we are given the confidence, resoluteness, and determination to dare to pursue and fulfill those dreams. Together, and only as a team, the Wolfpack endures all, and enables and sets those dreams in motion.

Dr. Hubbard — I cannot thank you enough for the critical role you played in not only turning my dreams into reality, but enabling and equipping me to dream in the first place. From the moment I woke you up during that first Skype meeting in sophomore year, you have believed in me and supported me unconditionally.

Your students never doubt that they are your constant priority, and that you care deeply for our wellbeing, academically, professionally, and personally. You are always grounded in reality and in what is important. You encouraged and trained us to focus on our goals and our pursuit of research, while ignoring the noise, the negativity, and the naysayers. You were an incredible role model, mentor, and advisor throughout this journey. Dr. Hubbard — "you were right."

I would not be where I am today without the support of Morpheus Lab Gen III members Dr. Michael Cunningham, Dr. Jose Mondragon, Dr. Joaquim Dias, and Nick Rymer. You all played a critical role in the development of my research, and of myself as a researcher and an individual. Michael — through our countless technical discussions and meetings, both in person and virtual, you were always available to talk through new ideas, you provided insightful feedback, and you helped me see the clarity through the fuzziness. Jose — I met you early in my undergraduate years, you have been a constant and reliable friend throughout this entire journey, and you were always available to offer needed support and guidance.

My graduate school experience was embedded at NASA Langley Research Center in the Pathways Program, and I would like to thank the Flight Dynamics Branch (FDB) for allowing me to pursue my graduate degrees while part of such a great group of role models and colleagues. In particular, I worked closely with team members in the FDB and the Dynamic Systems and Control Branch (DSCB) across the Learn-to-Fly (L2F) and Modeling and Control for Agile Aircraft Development (MCAAD) projects. I would like to acknowledge and thank my team members for all of your support and technical discussions, and for the efforts conducting flight

test operations and collecting flight data used in this research.

I would also like to thank my NASA Pathways mentors whom I worked closely with throughout my internship. Mark Croom — you encouraged me to think outside of the traditional box, and to always look toward the stars. Gene Heim — your "can do" attitude, optimism, and energy were inspiring and infectious, and you applied exemplary rigor to every problem we came across. Dr. Bruce Owens — you brought me over the finish line with your encouragement and support, despite my infringing upon your exclusive FDB PhD rights.

Dr. Gene Morelli — although I never had the privilege of taking your Aircraft System Identification course, I learned so much from you directly during our hours of technical discussions. Your door was always open to chat, and your expertise and insight into the field of aircraft system identification are unrivaled. Your commitment and dedication to research are truly admirable, and your wide-ranging ideas and suggestions always gave me food for thought. Gene — I very much appreciate your time, patience, recommendations, consultation, guidance, and insight throughout my dissertation research.

Dr. Jared Grauer — you have been an integral part of my graduate school experience, both at Morpheus Lab as well as at NASA. You have offered unparalleled patience, time, expertise, support, guidance, mentorship, and friendship throughout my research and beyond. You helped encourage me through the setbacks and joined in celebrating the successes. You offered expertise and provided direction, while encouraging individuality. Your accessibility and dedication went above and beyond, and I truly appreciate your role and involvement throughout this work.

Dr. Alison Flatau — you started off as my undergraduate academic advisor, and went on to offer me my first research experience in Aerosmart lab after my freshman year. You truly believed in me from the very beginning, you sent me to my first conference to experience where research comes together, and you were always available to offer advice.

Dr. Zohaib Hasnain — I began working with you after my freshman year in Aerosmart Lab through your own PhD work, and that is where I got my first research experience. You believed in me and in my potential before I had even taken my introductory aerospace courses, and you provided me with the inspiration and confidence to forge my own path forward. From the perspective that one domino can trigger the whole journey, it all started working with you.

Last but not least, I would like to thank my family for their unconditional support throughout this entire neverending pursuit of education. My father, Elliott Weinstein, showed me a determination to take the difficult path, a deep respect for math and science, and the importance of independent and resolute thinking. My mother, Gitta Weinstein, provided the much-needed support, nurturing, and reassurance to pursue the goals I set forth to accomplish, and made sure that I balanced work with other important goals. My siblings, Yossi, Ari, and Dassy, my sister-in-law, Deena, and my nieces and nephews offered levity and encouragement, and helped me to balance academia with other priorities in life.

Table of Contents

List of Tables

List of Figures

Nomenclature

Roman Symbols

a_x, a_y, a_z	body-axis translational acceleration
b	wing span, ft
C_X, C_Y, C_Z	body-axis nondimensional force coefficients
C_l, C_m, C_n	body-axis nondimensional moment coefficients
C_L, C_D	wind-axis nondimensional force coefficients
$\bar{c}$	wing mean aerodynamic chord, ft
1D	one-dimensional
2D	two-dimensional
$\boldsymbol{D}$	dispersion matrix
d	propeller diameter, ft
I_x, I_y, I_z, I_{xz}	inertia tensor elements, slug-ft^2
J	advance ratio
m	mass, slug
N	number of data points
p, q, r	body-axis roll, pitch, and yaw rates, rad/s
$\bar{q}$	dynamic pressure, lbf/ft^2
$\mathbb{R}$	real number
R^2	coefficient of determination
S	wing reference area, ft^2
T	thrust, lbf
u	model input variable
V	true airspeed, ft/s
w	validity function
x	model explanatory variable
y	model output
z	response variable

Greek Symbols

α	angle of attack, rad or deg
δ_e	elevator deflection, rad
θ	parameter estimate
λ_ν	residual threshold factor
λ_σ	standard deviation factor
$\lambda_{\sigma,norm}$	number of standard deviations to normalize
λ_{ff}	forgetting factor

λ_s	smoothness factor
μ	mean
ν	residual
Σ	covariance matrix
σ	standard deviation
ϕ	partitioning variable
Ω	propeller speed, rad/s

Superscripts

$^{-1}$	matrix inverse
$\bar{}$	mean
$\hat{}$	estimate
$\dot{}$	time derivative
^{T}	transpose

Acronyms

AAM	Advanced Air Mobility
CART	Classification And Regression Trees
CAS	Convergent Aeronautics Solutions
CFD	computational fluid dynamics
EV	explanatory variable
EM	expectation-maximization
L2F	Learn-to-Fly
LMN	local model network
LOLIMOT	LOcal LInear MOdel Trees
MCAAD	Modeling and Control for Agile Aircraft Development
MOF	multivariate orthogonal functions
MSE	mean squared error
PTI	programmed test input
PV	partitioning variable
RMS	root mean square
RSS	residual sum of squares
SISO	single input single output
SNR	signal to noise ratio
SPLITR	Smoothed Partitioning with LocalIzed Trees in Real time
UAM	Urban Air Mobility

Chapter 1

Introduction

1.1 Motivation: a new era of air vehicles

In the past, the design objectives for aircraft have largely been driven by military, commercial transport, and business transport utilization, with relatively few large companies rising up to dominate the market, and development programs that included up to billion dollar budgets, extensive workforces, and prolonged durations. More recently, advancements in autonomous technologies, electric propulsion, and computing capabilities have broadened the applications and advantages of taking to the skies to undertake tasks previously done on the ground [1].

A proliferation of Unmanned Aerial Systems (UAS) of varied shapes and sizes has enabled multiple sectors to harness the versatility for specialized applications. The military can utilize unmanned vehicles for intelligence, surveillance, and combat, and commercial applications include crops monitoring, power lines inspection, search and rescue, and package delivery. Perhaps the most poignant and far-reaching

potential lies in transportation. Calls for Mobility On Demand (MOD) leverage these technologies to transform existing transportation systems and provide travelers with flexible, customized, and efficient transit options [2]. Advanced Air Mobility (AAM) envisions on-demand, accessible aviation services that can transport people and cargo across rural, suburban, and urban environments, with Regional Air Mobility (RAM) expediting inter-city transit and Urban Air Mobility (UAM) easing congestion with solutions that specifically address the challenges of air vehicle operation within densely populated urban environments [1].

The objectives and associated demands of AAM have blurred the traditional dichotomy of air vehicle categorization as conventional fixed-wing or helicopter, as unconventional concepts are emerging to address the challenges associated with wide ranges of travel distance, capacity, environments, and capabilities. From small-scale package delivery drones, to Personal Air Vehicles (PAVs), air taxis, and regional transports, new configurations span multicopters, tiltrotors, vertical take off and landing (VTOL), and lift + cruise, among others [3, 4].

This revolutionary new frontier of air vehicle utilization, and the urgency associated with ambitious demands for imminent implementation, have created a competitive market that is occupied by both established large aircraft companies as well as emerging startups [1, 4]. Electric VTOL (eVTOL) concepts have proliferated into hundreds of designs competing to fill the void. The success of an individual design is based to a large degree on minimizing budgets, reducing the necessary workforce, and expediting the program development. This all must be done while maintaining strict standards of vehicle safety, reliability, and certification, and leveraging

new technologies to provide autonomy, intelligent systems, and advanced onboard capabilities.

With reduced budgets and manpower, expedited timelines, and novel vehicle configurations that have complex dynamics and aerodynamics, there is urgent demand for a new collection of tools to efficiently, safely, and expeditiously develop these air vehicles. The next section will describe the traditional aircraft development process and the reasons why it may not meet these requirements, as well as an alternative approach offered by the NASA Learn-to-Fly concept. This new approach also offers valuable improvements to developing even more conventional vehicles. The rest of the chapter will cover background material and a literature review of aircraft system identification, which is at the core of aircraft development.

1.1.1 Traditional aircraft modeling and control development process

For a given aircraft design, the conventional paradigm for the development of aircraft flight systems is a lengthy, iterative process that involves estimating an aerodynamic model, building a simulation model, and designing a control law and guidance algorithm, where each step involves extensive testing and analysis, as outlined in Fig. 1.1 [5, 6]. A core element of the aircraft flight systems that the other components rely on is the aerodynamic model, which can provide physical insight into the aircraft stability and control characteristics, inform a simulation model to predict the aircraft response, and impact model-based control law designs

and guidance systems. Given the central role of the aerodynamic model, the model fidelity is an important factor to ensuring the safety and reliability of the aircraft operations.

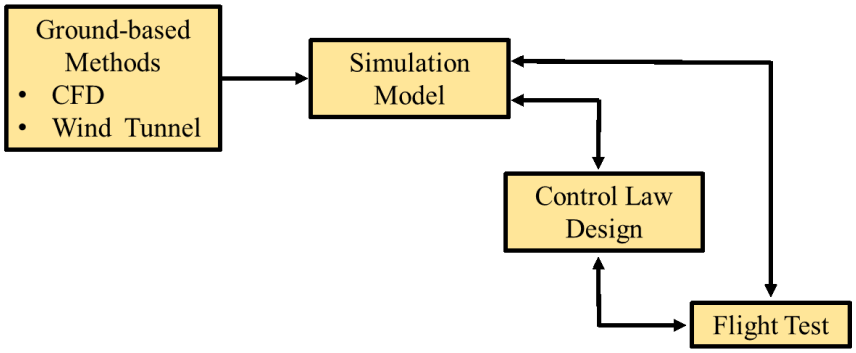


Figure 1.1: Conventional aircraft modeling and control development process (adapted from Ref. [7]).

The traditional approach to estimating an aerodynamic model may involve both numerical and experimental test techniques. Vortex lattice methods (VLM) based on first principles are easily accessible in software packages such as MIT's Athena Vortex Lattice (AVL) and the United States Air Force's Digital DATCOM, and can provide preliminary stability and control estimates rapidly [8]. Advancements in numerical computing have allowed computational fluid dynamics (CFD) to become a valuable and flexible tool to simulate flow conditions. However, fidelity limitations exist from grid geometry and flow condition approximations, test cases take a long time to run, and these methods have difficulty capturing complex flow fields, particularly at low Reynolds numbers where UAM vehicles predominantly operate [9]. Both static and dynamic captive wind tunnel tests are often used to develop an aerodynamic database using a scaled aircraft model. However, test entries

can be expensive and time consuming, and results can be subject to errors from dynamic scaling, wall and sting interferences, and incorrect Reynolds number and Mach number matching [10]. Each of these test techniques is an effort to emulate the aerodynamics that the full-scale aircraft would experience in flight, so flight testing provides the most authentic data for aircraft modeling. Nevertheless, flight test campaigns can be costly, lengthy, and risky to execute, and data quality can be degraded by practical aspects such as measurement noise, insufficient dynamic excitation, and unmodeled disturbances.

All of these modeling methods traditionally require extensive time, cost, planning, analysis, and iteration, and while improvements continue to be made in software and hardware capabilities, there remain to be discrepancies between numerical, wind tunnel, and flight test results. Furthermore, this entire process may need to be repeated for a vehicle configuration change. Often when a new vehicle is being developed, a similar past vehicle is found and used as a starting point to provide a baseline aerodynamic model that can offer initial physical insight. However, in the fast-moving innovative air vehicle environment inspired by UAM, the hybrid design space that these vehicles fill is unprecedented, so the vehicles lack much similarity to one another, and particularly to past well-established vehicles. For these reasons, and the others discussed in this section, the traditional aircraft modeling and control development process may not be sufficient to meet the demands of UAM vehicle development, and this realization additionally illuminates the inherent inefficiencies involved with using this process even for more conventional vehicles.

1.1.2 Learn-to-Fly aircraft development concept

The NASA Learn-to-Fly (L2F) concept is an advanced technology development initiative that aims to improve the traditional aircraft modeling and control development process with a new paradigm that offers efficient, automated, and intelligent methods that minimize the time, effort, and analysis involved. The approach is to replace most of the ground-based testing and analysis with automated, onboard tools that provide in-flight real-time aerodynamic modeling, learning control, and guidance, with little reliance on prior knowledge of the particular aircraft aerodynamics, as depicted in Fig. 1.2. The motivating analogy is that just as a young bird can jump out of its nest and "learn to fly" on its own by "figuring it out," an aircraft can be equipped with a robust set of tools that will provide it with the self-awareness and learning capabilities to learn how to fly and to guide itself safely.

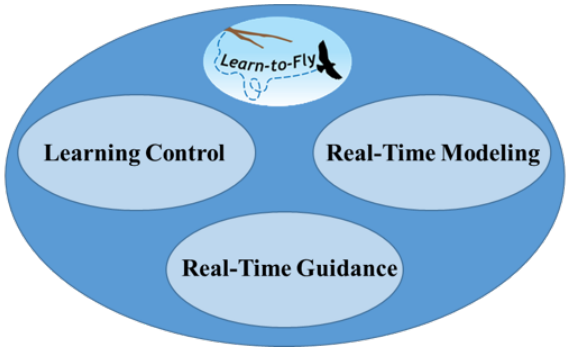


Figure 1.2: Learn-to-Fly aircraft development process (adapted from Ref. [7]).

L2F aims to develop a toolbox of advanced technologies to facilitate rapid development of new or modified vehicles by improving the efficiency of flight testing with automated onboard capabilities to enable safe autonomous envelope expan-

sion and learning control. Since traditional understanding of conventional aircraft aerodynamic properties cannot be fully relied upon for these unprecedented configurations, the tools must be adaptive, customizable, and informative as these systems are explored. The objective is to allow each vehicle to "be all it can be," the extent of which is dependent on the stability and control characteristics, and control power and bandwidth.

This enabling technology can actually be used across the full lifecycle of the aircraft. So far, the focus has been on expediting the aircraft modeling and control development process where the goal is to learn about the aircraft, to build the nominal aerodynamic model, and to enable safe envelope expansion. These tools can also be useful in the future for certification purposes by expediting and streamlining the required flight testing component with efficient test techniques, and also supplying rich, informative flight data for analysis. Finally, during flight operations L2F algorithms can be running in the background to provide real-time self-awareness for autonomous systems, with applications in health monitoring, fault detection, and self-healing capabilities that can provide a valuable layer of safety and reliability.

L2F efforts have spanned numerous projects based at NASA Langley Research Center (LaRC), with a strong emphasis on experimental flight testing. Early work began with developing tools for global nonlinear aerodynamic modeling, with later work building on these capabilities with model-based control and guidance methods [5, 6]. Several recent projects will be summarized below, but are not intended as a comprehensive overview.

Under a NASA Aeronautics Research Mission Directorate (ARMD) Seedling

project from 2013–2014, several novel piloted excitation inputs were tested on the MB-326M Impala at Mojave, CA, and a global nonlinear aerodynamic model was updated onboard using batch data from several maneuvers at varying flight conditions [11, 12].

In 2015, the Bat-4 was flown at NASA Wallops Flight Facility using the NASA Langley AirSTAR flight testing capability with a Mobile Operations System (MOS) ground station. With the aircraft remotely piloted from the MOS, this research demonstrated automated multisine excitation inputs injected onto the control surfaces, along with real-time nonlinear aerodynamic model development that was computed offboard using ground-based systems and real-time telemetry [7].

The Convergent Aeronautics Solutions (CAS) project from 2016–2017 took an aggressive approach to evaluate the feasibility of the broader L2F vision: to merge real-time aerodynamic modeling, learning control, and automated guidance using standard onboard computing hardware [5]. Flight tests using a MiG-27 Foam Target Drone at the NASA City Environment Range Testing for Autonomous Integrated Navigation (CERTAIN) range, and using a novel joined-wing aircraft known as Woodstock at Fort AP Hill, were used to simulate a bird jumping out of its nest by releasing these unpowered gliders from a tethered balloon [5, 13]. In particular, the Woodstock vehicle was purposely designed to be unconventional with twelve control surfaces and an all-moving empennage for pitch and directional control, to test the L2F algorithms to their limits. A more conventional test aircraft known as E1, and pictured in Fig. 1.3, provided the flexibility of allowing a ground-based pilot to take off and land, with additional flight modes that enabled automated control inputs

injected during piloted envelope expansion, and fully autonomous flight to test the control law and guidance algorithms [5, 13]. Results of these flight tests showed a successful onboard synthesis of automated excitation inputs, real-time global nonlinear aerodynamic modeling, model-based control using adaptive nonlinear dynamic inversion as well as a classic autopilot control law, and an automated guidance system that specified desired flight conditions and enabled envelope expansion under the constraints of the testing range [14-16].

L2F tools are continuing to be developed under Modeling and Control of Agile Aircraft Development (MCAAD), part of the NASA Transformative Aeronautics Concepts Program's (TACP) Transformational Tools and Technologies (TTT) project. MCAAD research continues to advance the L2F onboard modeling and control capabilities with improvements such as in-flight moment of inertia estimates, autotuning control laws, and additional modeling approaches that are frequently flight tested on the E1 platform [6, 17-20]. Ongoing and future L2F work looks at applying these tools to other novel UAM-inspired vehicles.

As discussed throughout this section, at the core of the aircraft dynamics and control development process lies the reliance on an aircraft model, and this research focuses on building an aerodynamic model from flight data. The aerodynamic model provides essential physical insight into the vehicle's response characteristics, and can inform safe operation within the flight envelope. Additionally, the aerodynamic model is central to the other aircraft development steps, including simulation models, control laws, and guidance systems.

1.2 Background on aircraft system identification

The goal of aircraft system identification is to develop a mathematical description of the aircraft that predicts the aerodynamic responses due to control inputs and flight variables. If the vehicle can be approximated as a rigid body with constant mass properties, aircraft flight dynamics are derived from Newton's second law expressed in the body frame as [21]

$$\boldsymbol{F} = \boldsymbol{F}_A + \boldsymbol{F}_T + \boldsymbol{F}_G = m\dot{\boldsymbol{V}} + \boldsymbol{\omega} \times m\boldsymbol{V} \tag{1.1}$$

$$\boldsymbol{M} = \boldsymbol{M}_A + \boldsymbol{M}_T = \boldsymbol{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \boldsymbol{I}\boldsymbol{\omega} \tag{1.2}$$

where $\boldsymbol{F}$ is the total applied force, $\boldsymbol{F}_A, \boldsymbol{F}_T$, and $\boldsymbol{F}_G$ are the applied aerodynamic, thrust, and gravity forces, respectively, m is the vehicle mass, $\boldsymbol{V}$ is the translational velocity vector, $\boldsymbol{\omega}$ is the rotational rate vector, $\boldsymbol{I}$ is the inertia matrix, $\boldsymbol{M}$ is the total applied moment, and $\boldsymbol{M}_A$ and $\boldsymbol{M}_T$ are the applied aerodynamic and thrust moments, respectively. With a propulsion model typically obtained from ground tests, the identification challenge lies in estimating a model for the six aerodynamic forces and moments in $\boldsymbol{F}_A$ and $\boldsymbol{M}_A$, shown in Fig. 1.3, as a function of the measured states and controls. The forces and moments are often nondimensionalized to remove the dependency on mass, geometry, and dynamic pressure, and expressed as

$$\boldsymbol{F}_A = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \bar{q}S \begin{bmatrix} C_X \\ C_Y \\ C_Z \end{bmatrix}, \qquad \boldsymbol{M}_A = \begin{bmatrix} L \\ M \\ N \end{bmatrix} = \bar{q}S \begin{bmatrix} bC_l \\ \bar{c}C_m \\ bC_n \end{bmatrix} \tag{1.3}$$

where $\bar{q}$ is dynamic pressure, S is the wing reference area, b is the wingspan, and $\bar{c}$ is the mean aerodynamic chord. By assuming small perturbations, the aerodynamic force and moment coefficients can also be linearized about a reference flight condition and expressed as a Taylor series expansion, and the parameters to be estimated are the stability and control derivatives. For example, an approximation of C_m can be expressed as

$$C_m = C_{m_0} + C_{m_\alpha}\Delta\alpha + C_{m_q}\frac{q\bar{c}}{2V_0} + C_{m_{\delta_e}}\Delta\delta_e \tag{1.4}$$

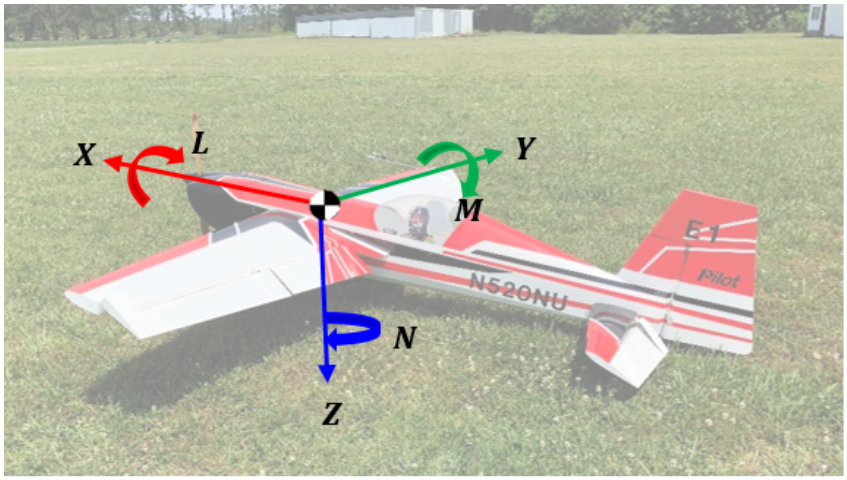


Figure 1.3: Aircraft body-axis forces and moments on the E1 test aircraft (credit: NASA Langley Research Center).

1.2.1 Modeling considerations

The remainder of Section 1.2 will discuss a variety of considerations and approaches for estimating an aerodynamic model from flight data. It is worth mentioning the well-known expression by statistician George E. P. Box that, "all models are wrong, but some are useful." A perfect model does not exist, and the goal is not to create one, but rather to find a sufficient representation that serves a specified purpose based on a set of reasonable assumptions. Prediction error can exist due to a variety of reasons, including noisy data, model structure errors, and parameter uncertainties. Addressing these points in advance will clarify the appropriate modeling approach from a wide variety of options. Any available *a priori* knowledge about the system to be identified should be used to inform the identification process, to ensure valid assumptions are made, and to make sure only the unknown components are estimated.

The purpose of the model may be to understand the open- or closed-loop dynamic and stability characteristics, evaluate aircraft performance or handing qualities, inform an aircraft simulation, design a control system, or any combination of these options. Common assumptions include considering the aircraft as a rigid body, and decoupling the longitudinal and lateral-directional dynamics. However, these simplifications need to be evaluated if, for example, the aircraft displays aeroelastic properties or exhibits strongly coupled motion.

Some of the common modeling considerations that can lead to simplifying the problem and guiding the modeling process are summarized as follows [22]:

White box vs. black box vs. gray box: White, black, and gray box models are characterized by the amount of prior knowledge that is used in the modeling process. A white box model can be derived entirely from first principles. A black box model assumes no prior knowledge and is based entirely on experimental data. Gray box models, which cover most practical identification problems, use any available physical insight to inform a relevant model structure, while the remaining unknown parameters are estimated through experimental data. "Black box" is also used to refer to a model that matches the input-output properties of the system, but which lacks interpretability and physical meaning, both in the structure as well as the estimated parameters.

Linear vs. nonlinear: If a linear approximation of the aircraft dynamics is sufficient for a given application, then the modeling problem is drastically simplified and the extensive body of linear systems theory can be used for identification and analysis. However, for more complex nonlinear dynamics involving full-envelope modeling or large amplitude maneuvers, a nonlinear model must be used.

Local vs. global: A local model can represent the system at a specified operating point, while a global model may be valid over the entire operational envelope. Often, local models tend to be linear while global models must take into account multiple operating points, as well as nonlinear regimes.

Parametric vs. nonparametric: Parametric models such as transfer functions, state space models, and polynomials are often used to characterize the aircraft dynamics, but *a priori* assumptions are usually required to pose a physically meaningful model structure and/or initial parameter estimates. Nonparametric models such as impulse responses can also provide insight into linear system dynamics.

Online vs. offline: If the model can be developed offline after all the data have been obtained, then the data can be processed in batch, and iterative, nonlinear optimization methods can be used. If online modeling is required, then practical estimation methods are those that are compatible with recursive updates and require reasonable computational power that can be provided by onboard flight computers.

Closed loop vs. open loop: Closed-loop identification is used to evaluate the dynamics of a system with feedback for applications such as handling qualities, certification, or control law evaluation, while open-loop identification can be used to estimate the natural dynamics of the system for physical insight and control law design.

Stable vs. unstable: Time-domain parameter estimation methods that include integration may have numerical divergence issues, so system stability can impact the system identification process. Additionally, unstable systems will require a control law to fly the aircraft and to gather flight data.

Stochastic vs. deterministic: If the system is deterministic, then the state estimation

process is not needed, and a wider range of methods can be used. For stochastic systems, statistical methods such as Bayesian approaches or Kalman filters must be used to account for random variables.

Rigid-body vs. aeroelastic: If the aircraft can be sufficiently approximated as a rigid body, the equations of motion can be expressed as in Eqs. (1.1 –1.2) and the system identification problem is simplified. If the aeroelastic properties are significant and coupled with the rigid-body dynamics, then the model form must be adapted to include the structural dynamics, and the aircraft must be outfitted with additional sensors to measure the relevant structural responses.

Steady vs. unsteady: If the aerodynamics are approximated as steady or quasi-steady, then a model structure that relies on instantaneous states or state estimates can be used. If unsteady aerodynamics are evident, then time-dependent terms such as lag states can be included.

Throughout the decision making used to inform a particular modeling approach, considerations should be made to ensure certain model properties that fulfill the purpose of the model. These characteristics are not binary standards, but rather guiding criteria to evaluate model quality. Ideally, the model should display all of these properties, but practically it usually exhibits a compromise among them. Common desirable model properties include [21-23]:

Consistency: If an increase in the amount of modeling data will result in the model approximation approaching a representation of the true physical system, the model is consistent.

Parsimony: The law of parsimony is a guiding principle to selecting a model that provides the prescribed level of prediction with the simplest model structure and the fewest possible parameters.

Interpretability: An interpretable model is one that is not viewed as a black box; rather, it is transparent such that the model structure and parameters provide physical insight into the system.

Predictive capability: A model that has good prediction capabilities captures the deterministic modeling information but does not fit the noise content, and can be used to predict data that were not used in the modeling process.

Robustness: A robust model is generalizable and provides results with good prediction despite noisy data.

Modularity: If the model is effectively compartmentalized such that poor data quality or high uncertainty in one part of the modeling domain do not negatively impact the model integrity in other regions, the model can be considered modular.

Model accuracy: Quantification of the uncertainty characteristics of the model output and estimated parameters is an indication of the reliability and accuracy of the model, and is a critical part of the modeling process.

1.2.2 A review of system identification tools, techniques, and processes

The system identification process, shown in Fig. 1.4 and discussed at length in Refs. [21, 24, 25], involves a complex series of steps including posing a desired model form, designing an experiment to obtain informative data, developing a model structure and estimating the model parameters, and validating the final model. Each of these steps will be discussed throughout this section within the context of aircraft system identification.

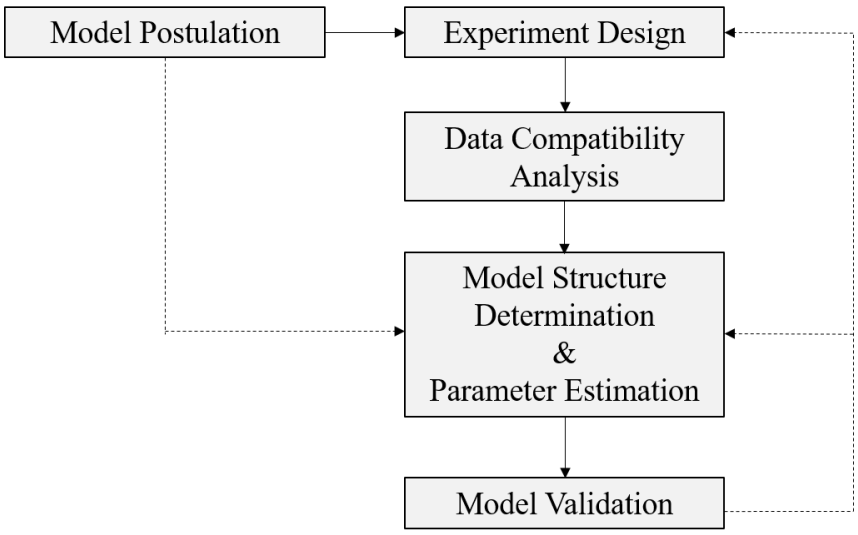


Figure 1.4: System identification process (adapted from Ref. [21]).

Model Postulation: Model postulation entails choosing a class of appropriate model forms through which to characterize the system. *A priori* knowledge about the system, underlying simplifications and assumptions about the dynamics, and specification of the purpose of the model influence the chosen form, which will impact the entire experiment and the remaining identification steps.

Often, the goal of aircraft system identification is to express the six aerodynamic force and moment coefficients as multivariate Taylor series expansions that characterize the functional dependencies of the forces and moments on states and controls. The truncated Taylor series expansion has inherent limitations, but is often a reasonable approximation. The estimated parameters are then stability and control derivatives which offer physical insight into the aerodynamic characteristics. The aerodynamic model can also then be incorporated into the equations of motion derived from first principles to further evaluate and simulate the dynamic response.

The model can also take the form of a transfer function to characterize the linear dynamic response of the aircraft to specified inputs expressed through poles, zeros, and gains. This form is most convenient for single input single output (SISO) cases, but can also be expanded to characterize pairwise inputs and outputs in a multiple input multiple output (MIMO) case. Alternatively, models can be postulated in state space form to directly express the equations of motion. In each of these approaches, the estimated parameters representing stability and control derivatives can be extracted from the model structure.

Typically, a certain amount of *a priori* knowledge of the system is a prerequisite for posing a representative model structure for identification. When this

information is not available or a meaningful model structure is unnecessary, non-parametric models such as impulse responses or frequency responses can be derived from test data and used to provide direct physical insight into the nature of the aircraft dynamic response to inputs, as well as provide valuable information to inform a parametric model structure. Alternatively, black box models such as neural networks focus on characterizing the input-output properties and allow the model structure to be developed automatically using training methods. However, these model forms do not offer physical insight into the system.

Experiment Design: The quality of the estimated model is predicated on the data used in the system identification, so experiment design is an essential step toward ensuring informative data for the modeling process. This step includes outfitting the aircraft with sufficient instrumentation, designing the excitation inputs, and specifying the flight conditions and maneuvers through which to gather data. The experiment is also strongly influenced by the postulated model and underlying assumptions, such as maintaining a linear response, assuring SISO assumptions, or decoupling the longitudinal and lateral-directional dynamics. If the aircraft is unstable or difficult to control in an open-loop setup, then an inner-loop flight control system may be needed to obtain desirable flight characteristics during flight testing. If the aircraft is piloted (onboard or remotely), the pilot must be trained to fly the aircraft through the flight envelope and excite the dynamics. Sometimes the perturbation inputs and/or the guidance are implemented autonomously through an onboard flight computer.

The goal of input design is to excite the aircraft dynamics at a particular flight condition in order to obtain informative flight data that capture the dynamic response of the aircraft. Single-input excitation forms can include impulses, sine waves, square waves, two-sided pulses such as doublets and multisteps, frequency sweeps, and multisines. Each input design is informed by two main considerations: sufficient amplitude to ensure data with good signal to noise ratios (SNRs), and sufficient frequency content to excite the aircraft dynamics at the modal frequencies to be modeled. The amplitude must be chosen to ensure sufficient SNR, subject to the constraints of the approximation, such as using a linear model and ensuring small perturbations about a flight condition. When the bandwidth of modal frequencies to be modeled is known, then the inputs can be tailored to excite a particular set of frequencies. Otherwise, wideband inputs must be used to excite a wide range of frequencies that are expected to include the desired modal frequencies.

When there are multiple simultaneous input channels, the excitation must be applied in a way that minimizes signal correlation so that it is clear which inputs influence which parts of the response variables. This can be done by applying the inputs sequentially in time, or by using orthogonal inputs on each control surface, such as orthogonal square waves or multisines [21].

Past L2F work has shown that Programmed Test Inputs (PTIs) that apply automated orthogonal optimized multisine perturbation inputs on multiple control surfaces simultaneously produce informative multi-axis data in an efficient manner [7, 13, 26-29]. The goal is to excite the aircraft dynamics using wideband inputs that consist of a sum of sinusoids at specified frequencies, phases, and amplitudes.

The PTIs are designed within a bandwidth that includes the expected range of aircraft modal frequencies, and they are phase-optimized for minimum relative peak factor to provide perturbations that do not cause the aircraft to deviate far from the nominal flight condition. The PTIs have been successfully demonstrated on L2F flight tests throughout the CAS and MCAAD projects using a flight computer to inject the excitation inputs on the control surfaces by summing these automated inputs with the pilot or control system commands [7, 14, 18].

Data Compatibility Analysis: Prior to identification, measured sensor data obtained from the flight test experiment must be processed to ensure the data are an accurate representation of the signals. Data compatibility analysis verifies that the data are kinematically consistent by comparing the measured responses with state estimates reconstructed using the known rigid-body kinematics. The signals are corrected by removing systematic errors such as time skews and sensor biases, correcting center of gravity offsets, removing dropouts or spikes, and applying filtering or smoothing techniques to improve the SNR [21, 30].

Model Structure Determination: Model structure determination entails finding a specific structure from a class of postulated models to describe a particular aircraft using the test data. Model parsimony is an important consideration in this step as motivation to choose the simplest model structure with the fewest estimated parameters that offers a sufficient level of prediction accuracy. Regardless of the postulated model form, model structure determination is an important step to cus-

tomize a model form for a particular data set to improve the accuracy of individual parameter estimates and minimize correlation between chosen explanatory variables.

Parameter Estimation: For a given model structure, parameter estimation entails using the measured input-output data to solve for the unknown model parameters (and associated uncertainties) by minimizing a particular cost function. In aircraft system identification, the three most common parameter estimation methods are equation error, output error, and frequency response error [21, 24, 31, 32].

Equation-error parameter estimation uses linear regression techniques based on ordinary least squares to minimize the sum of squared differences between the response variable data and the model output for a model structure that is linear in the parameters [21]. Some advantages of the equation-error method are the simplicity of formulation and straightforward non-iterative solutions, the absence of temporal requirements in the regressors that allows multiple maneuvers to be concatenated, the expansion to stepwise regression methods to assist in model structure determination, and the flexibility of allowing nonlinear modeling terms. Additionally, this method can be formulated recursively, which can enable real-time updates to a model as new information is received. A disadvantage of equation-error parameter estimation is the inherent assumption of deterministic regressors; if there is measurement error in the regressors, the least squares estimates are biased, inconsistent, and inefficient. Nevertheless, high quality sensors and data preprocessing can reduce the measurement errors to provide reliable, useful results.

The output-error parameter estimation method often formulates the model

as the aircraft equations of motion in state-space form, and solves for unknown parameters by minimizing the error between the measured or calculated data and the output equation [21, 33]. The output-error method requires the states to be calculated through integration and uses nonlinear optimization such as the Gauss-Newton method to estimate the parameters that appear in the equations of motion. Advantages of the output-error method are flexibility in the model structure, and that the maximum likelihood parameter estimates are unbiased, consistent, and efficient when there is no deterministic modeling error. Some disadvantages of the output-error method are that initial parameter estimates are required, and iterative nonlinear optimization solution methods are used, which can encounter convergence problems and can be difficult to implement for real-time modeling.

Both the equation-error and output-error parameter estimation methods can be applied for test data expressed in the time domain or the frequency domain. Time-domain estimation is straightforward, as the goal is often to match time history data, and the bias term associated with the flight condition can be extracted. For frequency-domain estimation, the data first need to be transformed into the frequency domain using the Fourier transform, but this approach offers several advantages, such as fewer data points required in the estimation and robustness to noise.

Another SISO approach to parameter estimation involving the frequency domain is to pose a cost function that minimizes the frequency response error between the test data and the model. This usually constitutes a two-step approach: first a nonparametric frequency response is computed, which can provide insight in and of

itself into the system dynamics, as well as inform the postulated model structure. Then a nonlinear optimization is used to estimate the unknown model parameters to match the frequency responses. This approach removes the effects of uncorrelated process and measurement noise and is especially useful for unstable systems, but it is restricted to linear models [30, 34].

Model Validation: Model validation is the final step through which the estimated model is evaluated to determine if it offers sufficient prediction, if the parameter estimates are physically reasonable, and if it possesses the other desirable model properties discussed at the end of Section 1.2.1. Validation criteria can include comparing estimated parameters with other sources of information such as first principles, CFD, or wind tunnel data. To ensure the model is generalizable and not overfit to the test data, model predictive capability is evaluated using representative validation flight data that was not used in the modeling process [21, 30].

Given the variations of approaches for each step, the aircraft system identification process described in this section is generally applicable and widely used. Popular software packages that employ these methods include SIDPAC developed by Morelli at NASA Langley [21], CIFER by Tischler from the US Army Research Laboratory [30], FITLAB by Seher-Weiss at DLR [35], and FVSysID by Jategaonkar at DLR [36].

1.3 A literature review of global aerodynamic modeling from flight data

For many applications, aircraft flight dynamics can be approximated as linear at a specified local flight condition. If this assumption is valid, the model postulation and model structure determination steps are simplified, and there is extensive literature in linear systems theory that can be used to understand the model properties and to guide linear control law design. If aerodynamics were globally linear, the estimated coefficients, e.g. C_{m_α} in Eq. (1.4), would be constant across the entire flight envelope, and only a small sample of data would be needed to estimate the globally valid parameters. However, if a full-envelope model is required, even traditional aircraft aerodynamics, and certainly novel UAM vehicle aerodynamics, are known to be globally nonlinear. Therefore, the modeling process must account for these complexities to produce a model that is globally valid across the entire flight envelope.

This section will describe several practical approaches to nonlinear full-envelope aerodynamic modeling. Most of the methods discussed in Section 1.2.2 on the aircraft system identification process are valid across many of these approaches, but two particular aspects will be addressed here that set the approaches apart from one another: flight envelope coverage and global model architecture. While the discussion on experiment design in Section 1.2.2 focused on designing excitation inputs to generate flight data with sufficient information content, another part of this step

entails designing maneuvers to collect data across the full flight envelope to be modeled. The model structure determination step will also be further addressed here to focus on global model construction as either a combination of numerous local (linear) models, or as a single global nonlinear model.

1.3.1 Developing an aerodynamic database of local linear models

The traditional approach to full-envelope aerodynamic modeling based on flight data alone involves developing a local linear model at numerous flight conditions and test points, and stitching these models together in an extensive lookup table, or "aero database."

For this approach, the envelope expansion component of experiment design consists of designing an extensive test matrix that is parametrized by selected variables upon which the aerodynamics are nonlinearly dependent. These scheduling variables can include terms based on the flight condition such as angle of attack, sideslip, power setting, airspeed or Mach number, or those associated with the aircraft configuration such as effects related to flap setting, weight, and center of gravity position. At each discrete test point, the aircraft is trimmed and dynamic excitation inputs such as doublets, 3-2-1-1 pulses, or frequency sweeps are applied as small amplitude perturbations to maintain a linear approximation. Often, the excitations are applied as single inputs sequentially at each point to excite separate modes individually, requiring multiple maneuvers at each test point. Additional maneuvers

also need to be performed to acquire separate validation data. Such flight test campaigns typically extend over a period of several months consisting of a large number of sorties, hundreds of conditions, and thousands of maneuvers performed [37-39]. Furthermore, if the aircraft configuration is modified, the flight tests may need to be repeated.

Following the test flights, the techniques discussed in Section 1.2.2 may be used to estimate a linear aerodynamic model at each test point, referred to as point identification. These models are used in tabular lookup tables to fill a comprehensive aerodynamic database that is parametrized by the various scheduling variables. Sometimes multi-point identification is performed to combine several point models over a range of flight conditions, such as angle of attack, to develop a single nonlinear model that is representative of a broader range of data [38]. This database, similar to one built from wind tunnel data, can then serve as a tool for continuous aircraft simulation studies, where the aerodynamic model and trim condition at any particular point or configuration are obtained by multidimensional interpolation and methods such as model stitching [40-43].

1.3.2 Global polynomial aerodynamic modeling

A linear model approximation about a trim condition may be unsuitable for large amplitude maneuvers or highly nonlinear flight regions such as stall, and a nonlinear model structure is required. This section discusses approaches to developing an analytical global nonlinear model that is valid across the full flight envelope. For

global aerodynamic modeling, maneuvers are designed to provide wide data coverage of the explanatory variables across multiple flight conditions, and are no longer restricted to small amplitudes; as a result, the entire flight envelope can be covered more quickly over fewer flights. Past work has used long-duration high-amplitude maneuvers with excitation applied by the pilot [44-46].

L2F research has developed specialized flight test maneuvers to efficiently gather flight test data for global modeling. These methods are often applied in conjunction with the automated orthogonal optimized multisine excitation inputs discussed in Section 1.2.2. Those inputs are overlaid on the maneuvers to provide multi-axis excitation and flight data with high information content and low correlation across the explanatory variables. During the Bat-4 flight tests, the L2F maneuvers were inspired by a bird being dropped out of its nest, by beginning at a nose-high flight condition and slowly varying the nominal pitch flight condition down to steady-level flight, and then back up [7]. For the E1 tests, the ground-based pilot performed slow approaches to stall and figure eights to obtain global data across a wide range of flight variables [18]. For the manned Impala flights, the pilot performed multi-axis uncorrelated "fuzzy inputs" overlaid on slowly varying flight conditions, as well as Spiral Powered All-axes fuZzy (SPAZ) maneuvers to expand the variations in flight data to include Mach number, load factor, and dynamic pressure [12].

Past work has shown that global nonlinear aerodynamic models can be expressed as polynomial expansions with higher order and multivariate terms by extending the Taylor series expansion. The challenge then lies in establishing a pool of

candidate linear and nonlinear model terms known as regressors, such as in Fig. 1.5, and then selecting those that retain the strongest modeling capability. Past offline work has used engineering judgment to specify candidate regressors up to a specified order, and stepwise regression or multivariate orthogonal functions (MOF) to choose the relevant modeling terms [29, 47, 48]. Notably, the orthogonalization procedure has also been used to develop the generic global aerodynamic (GGA) model structure that is applicable across a wide range of aircraft [49]. Additional work has investigated the inclusion of spline functions in the regressor pool to improve the identification of localized aerodynamics [48, 50], while alternative approaches have considered multivariate simplex splines [44, 51]. L2F work has extended the multivariate orthogonal function modeling approach to a recursive formulation that can be used to efficiently model the nonlinear aerodynamics, and that has been successfully demonstrated in flight in real time across several flight vehicles [7, 12, 14, 18]. Another successful L2F global modeling approach uses frequency domain techniques to estimate local models for sequential windows of data in time as a function of each window's nominal flight condition, and then uses these local models to fill a global table [18].

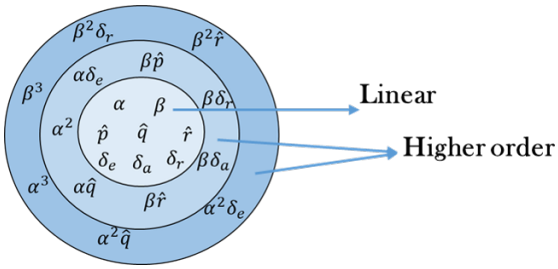


Figure 1.5: Global polynomial modeling by selecting from a specified pool of candidate modeling terms.

An alternative approach to global nonlinear aerodynamic modeling is known as data partitioning, through which the global aerodynamics are partitioned in the explanatory variable space into several regions where local (polynomial) models are estimated [21]. Whereas approaches like stepwise regression and MOF characterize nonlinearities by incorporating higher-order and multivariate modeling terms, the data partitioning method instead divides the flight data into regions known as cells where simple models can be used. The global nonlinearities are often characterized by particular explanatory variables, such as nominal angle of attack, which can be divided into local regions in which the effects of the nonlinearities are mitigated or removed. If the data are partitioned by Mach number and angle of attack, for example, the resulting cell structure might appear as in Fig. 1.6. If the local model can be approximated as linear, then extensive linear systems theory can be employed to investigate and understand the aerodynamics in each region. With certain parameter estimation methods, such as equation-error least squares in the time domain, the data associated with the models in each cell do not need to be contiguous in time, so flight test data across the global flight envelope can be sorted into the local region to which they belong. Data partitioning for aerodynamic modeling was first employed in Refs. [45, 46] for large amplitude maneuvers. In these cases, a fine prescribed resolution of cells was specified through extensive analysis of data density, and so the cell locations were not customized based on particular nonlinear variations in the data, which resulted in a complex cell structure. Local model networks (LMNs) extend the idea of data partitioning and provide an architecture for the global model by expressing the model output as a smoothly weighted combination

of the local models, and this approach was used to model the aerodynamic coefficients for nonlinear X-31 flight data [52-56]. Past L2F work also looked at data partitioning approaches within the context of fuzzy logic to update aerodynamic model parameters for the Impala aircraft in real time [12].

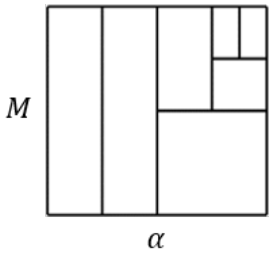


Figure 1.6: Cell structure partitioned by Mach number and angle of attack.

1.3.3 Other modeling approaches

Although polynomial aerodynamic model structures provide certain benefits such as simplicity and interpretable parameters, other model forms have also been used for aerodynamic modeling by considering a black box system and estimating an input-output model without prior knowledge needed to inform the model structure. This more flexible strategy allows for powerful computational modeling tools that span many different fields including machine learning and statistics, and that extend outside of the more traditional aerodynamic modeling approaches [22].

Neural networks offer a powerful tool for universal function approximation that has been applied to aerodynamic modeling. Many neural network models estimate the state derivatives or the total aerodynamic forces and moments. In Ref. [57], the back propagation method was used to train a feed-forward neural network (FFNN)

to estimate a flight dynamics model using simulated flight data, and in Ref. [58], the aerodynamic coefficients were estimated for a sparse wind tunnel and numerical simulation database. In Ref. [59], an extended Kalman filter was used to train a computational neural network to estimate the aerodynamic coefficients for simulated flight data with function-derivative training that minimizes not only the function error, but also the function derivative error. Reference [60] specifically applied neural networks to global simulated flight data up to an angle of attack of 60 deg using a variety of neural network structures of varying complexities.

Other neural network approaches aim to more specifically estimate the stability and control derivatives in addition to the force and moment coefficients. In Ref. [61], the "Delta method" was used with a FFNN to estimate the lateral-directional aerodynamic parameters by perturbing the input values in accordance with the definitions of stability and control derivatives, and training separate delta networks for each perturbed state to compute the finite differences. In Ref. [62], the Neural-Gauss-Newton method was used to estimate the aerodynamic derivatives with flight data from the Hansa-3 aircraft. Note that although neural networks are capable of complex nonlinear modeling, these approaches that focus on stability and control derivatives still use linear perturbation flight data. Neural network-based models have also been used for adaptive nonlinear control applications, such as for a nonlinear dynamic inversion control law [63, 64]. Other machine learning tools such as support vector machines have also been applied to aerodynamic modeling [65].

Models can also be developed as networks of other functions such as radial basis functions. In Ref. [66], a radial basis function neural network was used to

model the aerodynamic forces and moments, and a first order partial differentiation scheme was used to obtain the stability and control derivatives.

Wavelets can also be used for modeling, have been used across a wide array of different fields, and have valuable localization properties. Instead of partitioning the explanatory variable input space as in the data partitioning approach, wavelet functions can effectively partition the temporal and frequency components of the data with the localization attributes of the "daughter wavelets." Stemming from Fourier methods, wavelets possess properties of "compact support" compared to global sine and cosine functions, which allow them to effectively be used to represent varying windows of data. The chosen window sizes allow each wavelet to represent different features, both local and global, across the temporal and frequency domains. Wavelets are useful tools to analyze a wide variety of signals, as they have the capability to capture highly localized or non-smooth behavior such as spikes, as well as low-frequency larger-scale components, effectively extracting particular features of the global signal. In particular, wavelet thresholding methods allow a signal to be reproduced by thresholding, or removing, certain small components that describe details that are not important. This can be relevant for denoising applications by explicitly removing certain attributes without the need to smooth the entire signal, which has value for flight data that may contain significant noise [67, 68].

In aerodynamic modeling applications, Ref. [69] uses several types of wavelet functions to estimate black box models for the force and moment coefficients using the output error method for wavelet coefficient estimation, while Ref. [70] combines the localization benefits of wavelets with the learning tools of neural networks to

develop a wavelet neural network aerodynamic model.

1.4 Summary and limitations of previous approaches

Section 1.3 discussed several approaches to modeling the global full-envelope aerodynamics. The traditional approach is to collect small perturbation local linear models about trim conditions across the flight envelope into an aerodynamic database that is parametrized across several flight conditions and vehicle configurations. This method takes advantage of linear systems theory and offers the flexibility of using different modeling methods that are suitable for various data sets; however, flight test campaigns tend to include inefficient, extensive test matrices, and a global analytical model is not obtained. Instead, database interpolation is typically required.

Global nonlinear modeling methods are not restricted to small amplitude maneuvers, can be used to more efficiently obtain data throughout the flight envelope, and can provide an analytical model that is globally differentiable and that can be used with nonlinear control law approaches. A global nonlinear polynomial model can include higher order and multivariate modeling terms to capture the nonlinear effects across the full flight envelope; however, it may sacrifice the ability to characterize significant localized variations in order to capture the global system behavior. In the global form, it is also not necessarily apparent where within the input space the model is performing better or worse, or where uncertainty is higher due to factors such as limited data obtained. The inclusion of candidate spline functions in the

regressor pool can provide improved localization, but without knowledge of where within the explanatory variables these terms should be placed, a general expansive pool covering all practical possibilities may need to be specified, which would increase the computational load, particularly for real-time onboard applications.

When flight data cover a wide range of explanatory variables and cannot be adequately captured by a single linear model, several local models can also be used to describe the large amplitude or nonlinear behavior through data partitioning [21,36]. This approach converts a complex nonlinear modeling problem into several local, and possibly linear, models by dividing the flight data into multiple regions or cells, and estimating a separate model using the data in each cell. These approaches retain the easily interpretable polynomial model form, while providing a nonlinear model that is globally valid, and that still captures the local aerodynamics in a modular way. Drawbacks of data partitioning lie in the increased model complexity compared to global polynomial models, the decision making that is necessary to determine the cell partitions, particularly if conducted in real time, the requirement to combine the local models across the cell boundaries, and the requirement for sufficient localized data to accurately estimate the local models. Past work that used fuzzy logic modeling in this context employed global regression techniques across all cells, which resulted in correlated regressors, high condition numbers, and interdependent estimated parameters with high uncertainties that could not be interpreted individually. As a result, despite the localization aspects, the model was considered as a black box. Furthermore, for real-time operation the cell structure needed to be frozen while only the parameters could be updated recursively [11,12].

Finally, other modeling methods that span the field of machine learning, such as neural networks and radial basis function networks, take advantage of powerful learning tools. However, they often require a large amount of training data, are computationally demanding to an extent that can hinder real-time operation, and prioritize representation of the input-output relationships without consideration for physical interpretability. Neural network approaches require specifications of the hyperparameters that may be difficult to determine, such as the number of hidden layers, number of nodes in each layer, training method, activation functions, and training cycles. In general, for black box models that have hundreds of parameters, it is difficult to discriminate between signal and noise, the data can often be overfit, and the model can therefore offer poor prediction capabilities. Additionally, if the model fit is poor in a particular region of the explanatory variables, it can be difficult to understand the reason and to target a particular part of the model for improvement, because it lacks transparency.

Wavelet transforms offer a unique extension of Fourier methods, have vast flexibility, and can be used to capture both local and global time and frequency content across large sets of modeling data. However, the flexibility lends into the need to choose the right method parameters, including the types of wavelets, and the dilation and translation specifications.

1.5 Research approach, goals, and contributions

As discussed in Section 1.1, this work is motivated by the NASA L2F concept, which aims to develop real-time, automated, onboard system identification tools. Section 1.3 provides a literature review of existing modeling approaches that have been used for aerodynamic modeling, while Section 1.4 discusses some advantages and limitations of various categories of approaches. The intention of this research is to look at the modeling process and evaluate existing methods with a fresh perspective, so Section 1.2 provides a background of the broader context of the aircraft system identification process as a whole, as well as the wide array of modeling considerations and goals. Finally, this section will describe the particular goals and modeling considerations within this work, and how they led to a new modeling approach.

Several modeling goals were first specified within the context of L2F, and then existing methods were explored to achieve them. First, particularly in the context of novel aircraft designs and configurations, the intention is to learn as much as possible about the vehicle. Physical insight into the aerodynamic properties is therefore a priority for the model to provide. As a result, polynomial aerodynamic model forms were chosen to easily offer that insight through the estimated stability and control derivatives.

The modeling approach should also prioritize good local prediction capabilities and modular characteristics, which can be addressed with data partitioning methods. Many current methods of data partitioning require extensive post processing

and analysis that are not compatible with the L2F priorities [45, 46], or resort to bisecting split locations, which can be restricting and offer poor meaningful resolution [53-56]. In the interest of having a model that emphasizes parsimony, with physically meaningful localized cells, a new approach is needed that allows variable split locations within this framework.

The model should also provide good global prediction capabilities. The technique of local model networks offers the architecture for data partitioning by expressing the global model output as a smoothly weighted superposition of the local models that are weighted across the cell boundaries to avoid discontinuities. Current methods that use local model networks tend to be offline and iterative to develop the cell structure [12, 53-56], but in accordance with the L2F goals, this method needs to be compatible with real-time operation onboard the aircraft. Furthermore, the method should reliably distinguish between deterministic content and noise in the data to avoid overfitting.

Combining these goals and associated methods, a new integrated modeling approach was developed known as Smoothed Partitioning with LocalIzed Trees in Real time (SPLITR). Particularly in the context of L2F, the objective of this work is to develop a new approach to model the nonlinear aerodynamic force and moment coefficients automatically for fixed-wing rigid-body aircraft. The method should be compatible with real-time operation onboard the aircraft with limited *a priori* knowledge of the aerodynamics required. The SPLITR approach is applicable to more general vehicles, including those more aligned with UAM, as well as other nonlinear systems outside of aerospace, but the scope of this work is confined to

applying them first to more conventional vehicles for which the results could be easily validated and understood. This approach leverages a set of valuable and well-established methods that have reliably been used in the past, combines them in a new and unique way, and adapts them to real-time applications with a new cell splitting procedure.

The specified modeling goals pursued in this work are that the model should be developed automatically in real time with limited prior knowledge required, it should be easily interpretable, provide physical insight into the aerodynamics, and offer good global and local prediction capabilities. The SPLITR method expands on LMN methods to allow models with variable split locations, and with real-time updates to both the cell structure and parameter estimates. General technical challenges that this work faces are abiding by the real-time compatibility constraint, and developing methods that can be easily applied to different systems with minimal tuning.

In reference to the modeling considerations discussed in Section 1.2.1, this work focuses on online gray box parametric modeling for open-loop global nonlinear aerodynamics. For simplicity, the methods developed are applied to rigid-body conventional aircraft, while the underlying approaches are developed to offer future applications toward more complex UAM vehicles. Therefore, the prioritized model properties include interpretability to provide physical insight into these novel systems, parsimony to ensure a simple model structure, and modularity to ensure that poor data quality will not impact the entire global model.

The original contributions of this research include the development, testing, and validation of a novel approach to global nonlinear modeling using automated

LMNs in real time, that is known as Smoothed Partitioning with LocalIzed Trees in Real time (SPLITR). The SPLITR approach addresses the modeling goals specified, and offers a robust methodology that is generally applicable. This novel method is demonstrated across both generic simulated test data as well as experimental flight test data, and offers user insight into both the modeling process and the model itself. The test cases shown in this work are demonstrated using a full set of data that is processed as if each point is received in real time.

Although SPLITR was motivated by, and developed for, the purpose of aerodynamic modeling within the NASA L2F concept, the potential applications go far beyond the scope of aerodynamics to model more general systems with similar goals and priorities, and several examples are used to demonstrate the broader SPLITR capabilities.

1.6 Dissertation outline

This dissertation is organized into six chapters summarized as follows:

Chapter 1 describes the motivation behind examining the aircraft development process within the context of a new and rapidly changing environment for air vehicles, and compares the traditional aircraft development process with the efficiencies offered by the L2F concept. It provides background on the system identification process and modeling considerations applied to aircraft. It also includes a literature review describing existing approaches to aerodynamic modeling. The chapter concludes with an overview of the research approach discussed in this dissertation, as

well as the goals and contributions.

In Chapter 2, the method of LMNs is explored in depth, and the terminology, nomenclature, and modeling approaches are discussed. A literature review of related LMN work and an overview of existing algorithms are presented as well, which extend outside the scope of aerodynamic modeling.

In Chapter 3, the Smoothed Partitioning with LocalIzed Trees in Real time (SPLITR) modeling method is introduced, which builds on past LMN research and expands the capabilities and applications to real-time modeling. Real-time parameter estimation, cell structure determination, and global weighting are discussed in detail.

Chapter 4 explores the utility of the SPLITR method by testing it using simulated data from piecewise linear and higher order functions. The test data complexity is gradually increased to provide clear conveyance and visualization of the modeling process, and the sensitivities of the modeling results on SPLITR user inputs are also explored in depth.

Chapter 5 discusses the results of developing aerodynamic models using SPLITR with simulated and experimental flight data. First, simulated flight data are used from a nonlinear F-16 simulation, and then SPLITR is applied to experimental flight test data from the NASA E1 and T-2 test aircraft.

Finally, Chapter 6 summarizes the SPLITR modeling approach discussed in this dissertation and provides future directions and applications.

Chapter 2

Local Model Networks

This chapter provides an overview of both the background of local model networks (LMNs), as well as practical existing approaches and research that employ this theory. First, LMNs will be introduced, along with descriptions of various aspects including the cell structure, the leaf models, and the global model architecture. These sections are intended to convey the scope and breadth of LMNs, and how they can be customized to particular applications. Then a wide selection of LMN construction algorithms is presented to tie together the selected LMN components into unique approaches.

2.1 Introduction to local model networks

Modeling complex nonlinear systems is a challenging problem that remains an active area of research. If the system can be sufficiently approximated as linear at several relevant operating points, a common approach is to collect a set of linear models into a database and to interpolate between those points. This technique

takes advantage of the insight and analytical tools offered from linear systems theory, but can require extensive planning and lengthy execution for data acquisition. One approach that can utilize the advantages of linear systems but also leverage automated global model development tools is LMNs. Through this method, the global nonlinear function is expressed as a weighted combination of several simpler local piecewise models that are partitioned across the range of input variables. LMNs tend to offer advantages of interpretability, flexibility, simplicity, and modularity.

The concept of expressing a complicated global nonlinear function as a combination of smaller, simpler parts is well known and widely used, in congruence with a "divide and conquer" strategy. Consequently, there is significant overlap in the literature between what are referred to as radial basis function networks, neural networks, Takagi-Sugeno fuzzy models, neuro-fuzzy models, local model networks, regression trees, model trees, hinging hyperplane trees, and data partitioning, among others [22]. In fact, several of these methods can be considered equivalent under certain constraints. Although similar work can be found across each of these fields, exclusive semantics are often used within each category, and that can inhibit a comprehensive literature review due to limited cross-referencing. Extensive discussion of the similarities and differences between some of these methods is given in Ref. [22], but this dissertation proceeds under the label of local model networks. Nevertheless, to ensure a broad overview is presented, several relevant methods and algorithms that adopt other names across the fields of research mentioned above will be referenced and discussed.

A local model network is an architecture that is used to approximate a nonlin-

ear function as a weighted combination of several piecewise local models, as shown in Fig. 2.1 for a basic network consisting of $k = 1, \ldots, M$ cells, with input variables $\boldsymbol{u} = \begin{bmatrix} u_1 & u_2 & \cdots & u_{n_u} \end{bmatrix}^T$ and output $\hat{y}$.

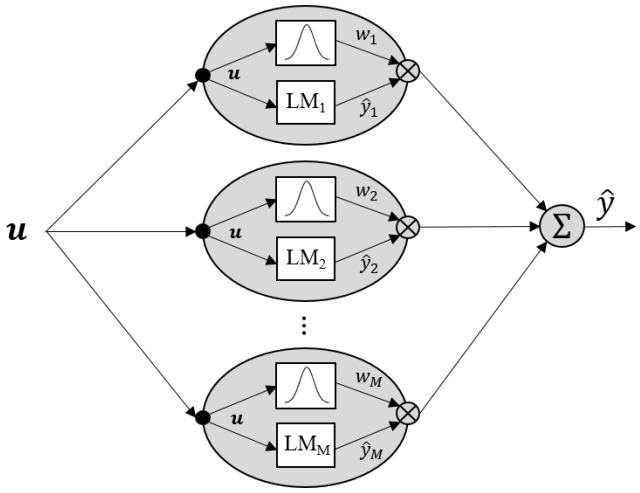


Figure 2.1: Local model network architecture with M cells (adapted from Ref. [71]).

Each cell, also known as a leaf, includes a local polynomial model, and a weighting function that describes the region of validity across the global domain of the input variables. The output of the k^{th} local model is

$$\hat{y}_k = \theta_{k0} + \theta_{k1}u_1 + \theta_{k2}u_2 + ... + \theta_{kn_u}u_{n_u} \tag{2.1}$$

where θ_{kj} is the parameter associated with the j^{th} input term for the k^{th} local model. The normalized validity function for the k^{th} local model, $w_k(\boldsymbol{u})$, which is a nonlinear function of $\boldsymbol{u}$, weights each local model and represents the strength of influence of the local model across the input variables. The validity functions commonly take the form of normalized Gaussian functions in LMNs. If the weight associated with

the k^{th} cell is $\hat{w}_k(\boldsymbol{u})$, then the normalized validity function is given as

$$w_k(\boldsymbol{u}) = \frac{\hat{w}_k(\boldsymbol{u})}{\sum_{j=1}^{M} \hat{w}_j(\boldsymbol{u})} \tag{2.2}$$

The validity functions are normalized to ensure a partition of unity such that the weights across all models sum to 1 at every point in the input space.

$$\sum_{k=1}^{M} w_k = 1 \tag{2.3}$$

The global nonlinear model output $\hat{y}$ is the superposition of the estimated local model outputs $\hat{y}_k$ that are weighted with the normalized validity functions, and is expressed as

$$\hat{y} = \sum_{k=1}^{M} (\theta_{k0} + \theta_{k1}u_1 + \theta_{k2}u_2 + ... + \theta_{kn_u}u_{n_u})w_k = \sum_{k=1}^{M} \hat{y}_k w_k \tag{2.4}$$

A local model network is generally a modeling approach that is used to approximate a nonlinear function as a weighted combination of piecewise local models, but there is a vast amount of flexibility in how the approach can be formulated. The following sections will describe several LMN properties and associated options that are relevant to this work to portray the extensive nature of LMNs. They will also provide background that will show the limitations in current methods and justify the new real-time LMN approach presented in Chapter 3.

For clarity, the LMN properties can be divided into three categories: 1) cell structure, 2) local models, and 3) global model architecture. Throughout the liter-

ature, each LMN approach is characterized by a selection of these properties along with a unique logic to guide the model development process through growing the cell structure. Several related algorithms will be discussed in Section 2.5.

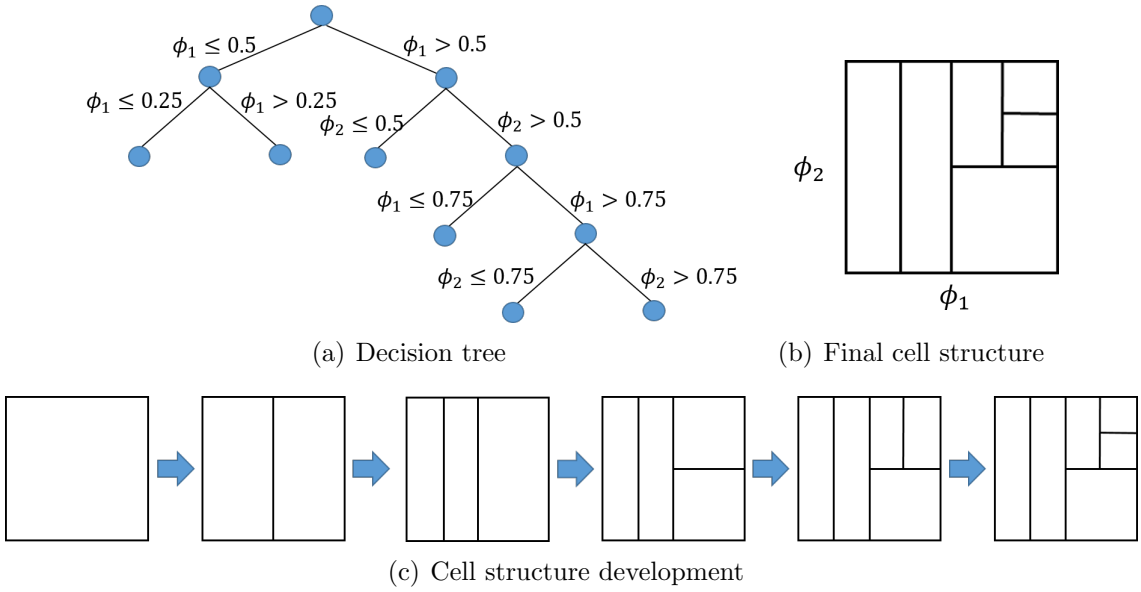
2.2 LMN cell structure overview

The LMN cell structure describes how the input space is partitioned into a set of local cells, each of which includes a local model that describes the system behavior in that region, and an associated validity function that describes the relevance across the global domain. The cell structure is intended to partition the nonlinear regions of the input space so that simpler local models are valid in each region. The cell boundaries can be designated through prior intuition, analysis of the system's nonlinear behavior, or through properties such as data density. Alternatively, they can be specified as a grid of constant or varying resolution across each dimension. However, partitioning for nonlinear behavior can be difficult to reliably predict, and if the cell locations are not customized based on particular nonlinear variations in the data, an overly complex cell structure may result with redundant local models and limited physical insight from the boundary locations. An alternative approach that this section focuses on involves developing the cell structure automatically by input space decomposition through recursive partitioning in a tree construction algorithm. In this approach, an initial coarse cell structure can also be specified initially to enforce certain cell boundaries, and then it can be further developed through data-driven methods.

The input space decomposition can be visualized through a regression tree to describe the sequential partitioning and to locate the cell to which a particular data point belongs. An example of a 2D regression tree and the corresponding final cell structure is shown in Figs. 2.2(a)–2.2(b) for two specified partitioning variables, ϕ_1 and ϕ_2. The root leaf at the top of the tree contains all of the training data across the input space. Each leaf split represents a binary partition of the parent cell into two child leaves, where the branches specify the locations across the variables that divide the parent cell. A data point belongs to a cell if it falls within the cell's partitions, and to locate the final cell, a path is traced from the root leaf through the interior leaves to a terminal leaf at the bottom of the tree by following the sequence of inequalities through the branches. Classic regression trees provide piecewise constant models in each terminal leaf. In other cases, a local polynomial regression model is given in each cell, which is evaluated to produce the local decision tree output.

Although the decision tree and cell structure appear to show "hard splits" which indicate a global piecewise model and imply that each data point belongs exclusively to a single terminal cell, in the LMN approach each data point can activate multiple cells due to overlapping interpolation regions defined by validity functions. The global LMN output will then be a weighted combination of several local cell outputs. Nevertheless, these depictions are still useful for visualizing the sequential partitioning and final structure, and for locating the primary cell where the model will be weighted most strongly for a particular data point. Note also that the decision tree is not a unique sequence to a particular final cell structure,

nor does it clearly show the order of all splits, but it illustrates the sequential input space decomposition and has implications for tree pruning. A unique depiction of the cell structure development sequence is instead shown in Fig. 2.2(c).



(a) Decision tree (b) Final cell structure

(c) Cell structure development

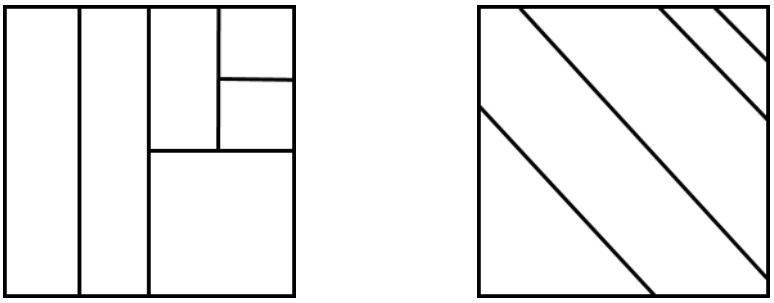
Figure 2.2: Regression tree construction and visualization.

In many cell structure development methods, a binary recursive tree construction search algorithm automatically decomposes the input space through a series of splits and data-driven decision making where at each step, an existing cell is split into two child cells until a stopping criterion is met. Section 2.5 will present several recursive partitioning algorithms and describe their individual approaches and decision-making logic.

The cell structure shown in the example in Fig. 2.2 is constrained to both axes-orthogonal and bisecting splits, but both the dimensionality and location of the split in each step can be different.

2.2.1 Split dimensionality: axes orthogonal vs. axes oblique

Axes-orthogonal partitions are drawn through a single dimension so that the cell boundaries are orthogonal to the input variable axes, as shown in Fig. 2.3(a) for two dimensions. This approach maintains the benefit of simplicity and interpretability of the resulting cells, and results in a hypercube cell composition. Axes-oblique partitions, as shown in Fig. 2.3(b), allow arbitrary splits that extend through two or more dimensions at once, and while they permit a more flexible cell structure, multi-dimensional splits can add complexity to the split logic and can sacrifice model transparency. Nevertheless, for high-dimensional problems with strongly coupled nonlinear dependencies, axes-oblique partitioning offers a more efficient strategy.



(a) Axes orthogonal (b) Axes oblique

Figure 2.3: LMN split dimensionality options.

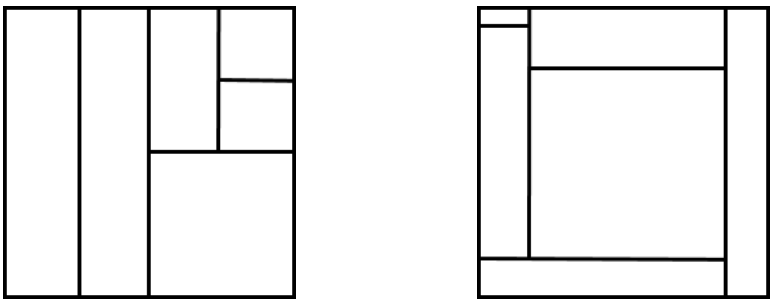
For axes-orthogonal partitioning schemes, the decision making surrounding each split consists of choosing the parent cell, the split dimension, and the split location within that cell. For axes-oblique partitioning, the boundary is often described by a sigmoidal shape where the parameters are optimized, thus expanding the complexity of the decision making. Consequently, axes-oblique partitioning strategies

tend to use more computationally expensive nonlinear optimization methods to calculate the cell boundaries.

2.2.2 Split location: bisecting vs. customized

In many axes-orthogonal partitioning schemes, the parent cell is bisected through a single dimension during each split as shown in Fig. 2.4(a), thus eliminating the split location determination and simplifying the required logic and computation. However, this approach can be highly restricting, particularly if the nonlinearities are biased toward one side of the cell, and it can result in redundant cells, overly complex models, and less meaningful cell locations.

For additional flexibility, a selection of candidate discrete split locations in each dimension can be proposed, and the best location that minimizes a specified cost function can be chosen. Alternatively, an arbitrary axes-orthogonal split location can be determined through an optimization or other process. Both of these options will produce a more customized and physically meaningful cell structure, as shown for example in Fig. 2.4(b), at the expense of added computation. For axes-oblique partitioning strategies, the split location is often optimized along with the dimensionality, as discussed in the previous section.



(a) Bisecting (b) Customized

Figure 2.4: LMN split location options.

2.3 LMN leaf models overview

Each cell contains a local model that describes the data in the region of validity of that cell. Most generally each local model can be considered independent of the others, and can be specified through first principles, *a priori* information, or otherwise separately. Despite this flexibility in specifying each local model independently, it is more common that all of the local models are developed together through least-squares regression during the cell structure development process. Even under the constraints of linear regression methods, there are still several attributes that allow flexibility in the resulting models, which will be discussed in this section.

2.3.1 Model inputs: explanatory variables and partitioning variables

The LMN input variables $\boldsymbol{u} = \begin{bmatrix} u_1 & u_2 & \dots & u_{n_u} \end{bmatrix}^T$ to each cell include both the explanatory variables (EVs), $\boldsymbol{x} = \begin{bmatrix} x_1 & x_2 & \dots & x_{n_x} \end{bmatrix}^T$, which are used as regressors for local function approximation, and the partitioning variables (PVs),

$\boldsymbol{\phi} = \begin{bmatrix} \phi_1 & \phi_2 & ... & \phi_{n_\phi} \end{bmatrix}^T$, which are used to partition the global model and to evaluate the validity functions. The variables $\boldsymbol{x}$ and $\boldsymbol{\phi}$ can be selected independently and may be specified to include all, some, or none of the same terms.

In the simplest case, and how it has been presented up to this point, $\boldsymbol{x} = \boldsymbol{\phi} = \boldsymbol{u}$, the EVs and PVs are equivalent, and the k^{th} cell's local model and validity function are given as

$$f_k(\boldsymbol{u}) = \hat{y}_k = \theta_{k0} + \theta_{k1}u_1 + \theta_{k2}u_2 + ... + \theta_{kn_u}u_{n_u}, \qquad w_k = w_k(\boldsymbol{u}) \tag{2.5}$$

If $\boldsymbol{x} \neq \boldsymbol{\phi}$, the EVs are different from the PVs, and the local model and validity function are

$$f_k(\boldsymbol{x}) = \hat{y}_k = \theta_{k0} + \theta_{k1}x_1 + \theta_{k2}x_2 + ... + \theta_{kn_x}x_{n_x}, \qquad w_k = w_k(\boldsymbol{\phi}) \tag{2.6}$$

where $\boldsymbol{x}$ and $\boldsymbol{\phi}$ can either be disjoint to contain separate sets of variables, or they can overlap to include common terms. The PVs that are not included in the EVs can be interpreted as variables that define the operating point of the local model and are considered scheduling variables for which the local model does not have a functional dependency, e.g. system configuration variables.

Any *a priori* knowledge about the behavior of the response variable can be used to specify the EVs and PVs. Many LMN construction algorithms will automat-

ically choose a subset of the specified PV pool with which to partition the model; nevertheless, it is important to designate a meaningful, relevant pool to reduce the curse of dimensionality, improve computational speed, and preserve model interpretability and robustness by preventing illogical partitions that may result from non-optimal split decisions.

There are generally two ways that the LMN is evaluated for each local model by processing the EVs. The first approach is to use global coordinates, i.e. by using the true values of the EVs across all of the cells in the entire input space. This would imply evaluating the local model output equation as it is presented in Eqs. (2.5–2.6). However, this is contrary to the typical representation of the Taylor series expansion about a specified operating point, and in this case, the bias term θ_{k0} is not easily interpretable since it is not the output of the k^{th} local model at the center of the associated validity function. The second approach is to evaluate the local model output equation using relative coordinates. In this case, the reference cell center points of each local model are subtracted from the EVs before evaluating the output equation. Both representations produce the same model output, with the difference lying in the magnitude and interpretation of the bias term [22].

2.3.2 Model order: constant, linear, or higher order polynomials

Each local model is expressed as a polynomial and can be a constant (zero order), linear (first order), or higher order polynomial. Linear polynomials are

commonly used because of the extensive linear systems theory for interpretation, analysis, and control law design, and because they tend to provide sufficient small perturbation models. However, for a highly nonlinear system or region of the PV input space, an increase in local model complexity by employing a higher order polynomial can reduce the LMN complexity by resulting in fewer required cells to adequately capture the behavior in that region. Thus, there is a tradeoff between local model order and the number of LMN cells, and as M decreases to 1, a single multivariate polynomial model emerges, similar to the global polynomial modeling approach discussed in Section 1.3.2. Again, any *a priori* information can be used to specify the local model order, either in general across all cells, or particularly for regions in which that insight is available.

2.3.3 Model structure: pre-specified vs. automated selection

Many LMN algorithms have the capability of automatically choosing a subset of the specified PV pool through which to partition the model. The selection of relevant EVs for each local function approximation can be specified *a priori*. Alternatively, the EV selection may be automated using model term selection methods such as stepwise regression to choose only those linear and/or higher order terms that retain significant modeling capability. Even for the strictly linear model structures, data correlation among regressors or poor SNR can degrade model quality, and model structure determination remains to be an important and involved step in

the modeling process. Furthermore, with no regressor customization, the premise is that every modeling term is relevant across the entire PV input space, which may not be the case, and for which a subset selection may produce a more parsimonious local model. Nevertheless, local model structure determination adds complexity and additional computation to the local model development.

2.3.4 Parameter estimation process: global learning vs. local learning

Each local model is a polynomial that is linear in the parameters, so equation-error least squares can be used for parameter estimation. For a given cell structure and validity functions assigned to each cell, there are two main approaches to LMN parameter estimation: global and local optimization [22, 23, 72].

In the global parameter estimation approach, the parameters for all of the local models are optimized simultaneously in a single global regression where the regressors in each cell are weighted according to that cell's validity function. For simplicity, the equations will be expressed for a case where all local models contain the same terms. For an LMN with n_x input terms, N data points, and M local models, the least squares cost function is given as

$$J(\boldsymbol{\theta}) = \sum_{j=1}^{N} \left(y(j) - \hat{y}(j)\right)^2 \tag{2.7}$$

The global model output is then expressed as

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{\theta}} = \begin{bmatrix} \boldsymbol{X}_1 & \boldsymbol{X}_2 & \cdots & \boldsymbol{X}_M \end{bmatrix} \begin{bmatrix} \boldsymbol{\theta}_1 \\ \boldsymbol{\theta}_2 \\ \vdots \\ \boldsymbol{\theta}_M \end{bmatrix} \tag{2.8}$$

where the regressor matrix for the k^{th} cell is given as

$$\boldsymbol{X}_k = \begin{bmatrix} w_{k0}(\boldsymbol{\phi}(1)) & w_{k1}(\boldsymbol{\phi}(1))x_1(1) & \cdots & w_{kn_x}(\boldsymbol{\phi}(1))x_{n_x}(1) \\ w_{k0}(\boldsymbol{\phi}(2)) & w_{k1}(\boldsymbol{\phi}(2))x_1(2) & \cdots & w_{kn_x}(\boldsymbol{\phi}(2))x_{n_x}(2) \\ \vdots & \vdots & \ddots & \vdots \\ w_{k0}(\boldsymbol{\phi}(N)) & w_{k1}(\boldsymbol{\phi}(N))x_1(N) & \cdots & w_{kn_x}(\boldsymbol{\phi}(N))x_{n_x}(N) \end{bmatrix} \tag{2.9}$$

Then the parameters across all cells are simultaneously estimated as

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{2.10}$$

Global optimization directly recognizes that the global output of the model is defined as a weighted combination of the local model outputs that are weighted according to their validity functions, and therefore estimates all parameters at once through a single cost function minimization. In this case, the regression matrix consists of sets of columns that include the weighted regressors for each local model, and the estimated parameter vector includes the sets of parameters across all local models. As a result, this approach is computationally demanding for problems with

a large number of local models and/or regressors. Furthermore, global optimization problems tend to have high correlation from the repetition of the regressors in each local model that only differ through their respective weights, and because much of the regression matrix is close to zero since only a small number of local models are highly active at any point in the PV input space. Consequently, the estimated parameters tend to have high uncertainties, and they are interdependent and cannot be interpreted individually, but rather only collectively with the other local models. As a result, the model loses much of its transparency, which is an important aspect of the motivation for LMNs.

Alternatively, local learning can be performed by treating the local models independently and posing a separate weighted regression problem for each local model to estimate the associated parameters by neglecting interaction with the other local models. Instead of a global regression to estimate all $M(n_x + 1)$ parameters at once, M separate regression problems are performed to estimate the $n_x + 1$ parameters for each local model individually. A weighted least squares optimization is performed for each local model where the weighting function is equal to the validity function.

The local learning approach is advantageous over the global optimization since it is more computationally efficient, the regression matrix is better conditioned, each local model can be estimated individually using different optimization methods, and the local model parameters can be interpreted individually. Furthermore, the local approach is more conducive to online learning with regards to both the numerical stability of updating the local parameters using the recursive least squares regression

algorithm, as well as updating the cell structure since the global regression is directly dependent on the cell structure.

While many LMN approaches use weighted local regression, the estimated parameters are influenced by the chosen weighting function, which may be arbitrarily specified based on the cell structure and not directly by the data. The weighting functions are not only often dependent on the current cell structure, but since each local model weight is normalized by the sum of the weights from all other local models as in Eq. (2.2), each local model is in fact co-dependent to some degree, even though the regression is performed independently. This is particularly relevant in real time if a split is made in one area of the PV input space, and the weighting functions — and parameters — of the other cells are still affected.

Another approach is to consider each local model entirely independent from the neighboring cells and to estimate the local model parameters with ordinary (unweighted) least squares using only the data points within the cell boundaries. Then, the global weighting problem can be considered separately by blending the local models afterwards. This method provides an isolated model of the data within each region, and the flexibility of adjusting the weighting functions. For real-time applications, this also offers the flexibility that the local models are independent in each region and are therefore not affected by the changing cell structure and associated weighting functions. In this approach, the least-squares cost function for the k^{th} cell with N_k data points belonging to it is

$$J_k(\boldsymbol{\theta}_k) = \sum_{j=1}^{N_k} (y_k(j) - \hat{y}_k(j))^2 \tag{2.11}$$

Further discussion comparing the parameter estimation approaches for LMNs can be found in Refs. [22, 23, 72].

2.4 Global LMN architecture overview

As discussed at the beginning of this chapter, numerous modeling approaches simplify a problem by breaking it down into several smaller simpler parts. Unlike with classical decision trees, the ultimate goal is not a collection of discontinuous piecewise models, but rather a global continuous model that is expressed as a combination of the local models. This section will discuss several ways that the local models can be combined, both through the weighting functions as well as the global model construction.

2.4.1 Validity functions: Gaussian weights and a partition of unity

Validity functions are used across a variety of disciplines under different names such as basis functions in basis function networks, activation functions in neural networks, and membership functions in fuzzy logic. The common operational utility is to provide a smooth transition between piecewise components in a global architecture, and to characterize the regions of validity of each local model by weighting their contribution to the global output across the ranges of relevant input variables.

The validity functions can be chosen from a variety of different shapes, including Gaussian, ramp, and boxcar functions, for example.

In local model networks, similar to radial basis function networks, Gaussian functions are often used, which can be easily generalized to multiple dimensions. Gaussian functions consist of two parameters: the center, which is located where the associated model has the strongest validity; and the standard deviation, which characterizes the influence of each local cell across the range of the PV. The validity function for the k^{th} local model is given as

$$\hat{w}_k(\boldsymbol{\phi}) = exp\left[-\frac{1}{2}\sum_{j=1}^{n_\phi}\frac{(\phi_j - c_{k,j})^2}{\sigma_{k,j}^2}\right] \tag{2.12}$$

where c_k is the center of the Gaussian function, and σ_k is the standard deviation. The Gaussian functions are often placed in the center of the cells, and the standard deviation is a function of cell width such that $\sigma_{k,j} = 0.4\lambda_s(b_{k,j} - a_{k,j})$, where λ_s is the smoothness factor that describes the influence of each cell, and $[a_{k,j}, b_{k,j}]$ define the range of the k^{th} local cell across the j^{th} PV. The scaling factor of 0.4 is an empirical result from Ref. [52], such that a nominal smoothness factor of $\lambda_s = 1$ can be used.

A partition of unity, which is introduced in Section 2.1, is used to ensure that the weights from all local models add to 1 at every point in the input space. When this is enforced through Gaussian validity function normalization, it can cause unintended consequences by changing the shapes of the normalized Gaussian functions, particularly for cells of largely varying widths. These side effects can include shifting the maxima from the cell center, and reactivation, or multi-modal behavior [73].

The implications for the global model output can be significant if the output shows waves as a result of the overlapping weights. To combat this, some approaches use a hierarchical tree structure and sigmoidal weighting functions, which will be introduced in the next section.

Finally, for simplicity the validity functions and the global input space are typically defined in a normalized coordinate system from 0 to 1, so the validity functions can be specified in this normalized space. For real-time applications, this implies specifying the minimum and maximum expected values for each PV for normalization.

2.4.2 Global LMN form: combination of local model outputs vs. combination of local model parameters

Consistent across LMNs is that the global nonlinear model output is computed as a weighted combination of the local models, but there are several ways that this calculation can be formulated, and the chosen method will drive other LMN properties. In the most common formulation, the global model output is expressed as a linear combination of the local model outputs that are weighted according to their validity functions, given in Eq. (2.13) for the i^{th} data point, and shown in Fig. 2.5.

$$\hat{y}(i) = \sum_{k=1}^{M} \hat{y}_k(i) w_k(i) = \sum_{k=1}^{M} \boldsymbol{x}(i) \hat{\boldsymbol{\theta}}_k(i) w_k(i) \tag{2.13}$$

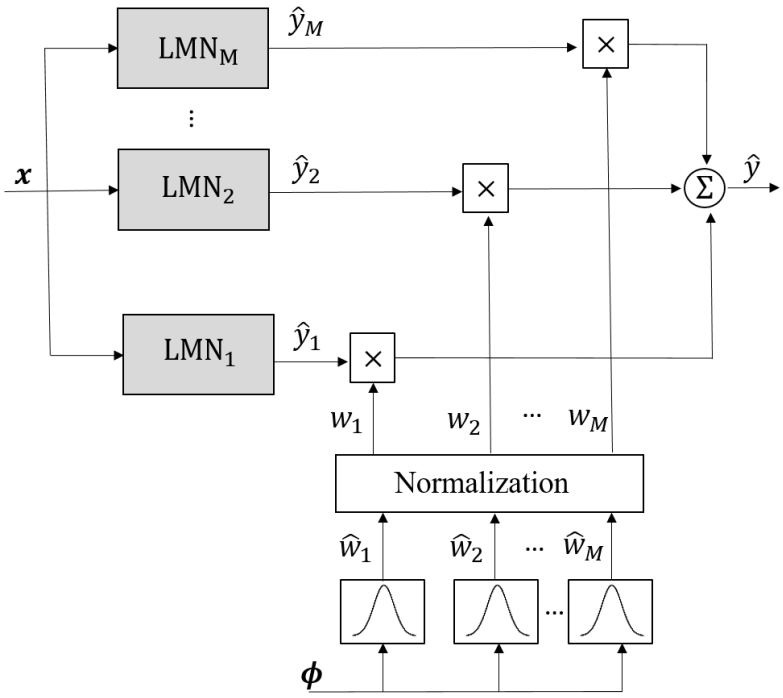


Figure 2.5: Global model output as a weighted combination of local model outputs (adapted from Ref. [74]).

A benefit of this formulation is that the global model can be constructed from a collection of local models that are not constrained to have the same model structure. Instead, the local models can each be expressed separately according to a mixture of *a priori* information and data-driven methods. Even if the local models are restricted to be polynomials, then this architecture allows the flexibility of various model structures and orders in different parts of the operating envelope.

Another architecture can be used if all of the local models have the same model form and model structure. In this case, the global model is a weighted combination of each of the common parameters from the local models, and therefore has the same model structure. This can be interpreted such that the global model parameters are scheduled according to the PVs and change throughout the operating

envelope, but the model structure is constant, similar to a linear parameter varying (LPV) model. Accordingly, the global model is much more transparent and can be interpreted at each operating point using the scheduled parameters. However, enforcing a constraint of a constant model structure across the entire PV input space eliminates the customization of model structure at particular conditions. The LMN output in this case is expressed in Eq. (2.14) for the i^{th} data point, and shown in Fig. 2.6.

$$\hat{y}(i) = \boldsymbol{x}(i) \sum_{k=1}^{M} w_k(i) \boldsymbol{\theta}_k \tag{2.14}$$

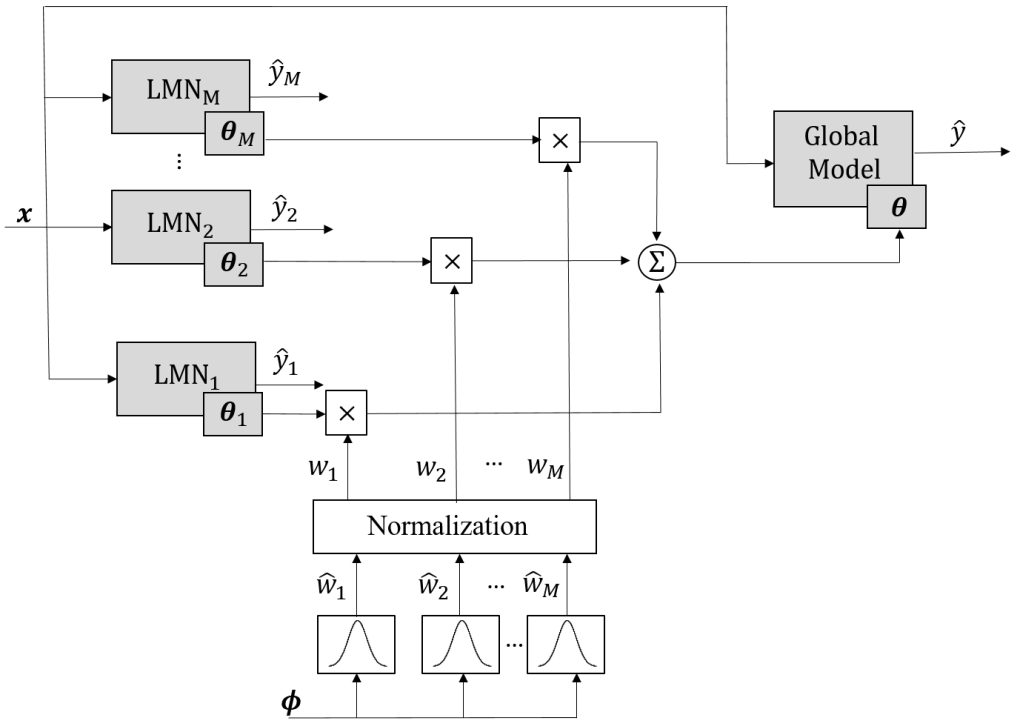


Figure 2.6: Global model output as a weighted combination of parameters across local models (adapted from Ref. [74]).

Although the cell structure is often developed in an iterative manner, as de-

picted as a tree in Fig. 2.2(a), the LMN for any given cell structure is considered "flat" if the model output is dependent only on the terminal leaves and their validity functions. A flat LMN output is straightforward to compute, and only the final cells and validity functions need to be stored.

Another approach is to have a hierarchical cell structure where although only the terminal leaf local models are used to compute the model output, the rest of the tree structure is maintained to calculate the associated validity functions. These hierarchical approaches tend to use sigmoidal functions for each binary split, which can be optimized to allow axes-oblique splits. Although hierarchical models are more complex to interpret, store, and compute, the nature of the binary sigmoidal validity functions at each level mitigates the normalization side effects of classical Gaussian validity functions. This approach has significant overlap with hinging hyperplane methods [75-77].

2.5 A literature review of LMN construction algorithms

The previous sections introduced many of the components and properties of LMNs to impart the depth and flexibility of this modeling approach. Accordingly, there is a lot of research that has explored these properties and numerous tree construction algorithms have been developed, each with a unique approach that is characterized by a selection of attributes discussed previously (among others), along with a splitting logic that guides the cell structure development process. This section summarizes some of these individual approaches to tie together the various

aspects of LMNs that have been discussed previously, and to describe how they can be implemented. Most work focuses on batch algorithms which operate offline, rely on iterative processes, and for which all of the training data are available in advance. A few techniques that address online adaptation are also introduced. Lastly, several LMN algorithms that specifically address aerodynamic modeling are discussed.

As detailed in the previous sections, from the split dimensionality to the regression to the global model construction, there are inherent and reciprocal advantages and drawbacks to different choices, and it is impossible to avoid all possible shortcomings in a single approach. Therefore, when choosing or developing a particular approach, it is important to specify, understand, and balance the model priorities, some of which are discussed in Section 1.2.1. One of the most common tradeoffs is between model complexity vs. interpretability. For example, axes-oblique splits may result in fewer cells, but the resulting model structure will be less transparent than for a model with axes-orthogonal splits. Also, additional levels added to the tree may improve the accuracy, but can reduce the robustness and predictive capabilities. The tree construction algorithm stopping criteria are therefore important factors that affects the model robustness, prediction, complexity, and overall performance, with the ultimate goal of producing the smallest tree that provides the desired accuracy, i.e. a parsimonious model. Splitting metrics and stopping criteria can involve options such as minimum number of allowable data points in each leaf, when a global cost function is not improving, statistical metrics such as RSS or MSE, or hypothesis testing on the residuals. Many methods combine a "growing" phase with an important "pruning" phase to find the final cell structure. In

general, the splitting logic across different algorithms must balance computational effort, effective stopping criteria, and performance of the resulting model. A real-time requirement eliminates some of these possibilities, given current computational capabilities. The following discussion of algorithms is intended to provide a broad overview of related work.

2.5.1 Batch LMN construction algorithms

Classification And Regression Trees (CART) is a generic decision tree building approach introduced by Breiman in 1984 that serves as a foundation for many decision tree algorithms [78]. At each level in the binary tree building process, all split dimensions and locations are considered for the next split, and the "best" candidate is chosen through a quantitative metric. The basic CART approaches develop piecewise constant models and switch discretely between submodels, so the submodels are discontinuous and there is no interpolation. Consequently, there is a sharp transition across cell boundaries and the global model is not smooth or differentiable. CART methods strongly emphasize the step of pruning by greedily expanding the tree size and then removing irrelevant leaves through cross-validation. MATLAB also has classification and regression tree objects and software in the Statistics and Machine Learning Toolbox that expand on CART theory [79]. RETIS [80] is an early method that expands on CART to allow local linear models in the cells. It also uses a Bayesian approach for the tree construction and the residual sum of squares (RSS) as the comparative cell metric.

SUPPORT (Smoothed and Unsmoothed Piecewise-POlynomial Regression Trees) [81] is an axes-orthogonal recursive partitioning algorithm that uses polynomial local models of fixed order where partitioning is informed by analyzing the distribution of residuals through t-tests. For a given leaf considered for splitting, the distributions of positive and negative residuals are analyzed by testing for differences in the means and variances from each set of data points, with the underlying premise that if the current model is sufficient, the residuals should not be significantly different. Cross-validation is then performed to determine if the predicted mean squared error (PMSE) exceeds a pre-defined threshold. If further splitting is justified, the split location is chosen as the average of the mean values in each set of residuals.

GUIDE (Generalized, Unbiased Interaction Detection and Estimation) [82] expands on the SUPPORT split logic to allow detection of pairwise interactions between variables. The data for each variable are divided into quartiles and used to fill a 2×4 contingency table where the rows are based on the sign of the residual. The χ^2 statistic and associated p-value are computed for each region in the curvature test. Pairwise interaction between variables is explored by dividing the 2D space into quadrants based on the sample median, building another 2×4 contingency table, and similarly computing the χ^2 statistic and p-value for the interaction test. The split variable is chosen based on comparing the p-values from each of the tests, and the split location is the sample mean of the chosen variable. GUIDE also allows the input variables to be chosen for regression, splitting, or both, and relies on a pruning stage, similar to CART.

SECRET (Scalable EM and Classification based REgression Trees) [83] offers axes-orthogonal or axes-oblique partitions for linear leaf models. It uses the expectation-maximization (EM) algorithm to convert a regression problem into a classification problem by finding two Gaussian clusters, and then assigning each data point to the cluster to which it has the higher probability of belonging. Classification tree techniques are then used to determine the split location as a hyperplane that minimizes a cost function.

MARS (Multivariate Adaptive Regression Splines) [84] is not considered a decision tree method, but rather performs recursive partitioning using spline functions in each cell. The approach emphasizes continuity, and the model developed through MARS can only be interpreted globally, at the expense of local model interpretability and transparency. MARS is a highly iterative, computationally expensive algorithm that relies on a forward (growing) phase and a backwards (pruning) phase to determine the number and locations of knot points. During each phase, numerous candidate knot locations are considered for inclusion or removal, and the next model iteration chosen is that which minimizes the residual sum of squares (RSS).

Model tree algorithms that have been developed by Nelles and others offer a wide array of flexibility and user preferences. LOcal LInear MOdel Trees (LOLIMOT) [22, 71, 85] is an axes-orthogonal algorithm that emphasizes simplicity in model construction and interpretation over complex optimization. Beginning with an initial model and cell structure, the local model that is performing most poorly is identified as that with the largest cost function defined as the weighted squared local model errors, and that local model is split next. Bisecting axes-orthogonal

splits are considered in each dimension, and a candidate model is calculated for each possible split adding one additional cell, using weighted least squares regression with Gaussian validity functions. The best division for each iteration is chosen from among the candidate models as that which most significantly improves a global cost function. Finally, a convergence test is performed to determine if the model meets a termination criterion to stop the additional splitting.

POLYnomial MOdel Trees (POLYMOT) [86, 87] builds on LOLIMOT such that, in addition to considering candidate bisecting splits for the worst local model during each iteration, the model could instead be modified by adding additional (higher order) terms using weighted stepwise regression. This approach balances the tradeoff of added complexity characterized by number of cells vs. number of parameters within each cell using a specified tradeoff complexity factor.

Finally, HIerarchical LOcal MOdel Trees (HILOMOT) [88] is an axes-oblique partitioning algorithm that uses a hierarchical tree structure, and can be considered a hinging hyperplane tree approach. The tree construction logic is also similar to LOLIMOT, except the worst local model can be split in an axes-oblique manner by utilizing nonlinear optimization to determine the split location and direction parameters of a sigmoidal splitting function.

2.5.2 Online LMN construction algorithms

Although less prevalent in the literature, there has been some work devoted to online tree construction algorithms as well. Often, these methods are closely linked

to an offline batch approach with modifications to allow for real-time, recursive operation. This can be useful for a time-varying model, an adaptive system, or for when the data are not available in advance.

There are two main approaches to online adaptation: freezing the cell structure and validity functions and only optimizing the local model parameters with new data in real time, or also allowing the cell structure to be grown as well. In Ref. [22], LOLIMOT was extended to both forms of online adaptation. For parameter optimization, recursive weighted least squares is used to develop the local model in each cell with new information. Ordinarily, with normalized Gaussian validity functions, every point theoretically activates every cell and is used to update every local model's parameters. To reduce the computational load in real time, an activation threshold can be implemented such that an individual point only adjusts the parameters of the cell(s) that it most strongly impacts.

The LOLIMOT approach to online adaptation of the cell structure is similar to the offline approach. For a given cell structure following a split, each cell initializes two background bisecting models in each dimension that are updated along with their active parent cell with the new information. If only the most active cell is updated with the new data point, then $2n_\phi + 1$ total cells will be updated, including the parent cell and the background submodels. If the data in a cell indicate nonlinear behavior, the partitioned submodels will perform better than the parent to justify a split. The candidate splits in a certain dimension will replace the parent cell if the sum of the local cost functions of the background models (multiplied by a factor) is less than the parent cost function. However, the approach of background submodels

can be computationally expensive, and the choice of the improvement factor is very difficult to specify in practice, and is not robust to noise variations.

Potts introduces two additional online axes-orthogonal approaches in an Incremental Model Tree Induction (IMTI) algorithm [89, 90]. The first method based on residual differences (RD) is similar to online LOLIMOT, except instead of bisecting background submodels, any number of evenly spaced submodels can be maintained in each current cell. A Chow test is then performed using the RSS from the parent cell and each of the two candidate submodels (left and right) to determine if the submodels offer a statistically significant improvement in error. The second approach is based on residual analysis (RA) and builds on the approaches taken in SUPPORT and GUIDE by performing t-tests on the means and variances of the positive and negative residuals, which are recursively updated with new data. It was shown that the RD method performed significantly better than the RA method for the applications presented.

In Refs. [91], Hametner describes a batch axes-oblique approach for growing a hierarchical logistic discriminant tree using the EM algorithm. In Ref. [92, 93], the split logic is adapted to be compatible for an online, evolving LMN by assuming that the residuals follow a χ^2 distribution for Gaussian white noise given a proper model structure, and splitting the model if the hypothesis test on the residual variance fails under a pre-specified confidence. With its use of logistic sigmoid weighting functions and a combined nonlinear least squares for both child models, this work deviates from the other online LMN approaches.

2.5.3 Aerodynamic modeling LMN construction algorithms

Although the LMN approach to nonlinear modeling has been used across a variety of disciplines, it has only had limited use within aircraft system identification. Because the focus of this research is on aerodynamic modeling, two LMN-related approaches that have been specifically applied to this problem will be introduced.

In Ref. [52], Seher-Weiss applies a "classical" LMN algorithm to X-31 aerodynamic data, which performs similarly to LOLIMOT with bisecting axes-orthogonal splits and Gaussian validity functions. This work is then built upon by introducing two modified approaches known as extended LMNs, under the assumption that classical LMNs allow only local linear models with EVs = PVs. In the nonlinear LMN approach, the PVs are specified separately from EVs, and nonlinear coupled terms are allowed in the local models as combinations of the EVs and PVs. The structured LMN approach offers a unique parameter-centric alternative LMN architecture for a case where the model parameters have important physical meaning. For the parametric model structure, a separate LMN is developed for each parameter, the output of each LMN is multiplied by the corresponding modeling term, and the collection of structured LMNs is summed to produce the global model output. On the one hand, this approach provides vast flexibility such that each parameter can be modeled individually, and splits made along PVs are localized only for the parameter associated with that LMN. However, the cost functions across the LMNs are interdependent, and so a change in one LMN causes the cost functions in the other LMNs to change. Thus, in contrast to LOLIMOT's more efficient algorithm

where only the worst LMN is modified during each iteration, in this case the entire LMN must be updated during each iteration following an adjustment in a single cell.

The second approach by Brandon and Morelli in Ref. [12] served as a foundational motivation for the work developed in this dissertation. It is presented as a Takagi-Sugeno fuzzy modeling method where the cell structure is iteratively developed through a search cycle. Although it was developed for real-time operation and was tested onboard an MB-326M Impala aircraft, the cell structure must be developed offline in an iterative search cycle using training data, and once the cell structure is frozen, the local model parameters can be recursively updated in real time. In this work, EVs = PVs, and ramp-based membership (validity) functions were used to partition the input space. In contrast to many other LMN-related work where the poorest performing local model is further partitioned with an additional (often bisecting) split during each iteration, in this approach the entire LMN is reconstructed each time. A single evenly spaced partition is added to each of the input variables one at a time, and the entire LMN is re-estimated with this new cell structure using global optimization where all of the local model parameters are estimated together. Because the total number of cells is the product of the number of partitions for each input variable, more than one additional cell can be added during each iteration. The best candidate model is chosen by comparing model fit criteria such as coefficient of determination (R^2). Several heuristic stopping criteria are possible, including comparing model fit metrics from subsequent iterations to ensure sufficient improvement.

2.6 Summary

The particular modeling goals that this work set forth to accomplish are discussed in Chapter 1, which led toward utilizing the LMN approach. Since the field of LMNs and related work is so vast, this chapter was devoted toward further exploring some of the properties of these approaches and related algorithms that apply the methods. The cell structure can be developed in various ways including customizing the split dimensions and locations; the leaf models can be adjusted to fit the data in a particular cell through choosing the model variables, order, and structure; and the global model architecture can reflect how the local models best fit together to form the global model output.

Many offline and online methods employ the approach of candidate sub-models where before a split is performed, several possibilities are considered, and the best one is chosen from among the options based on a cost function. In this approach, the resulting cell structure is guaranteed to perform better than the previous one through the metric of that cost function. For online computation, however, candidate models can be inefficient and require high computational and memory loads. Other methods have explored the use of residual analysis to inform the splits, but these methods tend to show inferior results compared to candidate models, and have not been explored in depth for real-time applications, particularly for aerodynamic modeling. The study performed throughout this chapter, both regarding the background material as well as the literature review, led toward developing the new approach discussed in Chapter 3.

Chapter 3

Smoothed Partitioning with LocalIzed Trees in Real time (SPLITR)

3.1 Introduction

A novel approach to real-time global nonlinear modeling, known as Smoothed Partitioning with LocalIzed Trees in Real time (SPLITR), is developed and presented in this chapter. As discussed in Chapter 1, this novel LMN method can be applied to other systems outside of aircraft; however, the attributes of aerodynamic modeling strongly influenced the approach itself within the context of the NASA L2F concept, and so it will be described within the scope of aerodynamics. Some of the specified modeling goals and priorities include an interpretable model that offers physical insight, both local and global prediction capabilities, and compatibility with real-time updates of both the cell structure and parameters.

Although the L2F concept generally assumes no *a priori* modeling information is available, it is more typical that some information is obtained from first principles,

wind tunnel tests, CFD, prior flight testing, or other sources. Therefore, it is important that the modeling approach be compatible with incorporating a range of depth of prior insight. As discussed in Chapter 2, LMNs offer great flexibility through various features that can be specified in advance or customized automatically, and some of these properties will be discussed in the context of the SPLITR approach and aerodynamic modeling throughout this chapter.

The SPLITR approach overview is shown in Fig. 3.1. The experiment design is an important step where the goal is to efficiently collect test data that include high information content within the bandwidth of the dynamics that are modeled. For the flight test data in this work, the experiment design methods developed under the NASA L2F concept and discussed in Section 1.2.2 were employed. The test data are then used to recursively update the associated local model to which they belong. The information about the model performance in each cell is used to inform the cell structure determination, which will partition the model to a finer resolution where the system is highly nonlinear. Finally, the global weighting ties the cells together into the local model network.

Figure 3.1: Overview of SPLITR modeling approach.

The SPLITR approach is divided into three main sections in this chapter to address aspects associated with the leaf models, which include the recursive param-

eter estimation process; the cell structure determination, which consists of the cell splitting logic; and the global weighting. The chapter ends with a detailed summary of the various user inputs and specifications associated with the SPLITR algorithm, along with recommendations and insights for how to designate them.

SPLITR software developed through this work used parts of the software toolbox called System IDentification Programs for AirCraft, or SIDPAC [21].

3.2 Properties and aspects of leaf models

Within each cell, a local model is estimated for each response variable y that represents the data contained within the cell boundaries defined in terms of the PVs. Most generally, the modeling data can be obtained from any singular data source, or the data can be calculated analytically and/or processed based on several sources of data for any variable to be modeled. The response variable is generally expressed as

$$y = f(\boldsymbol{x}, \boldsymbol{\phi}) \tag{3.1}$$

where $\boldsymbol{x}$ are the explanatory variables (EVs) used in each local model, and $\boldsymbol{\phi}$ are the partitioning variables (PVs) that partition the input space into local regions.

In this work, the estimated model is a linear expansion in terms of explanatory variables that are measured. Other model forms can also be used, such as general polynomials, state space representations, or transfer functions, among others.

3.2.1 Selection of model explanatory and partitioning variables

In the SPLITR approach, the EVs can be specified separately from the PVs to include all, some, or none of the same variables. Each response variable is modeled separately and independently, so the EVs and PVs can be specified for each model based on any *a priori* insight. This flexibility can allow experience to inform the modeling process. In this work, the relevant terms through which to partition the model are automatically chosen from a specified pool of possible PVs. However, in the interest of limiting computational requirements for a large number of options, this pool should be specified carefully and sparingly.

Particularly for conventional fixed-wing aerodynamic modeling, the EVs may include air flow angles α, β, body-axes angular rates p, q, r, and control surface deflections δ. The PVs are typically the variables through which an aerodynamic database lookup table might be scheduled, such as Mach number, power setting, or nominal angle of attack. For rotorcraft modeling, the wind and stability axes are not defined in hover, so EVs α and β are typically replaced with body-axis translational velocities u, v, w, and PVs may include V_0 or $\bar{q}$ as well.

In this work, for simplicity and interpretation of the simulation examples, global EV values were used in the local models, so the EV data were not defined relative to the reference point at the center of each cell.

For each test case shown throughout this work, the PV input space was normalized for validity function evaluation, so the maximum and minimum expected

values for each PV were specified in advance. If the range is specified as too small, the validity function weights will become saturated at the edges. If the range is too large, then there may not be enough resolution for the validity functions to adequately weight the range of actual data within the PV input space.

3.2.2 Specification of local model order

LMNs emphasize the idea that an arbitrary nonlinear function can be represented by a number of smaller linear components, so the examples discussed in this dissertation are restricted to linear modeling terms. The parameter estimation regression methods presented though are in no way restricted to linear terms.

For aerodynamic modeling from flight data in particular, the traditional approach involves linear approximations at many relevant operating points throughout the flight envelope. Other approaches utilize global multivariate nonlinear polynomials by selecting the relevant modeling terms through methods such as stepwise regression or multivariate orthogonal functions. These two sets of approaches are different methods to account for nonlinearities, i.e. by utilizing many linear models, or employing a single global multivariate nonlinear model. LMNs can take advantage of the fact that traditional aerodynamics tend to be linear throughout much of the flight envelope by collecting a set of local linear models for the cells. However, highly nonlinear stall, post-stall, or spin regimes may be captured better with a single higher order polynomial model instead of many linear models.

3.2.3 Determination of model structure

In this work, a local linear model structure is assumed, and the EVs are fully specified *a priori*, so model structure determination is not incorporated into the automated modeling process. If higher order model terms are allowed, then a model structure determination method would be necessary. Even with strictly linear terms, model structure determination could be an important step to choose the relevant EVs from a broader pool. However, LMNs already face the challenge of adapting the cell structure in real time, so adapting the modeling terms as well is an additional difficulty for onboard real-time computation. For this reason, the local model structure is assumed linear, fixed, and uniform for all local models.

3.2.4 Parameter estimation process

The parameters for each model are estimated by minimizing the equation error in the least-squares sense, with this process described in detail in Refs. [21, 22]. The regression equation can be expressed as

$$\boldsymbol{z} = \boldsymbol{X}\boldsymbol{\theta} + \boldsymbol{\nu} \tag{3.2}$$

where $\boldsymbol{z} \in \mathbb{R}^N$ is the response variable data to be modeled over N measurements, $\boldsymbol{X}$ is the $N \times n_x$ matrix of explanatory variables, $\boldsymbol{\theta} \in \mathbb{R}^{n_x}$ is the vector of model parameters, and $\boldsymbol{\nu} \in \mathbb{R}^N$ is the modeling error.

The cost function to solve for the best estimate $\hat{\boldsymbol{\theta}}$ that minimizes the sum of

squared differences between the measured response variable and the model output is

$$J(\boldsymbol{\theta}) = \frac{1}{2}(\boldsymbol{z} - \boldsymbol{X}\boldsymbol{\theta})^T(\boldsymbol{z} - \boldsymbol{X}\boldsymbol{\theta}) \tag{3.3}$$

The solution that provides the optimal result is

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{z} = \boldsymbol{D}\boldsymbol{X}^T\boldsymbol{z} = \boldsymbol{M}^{-1}\boldsymbol{X}^T\boldsymbol{z} \tag{3.4}$$

where $\boldsymbol{M} = \boldsymbol{X}^T\boldsymbol{X}$ contains the information content in the regressor data, and $\boldsymbol{D} = \boldsymbol{M}^{-1}$ is the dispersion matrix.

The Cramer-Rao lower bound, which is an asymptotic estimate of the parameter covariance matrix, is a measure of uncertainty in the parameter estimates. If the residuals are assumed to be white, it is defined as [21]

$$\boldsymbol{\Sigma}(\hat{\boldsymbol{\theta}}) = E\left[(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta})(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta})^T\right] = \hat{\sigma}^2(\boldsymbol{X}^T\boldsymbol{X})^{-1} \tag{3.5}$$

The model fit error variance is then approximated as

$$\hat{\sigma}^2 = \frac{(\boldsymbol{z} - \hat{\boldsymbol{y}})^T(\boldsymbol{z} - \hat{\boldsymbol{y}})}{N - n_x} \tag{3.6}$$

While the formulation defined above can be used to estimate a model with batch data, $\hat{\boldsymbol{\theta}}$ can also be updated recursively for real-time parameter estimation. If the model is defined at the $(i-1)^{th}$ data point, then as new measurements $z(i)$ and $\boldsymbol{x}(i)$ are received, the parameter estimates can be updated through Eqs. (3.7

–3.9) [21].

$$K(i) = \frac{\boldsymbol{D}(i-1)\boldsymbol{x}(i)}{\lambda_{ff} + \boldsymbol{x}^T(i)\boldsymbol{D}(i-1)\boldsymbol{x}(i)} \tag{3.7}$$

$$\boldsymbol{D}(i) = (1/\lambda_{ff})[\mathbb{I} - \boldsymbol{K}(i)\boldsymbol{x}^T(i)]\boldsymbol{D}(i-1) \tag{3.8}$$

$$\hat{\boldsymbol{\theta}}(i) = \hat{\boldsymbol{\theta}}(i-1) + \boldsymbol{K}(i)[\boldsymbol{z}(i) - \boldsymbol{x}^T(i)\hat{\boldsymbol{\theta}}(i-1)] \tag{3.9}$$

The forgetting factor λ_{ff} is included to weight past information to allow the model to update more quickly with new information, which is particularly useful in this work to allow the cells to adjust more quickly to new information following splits. The parameter covariance is updated with a recursive estimate of the fit error variance as

$$\hat{\sigma}^2(i) = \left(\frac{i-1}{i}\right)\hat{\sigma}^2(i-1) + \frac{1}{i}\nu^2(i) \tag{3.10}$$

$$\boldsymbol{\Sigma}[\hat{\boldsymbol{\theta}}(i)] = \hat{\sigma}^2(i)\boldsymbol{D}(i) \tag{3.11}$$

After the modeling process has been completed, the effectiveness of the resulting model can be described by several modeling metrics. In particular, the coefficient of determination (R^2), defined in Eq. (3.12), is a model fit quality measure that varies from 0 to 1 and describes how much of the variation in the data

about the mean value is captured by the model.

$$R^2 = 1 - \frac{\sum_{i=1}^{N}[z(i) - \hat{y}(i)]^2}{\sum_{i=1}^{N}[z(i) - \bar{z}]^2} \tag{3.12}$$

The equation-error formulation enables recursive parameter updates as data are obtained in real time, and it is performed in the time domain to allow points that are not contiguous in time to be incorporated into the associated models in each cell. For example, Fig. 3.2 shows F-16 simulation data that are divided into three cells across the complete range of α, where the data associated with the α range in each cell are incorporated into the respective local models of low-, mid-, and high-α ranges.

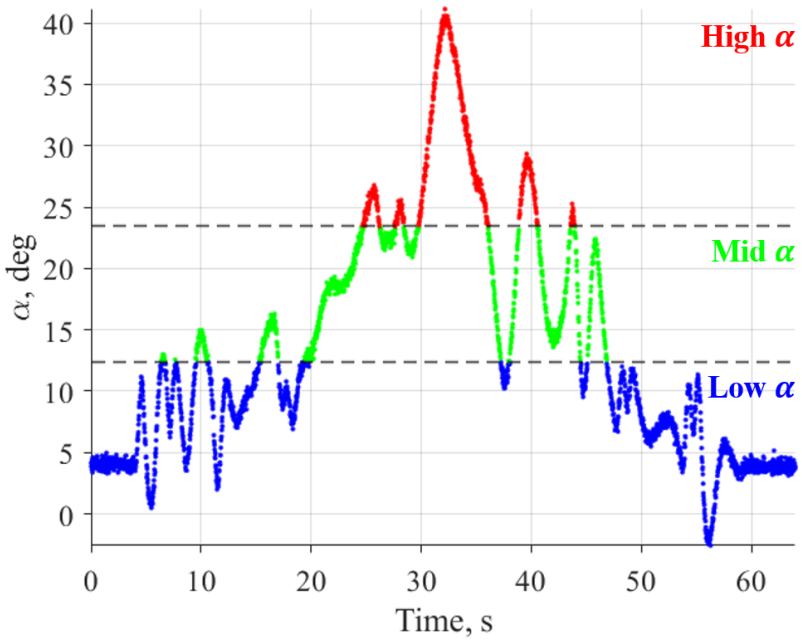


Figure 3.2: F-16 simulation time history data with α partitioning.

The parameter estimation can be performed using either a global or local optimization process, as discussed in detail in Section 2.3.4, and which will be sum-

marized here. Prior work employs global learning where the parameters across all of the cells are optimized simultaneously in a single global weighted regression problem [12]. This centralized approach can lead to an ill-conditioned regression problem and interdependent parameter estimates with high uncertainties that cannot be interpreted individually [23]. Other work utilizes local learning by posing a separate weighted regression problem for each local model [52]. However, the individual cell weights are a normalized function of the weights in all of the cells. So when a new measurement or a new cell is added, the estimates from all other cells would change, meaning the parameter estimates from all cells are still interdependent. In this work, the local parameter estimation problem is completely decoupled from the global modeling by first estimating the parameters in each cell, and then overlaying the validity functions afterwards. This approach retains the interpretability of the model in each cell, and allows new measurements to simply be incorporated into the model to which they belong, irrespective of the other cells. It also offers flexibility for validity function customization to blend the local models together.

3.3 Cell structure determination process

The cell structure is developed through a decision-making process that is used to successively partition the data into relevant local regions. The guiding objective is to improve the global model accuracy by accounting for nonlinearities, while maintaining a parsimonious model with as few cells as possible.

3.3.1 Cell structure problem statement

This work constrains the splits to be axes-orthogonal because of the real-time operation constraint, emphasis on interpretability of the cell structure, and because the PV pool for aerodynamic modeling is typically low-dimensional, with some of the relevant PVs discussed in Section 3.2.1. The common method of bi-secting axes-orthogonal splits, however, can be highly restricting, in particular if the nonlinearities are biased toward one side of the PV input space as is typical for aerodynamics, and which can result in redundant cells and overly complex models with less meaningful split locations. The SPLITR approach more generally allows variable breakpoints that are informed by a new method of residual analysis, which forgoes the advantages of batch processing and iterative optimization methods for real-time compatibility.

Accordingly, the cell structure determination problem can be summarized by posing the following questions:

Is a split needed? Given the current cell structure, the model is evaluated to determine if it is sufficient, or if the data in one or more cells are not characterized well by the local models.

Which cell to split? Which local model is performing poorly? Is it due to noisy data, or is it due to a nonlinearity that this local model is not sufficiently capturing? The approach needs to keep track of estimates of noise in the data, as well as model fit statistics.

Which dimension to split? Once a particular cell has been identified as having a poor model due to unmodeled behavior, then allowable axes-orthogonal split dimensions include those identified as PVs. If the relevant possible sources of nonlinearity are known in advance, this pool can be more carefully specified to reduce computation.

Where to split along a dimension? Where along a certain dimension in a specified cell should the next breakpoint be placed? If the developed cell structure is intended to offer physical insight through the partition locations, then these breakpoints must be carefully selected.

When to split? How much data or statistical evidence is needed to split the cell? If new cells are added too rapidly, the result can be an overly complex model with redundant cells, whereas if splits are performed too cautiously, then the model is degraded in real time.

3.3.2 Preliminary guiding aspects and overview of cell structure determination

Many past methods use an iterative, offline cell structure determination process which is driven by improving a global model fit performance metric. During each iteration, a binary axes-orthogonal split is typically proposed in all existing cells with all possible dimensions, and the entire model is recomputed for each case, resulting in a series of proposed models that all have one more cell than the previous iteration. The proposed cell structure that improves the global performance metric the most is chosen for the next model iteration [52, 71, 79]. With the advantage

of being offline, this iterative approach has access to all of the modeling data at once, and estimates a new set of parameters for each proposed model. This is not the situation for a real-time approach, where all past data cannot be practically stored and used to iteratively recompute parameter estimates after a new cell is added. Therefore, a different real-time decision-making procedure for cell structure determination is necessary.

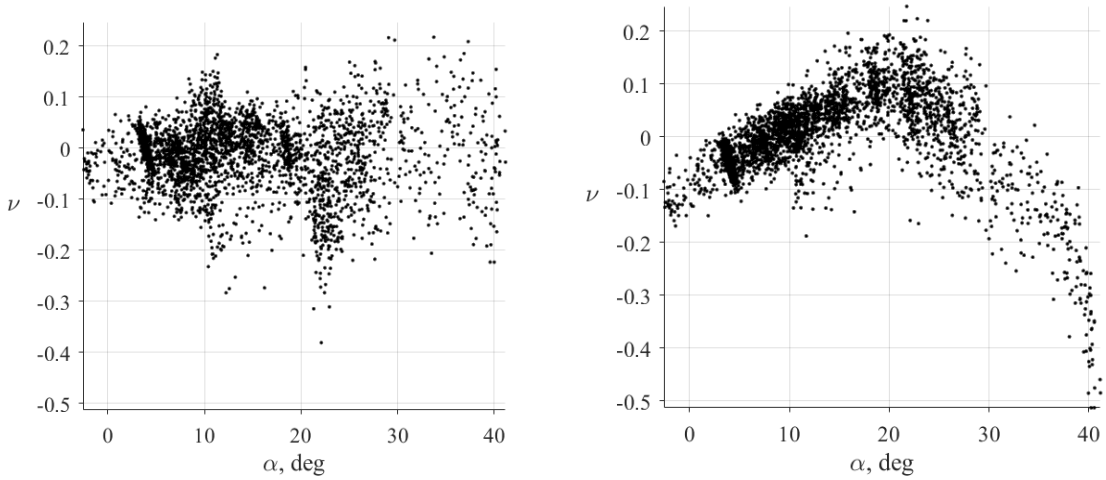
A challenge with the offline approaches, and one that is apparent in any system identification problem, is choosing an ideal metric that adequately characterizes the model fit quality, and serves as a reliable stopping criterion to preclude excessive splits and ensure a parsimonious model. Common model fit statistics such as the coefficient of determination (R^2), Akaike information criterion (AIC), residual sum of squares (RSS), etc., or a combination thereof, are used to characterize the model fit during the model development process for these approaches. But while they can be useful tools for model assessment, they are global measures of the fit, and can provide misleading information about the localized model fit quality if not interpreted correctly. Furthermore, a satisfactory threshold can be arbitrary across different data sets with varied data quality and noise levels.

The SPLITR approach to cell structure determination, similar to that of parameter estimation, examines the performance of the local models instead of the global model fit. This localized approach infers that if the local models perform well, then the weighted combination that comprises the global model will provide a good fit too. Instead of considering model fit statistics, SPLITR examines the essence of the information contained in these metrics, which are the residuals, i.e. the dif-

ferences between the measured data and the estimated model output, as shown in Eq. (3.13) for the i^{th} measurement.

$$\hat{\nu}(i) = z(i) - \hat{y}(i) \tag{3.13}$$

The residuals represent samples of the modeling error, and they can be used to assess model adequacy and reveal model deficiencies through a residual characterization procedure, which will be described in the data processing sequence next. Furthermore, the individual local model residuals are discrete samples that describe the errors locally, thus providing information regarding the magnitude of the model error as well as the location within the ranges of measurements. Ideally, the residuals should have constant variance, and be mutually uncorrelated. In flight data, the residual variance magnitude is often a function of certain flight variables, but should still not display deterministic character. Figure 3.3(a) shows the residuals as a function of α from a SPLITR model using the F-16 data set. Notice that the variance tends to increase at higher α, which is consistent with the larger amplitude of noise in that region. Figure 3.3(b), on the other hand, shows residuals with a deterministic character that indicates poorly modeled aerodynamics, and which results from a model with only a single linear cell for the same F-16 data. Ideally, the whiteness of the residuals in each cell would be evaluated through the autocorrelation, but since the data are not necessarily contiguous in time in each cell, this method is not feasible, and another approach must be formulated.



(a) SPLITR nonlinear model

(b) Linear model

Figure 3.3: Residuals for F-16 models of $C_L = f(\alpha)$.

The SPLITR process can operate under two separate update rates, with the process overview shown in Fig. 3.4. At a faster rate, each new measurement is processed immediately so that all of the useful information it contains is extracted, and the model parameters of the associated cell are recursively updated. Since the cell structure is not expected to change rapidly, the cell splitting procedure operates at a slower rate, which is chosen to be outside of the expected range of dynamics that are being modeled, so that the cell structure may be modified at least as fast as the dynamics. The global model blending, which is a function of the cell structure, is updated along with the cell structure changes. These update rates can be modified for other systems and data rates. A detailed depiction of the data processing and cell splitting procedures is presented in Fig. 3.5 and will be descried in detail in the next section.

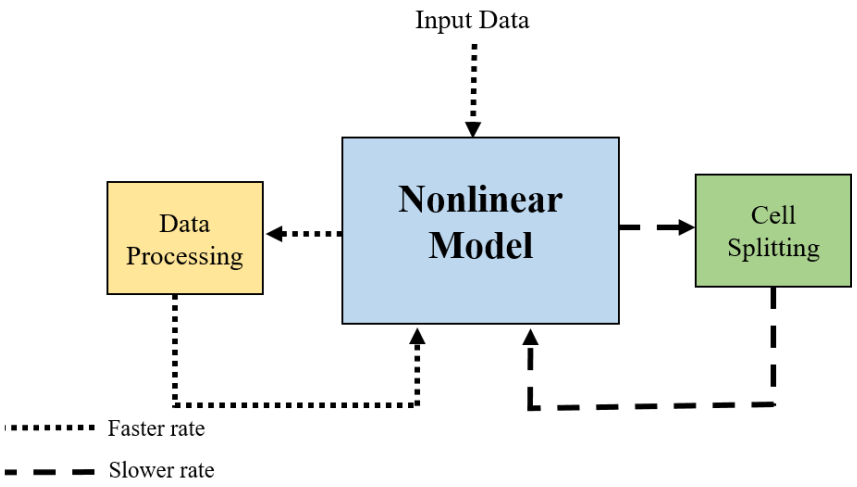
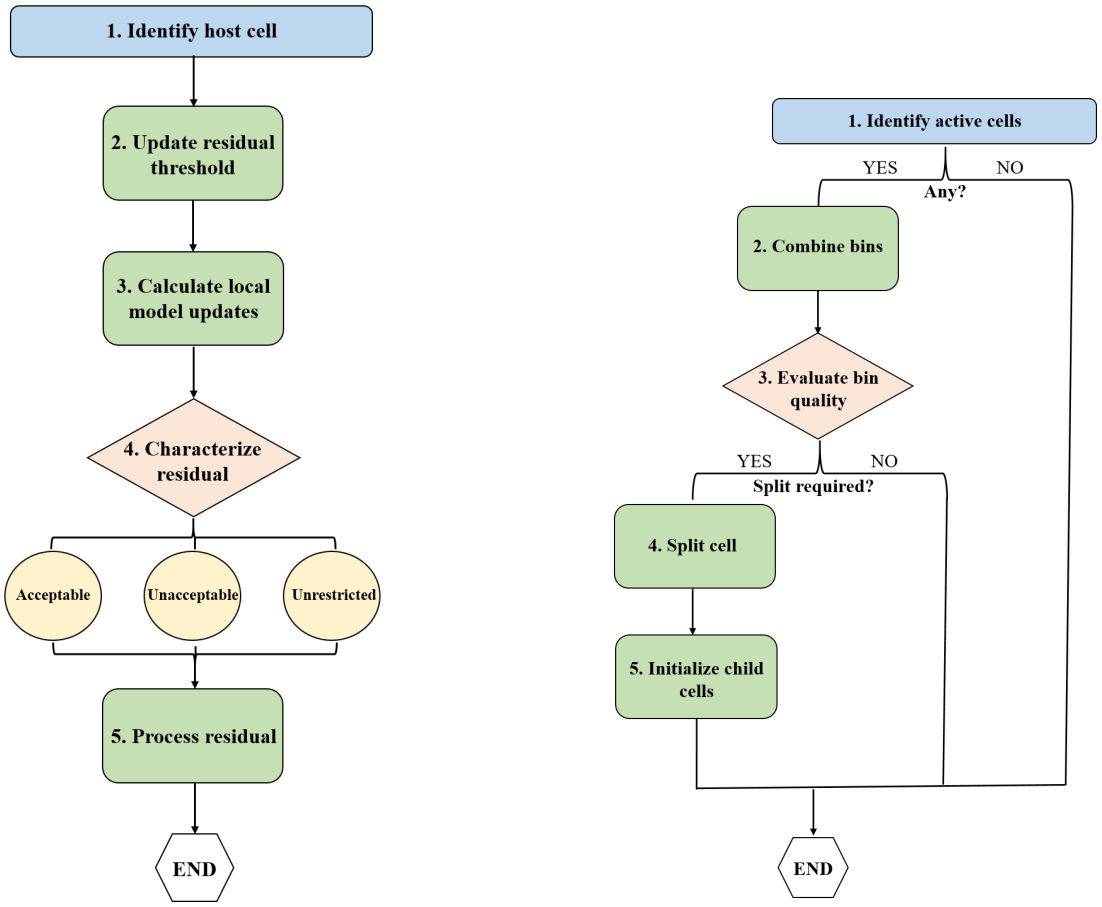


Figure 3.4: SPLITR process overview.



(a) Data processing procedure

(b) Cell splitting procedure

Figure 3.5: Data processing and cell splitting procedures for SPLITR algorithm.

3.3.3 Data processing procedure

The data processing sequence is shown in Fig. 3.5(a), and each step will be summarized below. The i^{th} measurement supplied to the model consists of the response variable $z(i)$, the associated vector of explanatory variables $\boldsymbol{x}(i)$, and the vector of specified partitioning variables $\boldsymbol{\phi}(i)$.

1. *Identify host cell:* For a given cell structure, each response variable measurement is parametrized by its associated PVs and belongs to a single cell.

2. *Update residual threshold:* Residual characterization is the method used to distinguish between random and deterministic properties in the residuals, the latter of which would be attributed to unmodeled dynamics, or nonlinearities in the data that are not captured by the local model. It is important not to mistake noisy data for a poor model fit. Therefore, an acceptable threshold is needed to characterize the residuals such that those that lie within the bound represent reasonable model error that includes noise influences, and those that lie outside represent deterministic model deficiency. To compute the threshold, each response variable to be modeled is passed through a real-time discrete high-pass filter to obtain an estimate of the noise. Note that this estimate is a wide-sense classification of noise that includes any other content that will not be modeled [12]. A running root-mean-square (RMS) estimate of the noise is computed for the response variable data in each cell, and it is multiplied by a factor λ_ν, to account for statistical variations and to

describe an acceptable residual threshold based on the noise level. These noise estimates are binned across the range of each PV, as will be detailed later in this section, so different cells across the ranges of PVs would be expected to have varied thresholds based on the noise levels in the data they contain.

3. *Calculate local model updates:* The parameter estimation scheme discussed in Section 3.2.4 is used to recursively calculate the updated local cell parameters with the information contained in each new measurement. The local model output for the i^{th} measurement incorporated into the k^{th} cell is

$$\hat{y}_k(i) = \boldsymbol{x}(i)\hat{\boldsymbol{\theta}}_k(i) \tag{3.14}$$

The associated local residual for that point is

$$\hat{\nu}_k(i) = z(i) - \hat{y}_k(i) \tag{3.15}$$

4. *Characterize residual:* Using the calculated residual threshold, the absolute value of each residual is characterized in one of three ways. An unrestricted residual is from a measurement that is unconditionally allowed into the model. This can be at the beginning of the modeling process as it waits to receive enough information for a reliable initial model, or as will be discussed later in this section, immediately following a split so the child cell models can adjust.

A measurement associated with an unrestricted residual is consequently not regarded for splitting purposes. An acceptable residual is one that is not considered an unrestricted residual, that is below the calculated residual threshold for that cell, and is regarded as supported by the local model accounting for noise. An unacceptable residual is one that is also not characterized as unrestricted, that is outside the residual threshold, and is therefore considered an indication of a poor model fit in the associated cell.

5. *Process residual:* To provide discretized error information within each cell that will inform a possible split location, the absolute values of the residuals are binned across the range of each PV based on a prescribed resolution that is the minimum acceptable width of a cell. The bins are designated at the beginning of the modeling process, and as the cell structure is modified, the cell to which they belong changes accordingly with the cell boundaries. Determining the split location is thereby reduced from a continuous problem across the range of a cell to a discrete choice along one of the pre-defined bin boundaries, which is more conducive to real-time operation and decision making.

   In real-time operation, the residuals in each bin must be processed in a way that their influence toward a possible split can be retained and recursively updated. The properties of the absolute values of the residuals in each bin are therefore represented by the mean and variance, which are given for N data points as

$$\mu_N = \frac{|\nu_1| + |\nu_2| + ... + |\nu_N|}{N} = \frac{\sum_{i=1}^{N} |\nu_i|}{N}, \quad \sigma_N^2 = \frac{\sum_{i=1}^{N} (|\nu_i| - \mu_N)^2}{N} \tag{3.16}$$

As new measurements are received, the mean and variance calculations of the associated bins are recursively updated as

$$\mu_{N+1} = \frac{\mu_N N + |\nu_{N+1}|}{N+1}, \quad \sigma_{N+1}^2 = \frac{\sigma_N^2 N}{N+1} + \frac{(|\nu_{N+1}| - \mu_{N+1})^2}{N} \tag{3.17}$$

Each bin contains a count of the different types of residuals, and a set of four recursively updated associated statistics: the mean and standard deviation of the acceptable residuals, μ^A, σ^A, and those of both the acceptable and unacceptable residuals together, μ^B, σ^B. Both types of residuals are combined in μ^B, σ^B as a measure of the total mean and standard deviation in the residuals. Note that the unrestricted residuals are not incorporated into the bin statistics.

Since the parameter estimates in each cell are changing over time as new data are obtained, the real-time residuals that are calculated and processed here are obsolete as soon as the parameters are next updated. Theoretically, if all of the past data were saved then the residuals could be recomputed each time the parameters are updated. However, this is considered an artifact of the constraints of real-time operation.

If the residual is characterized as unrestricted or acceptable, the parameter

updates computed in step 3 above are accepted, and the local model is updated accordingly. If the residual is considered unacceptable, then the local model is returned to its prior state, and the associated measurement is stored. Note that although some data are subsequently stored and later accessed, this amount of data is small compared to the total amount of data, and is considered reasonable for onboard computation. Alternatively, this data storage step could be skipped, which would effectively discard the measurements associated with the unacceptable residuals and rely solely on future data, as discussed in step 5 of the cell splitting sequence below.

3.3.4 Cell splitting procedure

The cell splitting procedure is shown in Fig. 3.5(b), and is described next. During this sequence, the residuals are evaluated to determine if the cell structure is adequate, or if a cell should be split. This procedure is detailed below.

1. *Identify active cells:* Because the cell structure determination is directly associated with residuals characterized as unacceptable, only those cells that have received measurements whose residuals are characterized as such since the last cell structure check need to be examined. Additionally, only the active bins within the cell, or the bins that actually contain data, are analyzed.

2. *Combine bins:* A user-defined minimum cell width may be conservatively small in order to capture higher order nonlinearity in certain ranges. However, a fine resolution may not be necessary if data are obtained quickly across a wide

range, and larger cell widths may therefore be considered first. Furthermore, with such a fine initial resolution, it might take a while for each bin to collect enough information to produce useful, informative bin statistics. Therefore, a maximum number of bins across each cell is also pre-specified, so that early on in the cell division process only larger splits will occur, and later on, the finer resolution can be considered.

This can be visualized through an example in Fig. 3.6 which shows a single cell with 5 bins across the range of the PV where the residual information is stored. The cell boundaries are shown as thick red lines. There are 10 minimum-width bins spaced throughout the cell and shown as gray dashed lines. If a maximum number of cell bins is specified as 5, these bins are combined as shown by the bin boundaries outlined in blue.

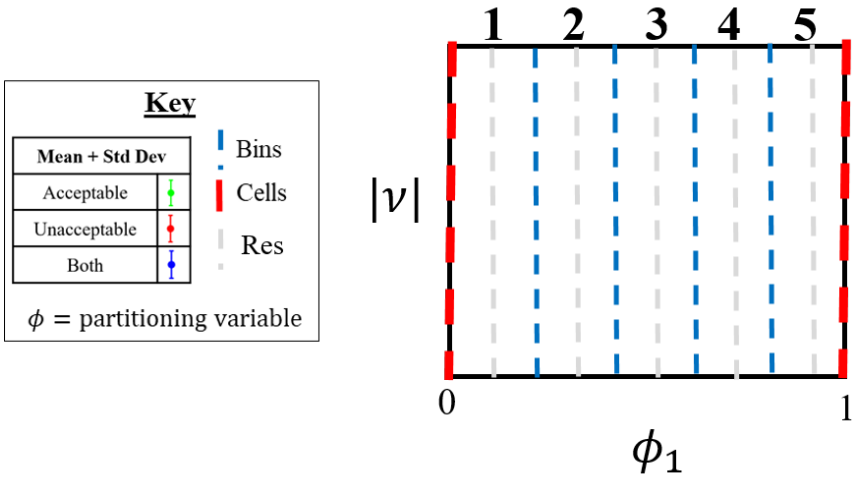


Figure 3.6: Sample cell with 5 bins.

The minimum bin width cannot be changed in real time since the associ-

ated past residuals are not accessible in order to compute new bin statistics. Therefore, from the beginning the statistics are computed in the minimum bin widths, and then those statistics from adjacent bins are combined during the cell structure checks up to the maximum number of bins. Equations (3.18 – 3.19) show how the means and variances, respectively, are combined for two bins using their individual values as well as the number of data points in each, designated as N_1 and N_2.

$$\mu_{N_1+N_2} = \frac{\mu_{N_1} N_1 + \mu_{N_2} N_2}{N_1 + N_2} \tag{3.18}$$

$$\sigma^2_{N_1+N_2} = \frac{\sigma^2_{N_1} N_1 + \sigma^2_{N_2} N_2}{N_1 + N_2} + \frac{\mu^2_{N_1} N_1 + \mu^2_{N_2} N_2 - (N_1 + N_2)\mu^2_{N_1+N_2}}{N_1 + N_2} \tag{3.19}$$

3. *Evaluate bin quality:* A pass/fail status of each (combined) bin is then determined to describe if the bin contains an acceptable amount of error, or if the indication of deterministic error is strong. If the mean of both the acceptable and unacceptable residuals is pulled beyond a specified number λ_σ, of standard deviations from the mean of the acceptable residuals, as shown in Eq. (3.20), then the bin is considered failed. This can be visualized through a depiction in Fig. 3.7.

$$\mu^B > \mu^A + \lambda_\sigma \sigma^A \tag{3.20}$$

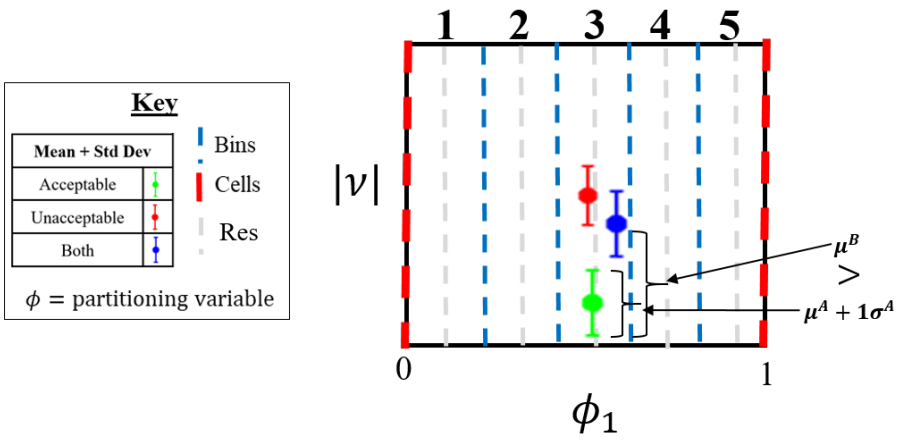


Figure 3.7: A bin fails if the mean of both types of residuals is pulled beyond a specified number of standard deviations from the mean of acceptable residuals.

A bin may contain both acceptable and unacceptable residuals, so Eq. (3.20) ensures not only that there are unacceptable residuals in the bin, but that their magnitude "outweighs" the acceptable residuals found there, according to this specified comparative metric. Furthermore, for a bin to fail the status check, it must contain a minimum number of residuals to ensure that sufficient information is obtained. If a bin fails the status check, the normalized severity of the unacceptable residuals is computed as in Eq. (3.21) to represent how many standard deviations the mean of both types of residuals is drawn away from that of the acceptable residuals, and is also saturated at 1.

$$\text{bin severity} = \frac{\mu^B - \mu^A}{\sigma^A \lambda_{\sigma,norm}} \tag{3.21}$$

The bin severities from adjacent failed bins are summed to obtain a group severity, which is then compared to a prescribed maximum total severity. If

it surpasses the limit, then that bin group is a candidate to inform the next cell split. Within a single cell, the group of bins with the largest total severity beyond the limit is chosen to inform the split. The above analysis can be performed in parallel across multiple PVs to choose both the split dimension and location along that dimension in a single cell through the same decision process. However, note that the discretized binning and calculations will need to be performed across all PV dimensions (n_ϕ times), so it is advisable to limit the number of partitioning options within the understanding of the system behavior and possible sources of nonlinearity.

4. *Split cell:* If a split has been justified, then using the selected set of bins, two split locations are considered. The cell may be split at the right boundary of the right-most bin or at the left boundary of the left-most bin. The chosen split location is that which partitions the failed bins toward the side with the smallest distance to the edge of the *active* PV input space, which is often the side with the fewest additional bins. This is in order to leave intact the largest region that may already be performing well. In the depiction in Fig. 3.8, the left boundary of bin 3 would be chosen.

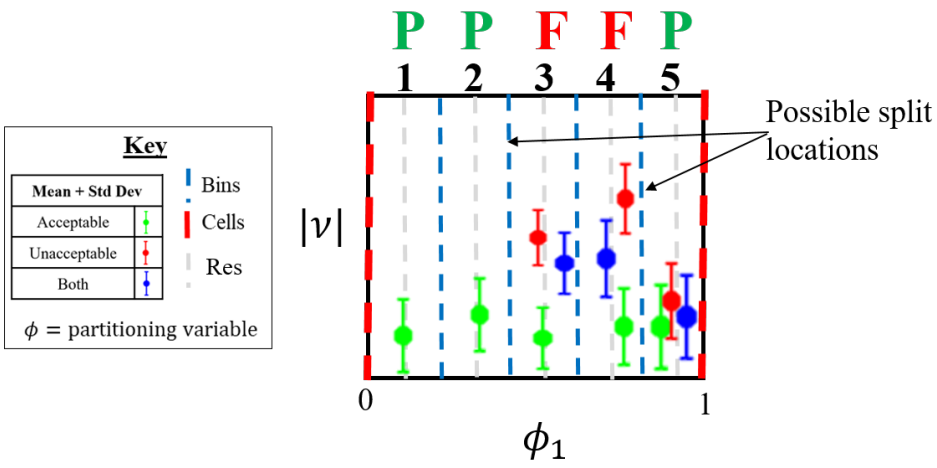


Figure 3.8: The split location is chosen as the right or left boundary of the collection of failed bins.

The parent cell is then divided into two child cells, with one on each side of the specified split location. The bin counts and statistics discussed in step 5 of the data processing sequence are zeroed for the child cells.

5. *Initialize child cells:* When a parent cell is divided into two child cells, the presumption is that the parent model was inadequate at least across a certain range of the data it included. However, its parameters were estimated using data across the ranges of both child cells, so when a split is performed, the parameter estimates associated with the parent model no longer equally represent both sides of the split. Four possible approaches to initializing the child cell models immediately following a split are discussed.

   First, ideally if all past modeling data are stored and accessible, then when a cell is split, the parameter estimates for the child cells can be recomputed in batch with all of the past data that belong in each cell. This approach,

however, is not practical in real time.

Second, all possible configurations of cell structures can be posed *a priori*, and as the data are received in real time, all possible cell models will be recursively updated, while only the models in the current cell structure will be activated. Note that in this approach, each measurement can belong to more than one cell. Then when a split is performed, the respective latent models will be activated. While this approach would also provide a model for the exact data in each cell, this would only be practical with a very coarse resolution of possible splits and few split dimensions. If there are multiple split dimensions and a fine resolution, the curse of dimensionality implies not only a large number of minimum resolution cells, but also an astronomical number of intermediate cell structures as the model is successively partitioned. If there is prior knowledge of the system behavior that can inform a reasonable number of possible cell structures, then this approach may be more practical.

Third, an approach that would be feasible in real time is to zero the model in each of the child cells and begin from scratch to estimate a new model with the future data obtained in the cell. Again, this would provide a model with the correct data; however, it is inefficient to discard all of the past data collected in the region of the parent cell, particularly because much of it could still be relevant to the child models.

Finally, a practical approach for real-time model initialization accepts the fact that the parent model was built using data contained in both child cells,

and despite that, each child cell's model parameters are initialized with those of the parent cell. Since the information matrix cannot be recomputed in real time with past data, the child cell is initialized with a percentage of the parent cell's information matrix, $\boldsymbol{M} = \boldsymbol{X}^T\boldsymbol{X}$, that is proportional to the number of past measurements in the parent cell that are within its range. These measurement counts are binned similar to the residuals and are summed across the bins that belong to each child cell. Note that the inherent assumption behind this approach is that each measurement contributes equally to the information matrix in terms of information content, which is not necessarily true, but which provides a good starting guess within the real-time constraints. The dispersion matrix, $\boldsymbol{D} = \boldsymbol{M}^{-1}$, is then used to continue the recursive parameter updates for the child cell, as in Eqs. (3.7 –3.9). An additional cell initialization proportion factor may also be used to further reduce the reliance on parent model information if the child cells are expected to differ significantly from the parent models. Lastly, the stored measurements associated with the unacceptable residuals from each side of the split are incorporated into the respective child models so that they are adjusted instantaneously to include the data that were deemed not fit well by the parent model. Following the child cell initialization, a specified number of new unrestricted measurements are also automatically allowed into the model to allow it to adjust to the new region it is defined over, before residuals are further characterized. The child cell's residual threshold can be initialized as well, based on the RMS of the noise estimates of the response variable data recorded in the bins belonging to

the cell.

3.4 Global weighting and combination of local models

Up until this point in the SPLITR modeling technique, the parameter estimation and cell structure determination processes within the LMN architecture have been decoupled from the model weighting process, and instead have focused on the local cells and preserving their accuracy and interpretability. The global nonlinear model, which is the output of the LMN, is then a weighted combination of the piecewise local models multiplied by the validity functions. In this work, the validity functions were chosen as normalized Gaussian weighting functions that are scaled based on the cell width, as discussed in Section 2.4.1. The weighting function shown in Eq. (2.12), as well as the associated scaling factor, was used in this work. The global model output over M cells for the i^{th} point is then given in Eq. (3.22), where the global model output is expressed as a weighted combination of the local model outputs, as discussed in Section 2.4.2.

$$\hat{y}(i) = \sum_{k=1}^{M} \hat{y}_k(i) w_k(i) = \sum_{k=1}^{M} \boldsymbol{x}(i) \hat{\boldsymbol{\theta}}_k(i) w_k(i) \tag{3.22}$$

Note that since the validity function is not directly incorporated into the regression and is simply overlaid on the local models, the resulting weighted nonlinear model is no longer optimal in the least-squares sense. The role of the validity functions, however, is primarily to ensure smooth transitions over the cell boundaries without strongly impacting the parameter estimates in the individual cells.

3.5 SPLITR user specifications and inputs

Chapter 3 so far has described in detail the SPLITR algorithm, along with several user-specified parameters that can be customized for a particular application or data set. This is intended to allow the user to incorporate any physical or other insight to tailor the method to provide the best results. For systems with little to no prior insight, these parameters can still be specified more generally. Some of the basic user specifications include the EVs associated with each response variable, a PV pool to be considered for partitioning each response variable, and the expected minimum and maximum value for each PV for normalization. Consistent across many modeling approaches, it is important that any user insight into the system to be modeled is used to inform these and other specifications. This section will summarize the particular SPLITR input parameters and offer guidance for how to designate them. The relative importance and impact of each parameter will also be highlighted to point out those that impact the results more strongly.

A list of the user-specified parameters for the SPLITR algorithm is given in Table 3.1, along with the baseline specifications used throughout this work, where applicable, and each will be described below.

Model Complexity	
Minimum cell width	—
Maximum # of cell bins	10
Maximum # of allowable cells	Inf
Residual Threshold	
High-pass filter order	4
High-pass filter cutoff frequency	3 Hz
Residual threshold factor, λ_ν	2 or 4
Filter window	100 points
Bin Status	
Standard deviation factor, λ_σ	0.75
# of standard deviations to normalize, $\lambda_{\sigma,norm}$	1
Total bin severity threshold	2
Minimum # of points in bin for failure	20
Unrestricted Residuals	
# of unrestricted points before initial model	250
# of unrestricted points following a split	150
Update Rate	
Parameter update rate	50 Hz
Cell splitting update rate	5 Hz
Global Model	
Smoothness factor, λ_s	1
Model Information	
Forgetting factor, λ_{ff}	0.995
Cell initialization proportion	0.2 or 1

Table 3.1: SPLITR algorithm parameters.

The first set of parameters are related to controlling the model complexity by supplying the resolution required to partition properly, while not allowing the model to become too complex.

- **Minimum cell width:** This specifies the minimum width for the cells in each dimension, as well as informs the number of bins in which the residual information is recorded. These minimum-width bins can then be combined up until the maximum number of cell bins (see below). This parameter is specified individually for each of the PVs. The split locations can only occur along

the discretized boundaries of these minimum-width bins, so this specification strongly impacts the eventual partitions, but is also linked to the maximum number of cell bins, which collectively will determine the choices for split locations at a given time and range of PV data. In many cases, the smaller the minimum cell width, the greater the flexibility for high-resolution cells. Nonetheless, the cell widths may actually never be defined as quite so narrow if wider cells are sufficient. On the contrary, the higher the resolution, the more bins and more computational power and memory required to store and process the information. Additionally, if there is moderate to substantial noise in the data, a high resolution may be unnecessary if a better split location is hidden by noise. If the partitions are too small, there may also not be enough data points or a high enough SNR to capture sufficient modeling information. If the minimum cell width is too large, then the resolution may simply not exist to capture certain nonlinearities that lie in the data, and the local linear model assumption is stretched. Currently, the cell resolution is constant across a given PV, whereas future work could allow for a variable resolution to reflect expectations of nonlinearity. For example, a larger cell width could be specified for low-α aerodynamics which tend to follow linear trends, and a smaller width at high-α to allow the model to be split in accordance with physical expectations and insight.

- **Maximum # of cell bins:** This parameter specifies the maximum number of combined bins that are considered in each cell for splitting purposes at a

given point in time. Even if a relatively small minimum cell width is specified, it is possible that early on in the model development, wider splits will be desirable and/or preferable to allow the model to capture larger regions first before proceeding to capture smaller nonlinear regions. It will also increase the likelihood that there are sufficient data points contained in the larger region through which to inform the splits. The minimum-width bins are used to store the residual statistical information, and then the means and standard deviations of adjacent bins are combined for analysis purposes. The examples in this work used a maximum of 10 cell bins.

- **Maximum # of allowable cells:** This will specify the maximum number of cells that are allowed to partition a single model, across all PV dimensions. It is meant as a failsafe to ensure the data are not overfit, or to allow the user to specify a reasonable limit. Since the SPLITR model development in real time is not an iterative process, overfitting is less likely to occur, but can still be a concern if e.g. the residual threshold factor is too low. An upper limit does not need to be specified.

The next set of specifications describes how the residual threshold is computed for each cell using an estimate of the noise content in the data.

- **High-pass filter order:** The high-pass filter order will designate how sharply the Butterworth filter Bode plot drops off at the specified cutoff frequency. In the examples shown throughout this work, a 4^{th} order filter was specified.

- **High-pass filter cutoff frequency:** The high-pass filter cutoff frequency is an important specification to indicate the cutoff below which the expected modal frequencies to be modeled lie, and above which frequency content should be considered noise or other unmodeled dynamics. A scaling factor was applied to typical general aviation aircraft flight dynamics to estimate a 3 Hz cutoff frequency to use in this work, which is outside of the expected range of scaled rigid-body dynamics that are modeled in the examples in this work. For other test data, estimates can be performed using sample data to determine the cutoff frequency. For the simulated test cases in this work, a 3 Hz cutoff frequency was used.

- **Residual threshold factor, λ_ν:** The residual threshold factor can be considered the most impactful SPLITR input parameter that influences the splitting results, and ultimately the entire model. This factor multiplies the noise estimate in each cell, and specifies how many standard deviations beyond the noise levels the splitting threshold should lie. It determines how easily a new split could be made, or how "trigger-friendly" the method is to allowing further splits. If λ_ν is too low, then the residuals will too easily be characterized as unacceptable, which can result in over-splitting. If λ_ν is too high, then splits will be very difficult to obtain, and will require exceptionally large residual magnitudes. For the basic simulated test data discussed in the next chapter with relatively low magnitude Gaussian white noise added to the data, a smaller residual factor of 2 was sufficient. However, for experimental flight

test data that include other unmodeled dynamics and noise content, disturbances, and other effects, a higher factor of 4 was used. If any test data are available in advance, they can be used to help inform the specification of this factor. For cases with no prior insight, these factors of 2 and 4 can be guiding estimates as a starting point. The risks of too many or too few splits may also be taken into account. If model simplicity is desirable, then fewer splits may be preferable. In general, additional splits should not deteriorate the model fit when there is a high split initialization proportion, in which case there may just be redundant cells. Future work can investigate adjusting this important parameter automatically.

- **Filter window:** An RMS of the high-pass filtered modeling data is used to inform the residual threshold over a specified window of data points. If that window is too wide, then it will require a significant amount of new data in a given cell to adjust the noise estimates of that cell. If the window is too small, then the noise estimate will not converge well. Prior data can be tested for noise convergence, but this work used a 2 second (100 data points) window for cases with variable noise content. If the noise is expected (or known) to be constant, then a much larger allowable window can be used for a more stable noise estimate.

The pass/fail status of each bin is used to determine if and where to split, and the following parameters influence the bin status.

- **Standard deviation factor, $\boldsymbol{\lambda_\sigma}$:** This factor arises in Eq. (3.20) to determine

how far the mean of both types of residuals needs to be drawn beyond the acceptable mean for a bin to fail. If it is specified as too low, then bins will fail and be conducive to splitting too easily, while if it is too high, bins will not fail easily enough and splits will be inhibited. In this work, $\lambda_\sigma = 0.75$ was used.

- **Number of standard deviations to normalize, $\boldsymbol{\lambda_{\sigma,norm}}$:** This factor in Eq. (3.21) is used to determine the degree to which a bin fails, for the purpose of comparing the relative failure with other groups of failed bins to choose the group that is most relevant to inform the split. This is considered the normalized severity that describes how many standard deviations of the acceptable residuals the mean of both types of residuals is pulled beyond the mean of the acceptable residuals. In this work, $\lambda_{\sigma,norm} = 1$ was used.

- **Total bin severity threshold:** To choose the group of failed bins that will be used to inform the split, the grouped severity must surpass a minimum bin severity threshold. Since the bin severity is saturated at 1 to account for cells that have only unacceptable residuals, this specification extends to determining the minimum number of grouped failed bins to split. In this work, this was set to 2 which requires at least 2 maximally failed bins, or otherwise greater than 2 failed bins.

- **Minimum # of points in bin for failure:** This specifies the minimum total number of points in a bin (regardless of residual characterization) for a bin to be considered failed. This will ensure that a bin with only few unacceptable

residuals is not used to inform a split, and that minimum data are obtained to make a determination on the bin status. In this work, this was set to 20 points. If this is too low, a bin may fail too quickly with too little information, whereas if this is too high, then it will have to wait for a large number of data points to inform a split.

Unrestricted residuals are an important part of the SPLITR logic to allow the local models to adjust prior to residual characterization that could lead to splitting. The associated user inputs are summarized below.

- **Number of unrestricted points before initial model:** This specifies how many points are allowed into the model as unrestricted to develop the initial global model with 1 cell, before points can be characterized. In this work, this was set to 250 points.

- **Number of unrestricted points following a split:** This specifies how many unrestricted points are allowed into the child cell models following a split before residuals are characterized as acceptable or unacceptable. If this is too low, then the model may not be given enough new data to adjust to the new region it is defined over, and further (unnecessary) splitting may result too quickly. If it is too large, then the splitting will take place too slowly. In this work, this was set to 150 points.

The SPLITR model update rates can vary between the parameter estimates and the split decisions, and these rates are described below.

- **Parameter update rate:** The parameter estimates in the associated local model are updated as each new data point is received. This update rate was set at 50 Hz for this work, as that is the update rate of the onboard flight control computer used for the experimental flight data.

- **Cell splitting update rate:** This specifies how often an additional split is considered. Depending on the dynamics of the system, it is usually not necessary to split the cells to reflect changing nonlinear dynamics as quickly as the parameter update rate. Additionally, there is a computational load associated with considering cell splits, so it is recommended to limit these checks. This specification should be set to a frequency near the maximum expected range of modal frequencies to be modeled to ensure that splitting does not occur slower than the dynamics of the system may change. In this work, it was set to 5 Hz, but could be much slower to mitigate computational requirements for the examples shown.

The global model is constructed as a weighted combination of the local models, and the smoothness applied can be varied.

- **Smoothness factor, $\boldsymbol{\lambda_s}$:** This determines the width of the Gaussian weighting functions, as discussed in Section 2.4.1. A large smoothness factor can be used for a system in which the cells have a significant amount of influence on each other, whereas a small factor is used for a system where the localized models in each cell should remain largely local, and the weighting functions are primarily to smooth through the cell boundaries. The individual cell Gaussian

standard deviations are also a function of cell width. This factor is set to 1 by default.

The final set of user inputs describe the balance required in remembering vs. forgetting information as the model is developed in real time.

- **Forgetting factor, λ_{ff}:** As discussed in Section 3.3, there is a constant tradeoff between passing along too much parent information, or effectively too much certainty that is hard to overcome with new data, versus not enough information and essentially throwing away past data. If λ_{ff} is too high (i.e. close to 1), then it will require a substantial amount of new data to overcome the parent cell's information matrix, and further unnecessary splits may occur as a result. This may be preferable though if the system is expected to be mostly linear, and the child model will not differ much from the parent. However, for a highly nonlinear system, if the child cells are expected to vary significantly from the parent, then a forgetting factor may expedite the convergence by "forgetting" more of the parent model's information. This forgetting factor can be applied only while the unacceptable residuals are incorporated into the child cells during the splits to expedite the convergence, or it can be applied continuously to allow the models to adjust as new data are received. Note that if a continuous forgetting factor is too low, splits may not occur at all because there will effectively be a single model that rapidly changes in time to adapt to the current conditions. In this work, a constant forgetting factor of 0.995 was used.

- **Cell initialization proportion:** This factor is an additional proportion used to multiply the parent information matrix to initialize the child cell information matrix. Note that this matrix is already proportioned based on the number of parent data points belonging to each child cell. To pass along more information, this can be set to 1. To enable the child cells to adjust more quickly, it can be specified as less than 1. In this work, a value of 0.2 was used for a piecewise linear test case, while a value of 1 was used for all other examples.

The preceding discussion on SPLITR user inputs, as well as the recommended initial settings and specifications, was based upon the simulation testing and sensitivity investigations discussed in Chapter 4 and flight data examples in Chapter 5. Note that several inputs, particularly the number of minimum points in a bin for failure, the number of unrestricted points before the initial model, and the number of unrestricted points following a split, do not directly reflect the performance or convergence properties within each bin or cell. Future work can aim to automate these specifications (and others) based on further analysis and statistical metrics.

The SPLITR user inputs summarized in this section contain many heuristic specifications that may be tuned with additional insight into a particular application and data set to improve the results, and future work can look at automating many of them. However, the examples shown throughout the next two chapters demonstrate that most of these parameters do not need to be modified at all from the baseline specifications to use the SPLITR algorithm well. Additionally, small changes in

many of them only have a minor impact on the modeling results. For the parameters that play a larger role in driving the model development, these explanations can provide insight for initial values which can be modified if necessary once data have been obtained.

3.6 Summary

This chapter presented a novel approach to automated global nonlinear modeling using local model networks in real time. Although this method can be generally applied to any data set, this research was conducted within the scope of aerodynamic modeling under the NASA L2F concept, which strongly impacted the technique development. In this approach, local linear models were used with a constant user-specified local model structure, but these can be generalized in future work to allow local nonlinear models and/or automated local model structure determination. Additionally, the splitting logic developed in this chapter was limited to adding additional cells based on a user-specified maximum resolution, while future work can consider model "pruning," or effectively removing past splits that are deemed unnecessary.

In this work, recursive equation-error least-squares parameter estimation was used to update the local models in each cell using data in the time domain. It is recursive to allow the local models to be updated with new data as they are received in real time. Equation-error least squares is a straightforward non-iterative modeling method that is compatible with recursive updates. Time domain data were used so

that the data in each cell do not need to be contiguous in time for each local model.

The cell structure determination process was uniquely developed for the challenges associated with real-time onboard cell adaptation. A residual analysis procedure was described as a way to determine split locations based on residual characterization in a set of bins across the range of the PVs. Residuals are characterized as acceptable to indicate that they are fit well by the associated current local model, unacceptable to indicate that they are not captured well by the current local model and are not attributed to noise, or unrestricted to allow the local model to develop. The residual characterization takes into account the noise content in the data in each cell to adjust the expectations of model fit. Finally, the SPLITR algorithm user inputs described throughout this chapter are summarized and explanations of the impact of each input are offered in a concise overview of the algorithm specifications.

The SPLITR method developed in this chapter will be further explored and results will be shown using simulated test data in Chapter 4 and flight data in Chapter 5.

Chapter 4

SPLITR Case Studies and Sensitivity Investigation with Simulated Test Data

SPLITR was introduced in Chapter 3 as a novel real-time automated global nonlinear modeling approach that was motivated by aerodynamic modeling in the context of the NASA L2F concept. This chapter will test the SPLITR approach and evaluate the results using simulated test data from simple specified functions before moving on to application-based test data in Chapter 5. This allows the SPLITR method to be presented, visualized, and understood better with known functions before expanding the problem complexity. The sensitivity of SPLITR modeling results to some of the user specifications discussed at the end of Chapter 3 will also be investigated in this chapter.

There are several descriptive figures shown throughout Chapters 4 and 5 that aim to clarify how the SPLITR method operates and to understand and visualize the results. They will be explained in detail only for the first example through which they are shown. Early test cases will also describe the entire model develop-

ment process in detail, while subsequent examples will serve as demonstrations that highlight only particular features of the results, and/or explore various aspects such as sensitivity to user specifications. Unless otherwise specified, all of the default specifications described in Section 3.5 were used throughout this chapter.

4.1 1D piecewise linear data

Because the SPLITR method operates by determining local linear regions through which to partition a nonlinear function, the first simplified test case looks at a piecewise continuous linear model that is defined in Eq. (4.1) for $0 \leq x \leq 1$.

$$y = \begin{cases} 10x & 0 \leq x < 0.2 \\ 2x & 0.2 \leq x < 0.6 \\ 10x & 0.6 \leq x < 0.8 \\ 2x & 0.8 \leq x \leq 1 \end{cases} \tag{4.1}$$

This function will be used to demonstrate the SPLITR model development in detail through a baseline test case, as well as explore the sensitivity of the results to the minimum cell width parameter and to additive Gaussian white noise in the modeling data. In each example, two passes through the input space were simulated, with 2500 evenly spaced data points from $x = 0$ to $x = 1$, and then an additional 2500 points from $x = 1$ to $x = 0$. A 50 Hz sample rate was assumed, with 100 seconds of test data. The child cell initialization proportion was set to 0.2 since the child cells are expected to vary significantly from the parent models. The minimum cell width

was specified as 0.05.

4.1.1 Baseline case

In this example, the piecewise linear function in Eq. (4.1) was used to demonstrate how the SPLITR algorithm operates based on the detailed explanation given in Chapter 3.

The time history of the modeling data generated from this function is shown in Fig. 4.1. Gaussian white noise was added to the output data with a standard deviation of 0.075 to obtain an SNR of 20. As shown in Fig. 4.1, the splits occur in time during the first pass of the input space, shortly after each breakpoint in the piecewise linear data when the algorithm recognizes nonlinear behavior outside of the linear representation of the previous segment. The second pass of the input space repeats the full range of x-values to allow the parameters in each cell to adjust with new data following the splits.

The evolution of the cell structure in Fig. 4.2 shows how the cells are successively partitioned and new ones are added across the range of x during each split. Initially, there is one cell defined across the entire range of x. The first split partitions the piecewise linear data in $0 \leq x < 0.2$ from the rest of the input space, resulting in cells 2 and 3. Accordingly, the second split partitions the second linear segment defined over $0.2 \leq x < 0.6$ from the rest of cell 3, resulting in a structure of cells 2, 4, and 5. Finally, the last split partitions the third segment defined over $0.6 \leq x < 0.8$, resulting in final cells 2, 4, 6, and 7, with the cell boundaries placed

at the piecewise linear breakpoints. This cell numbering scheme is used throughout the results shown in this work, and the cell structure evolution depictions can be used as a source for other figures and discussions that reference the cell numbers.

The final cells are shown in Fig. 4.3 across the range of x, along with the modeling data and the piecewise linear models. The bin locations, and correspondingly all possible split locations, are shown as light gray dashed lines across x, and indicate that the "correct" split locations were chosen. The local models in each cell are shown to represent the data in that region well, with the residuals confirming a good model fit.

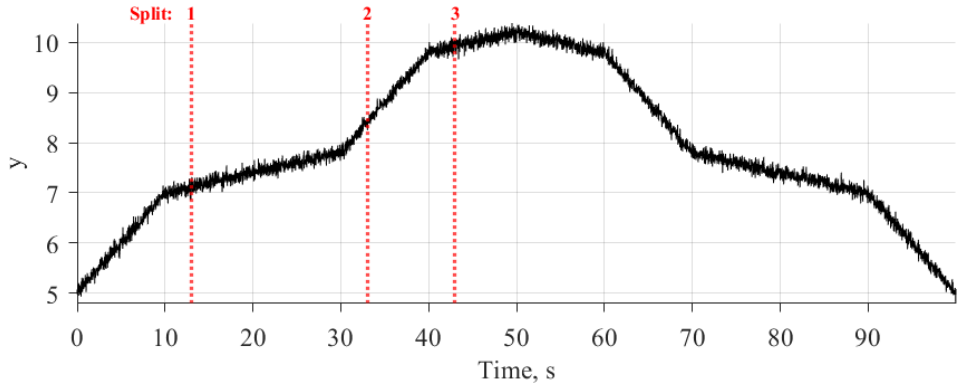


Figure 4.1: Time history of response variable for piecewise linear baseline case.

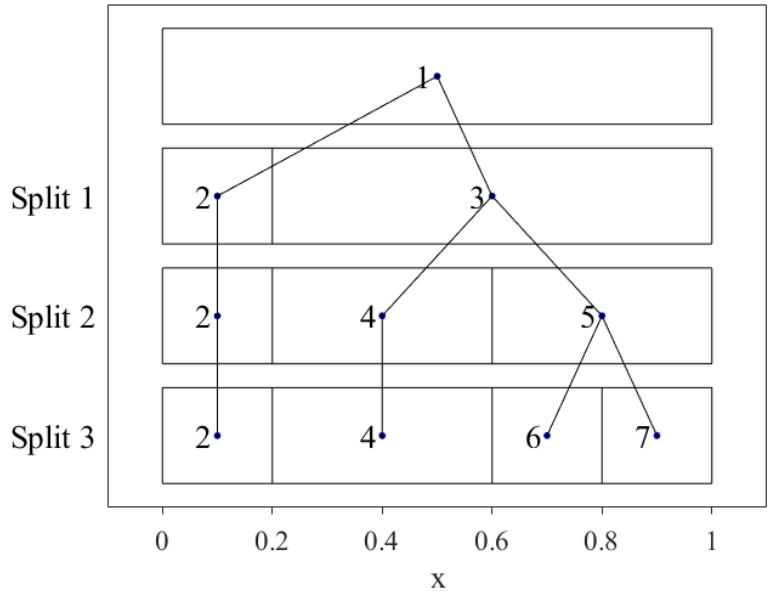


Figure 4.2: Cell structure evolution for piecewise linear baseline case.

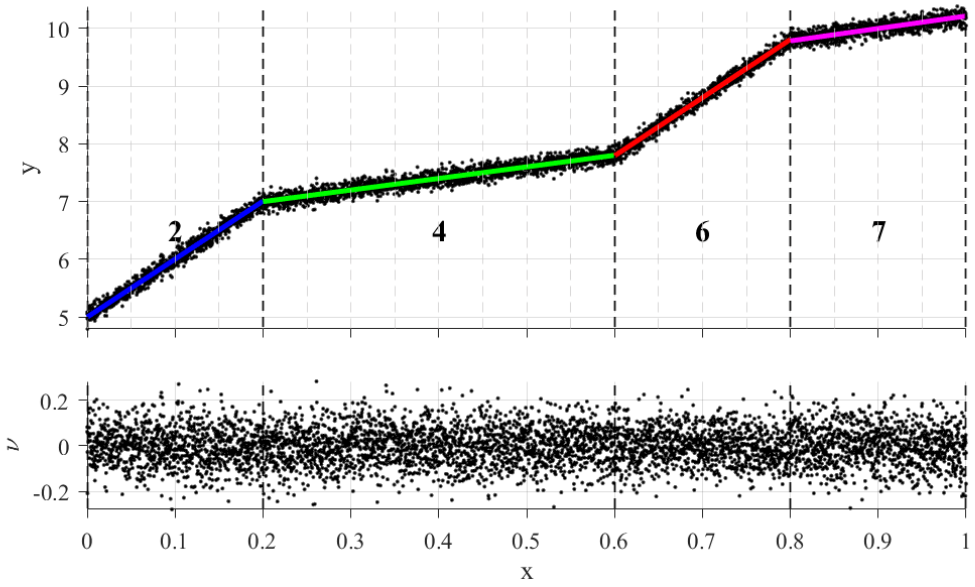


Figure 4.3: Local model fit and residuals for piecewise linear baseline case.

The split logic and decision making that resulted in the final cell structure and model parameters are based on the residual characterization procedure discussed

in Chapter 3. A closer look through a depiction of the cell structure decision-making process will clarify how the cells evolve over time based on the available data. Figure 4.4 displays the binned local residuals just prior to each split, i.e. it shows a characterization of the local model fit quality that triggers a split. The last plot shows the final cell structure once all of the data are obtained. The residuals are plotted across the complete range of x in each plot, and the thick vertical dashed red lines mark the boundaries of the cells. The light gray lines indicate the minimum cell width bins across x. Although these bins are constant across all cell structures, the cell to which each of them belongs may change. The bin boundaries are outlined in blue to indicate active grouped bins within the range of new data in a given cell, and are also assigned an active bin number shown at the center and top of each bin. The numbers for the bins that are considered failed at the time of the split are also colored red. The active range of bins that contain new data in each cell is also outlined on each side with the bin boundaries shown in red. Note that since the maximum number of cell bins is 10, minimum-width bins may be combined to form the active bins (although that is not necessary in this case).

The acceptable residuals are shown as black dots, the unacceptable residuals as red $\times$'s, and the unrestricted residuals as cyan circles. The mean and standard deviation of the residuals in each bin are shown in green, red, and blue, for the acceptable residuals, the unacceptable ones, and both of the two together, respectively. Only residual data obtained in each cell following a prior split are shown, such that data from the parent cell are excluded. The exception to this is for the stored parent unacceptable residuals that are incorporated into each child cell following a split.

Recall that those stored and inherited data points are considered unrestricted, and are not used to define active bins.

Beginning with the first linear model developed for $0 \leq x < 0.2$, notice that although there are several residuals that are characterized as unacceptable, the mean of both types of residuals is still small, and so the bins pass the status check. Beyond the breakpoint at $x = 0.2$, however, the unacceptable residuals outweigh the acceptable ones as the linear model with a constant slope can no longer represent all of the data adequately. Bins 3—4 are considered failed, so the first split is performed at the left boundary of bin 3. Correspondingly, the second split partitions the data in $0.2 \leq x < 0.6$ with the split location at the left boundary of bin 8 in cell 3. Similarly, the last split occurs at the left boundary of bin 4 in cell 5 to obtain the four final piecewise linear models across the range of x. Most of the residuals shown in the last plot for the final cell structure were obtained during the second pass of the data, and they exhibit a good fit in each cell. There are some unacceptable residuals shown throughout the cells, but there are no failed bins since most of the residuals are acceptable. Many of the unrestricted residuals in cyan are designated as such due to the minimum number of allowable points following a split. In this simplified example, 150 points may be more than necessary, but it was used consistently across all examples.

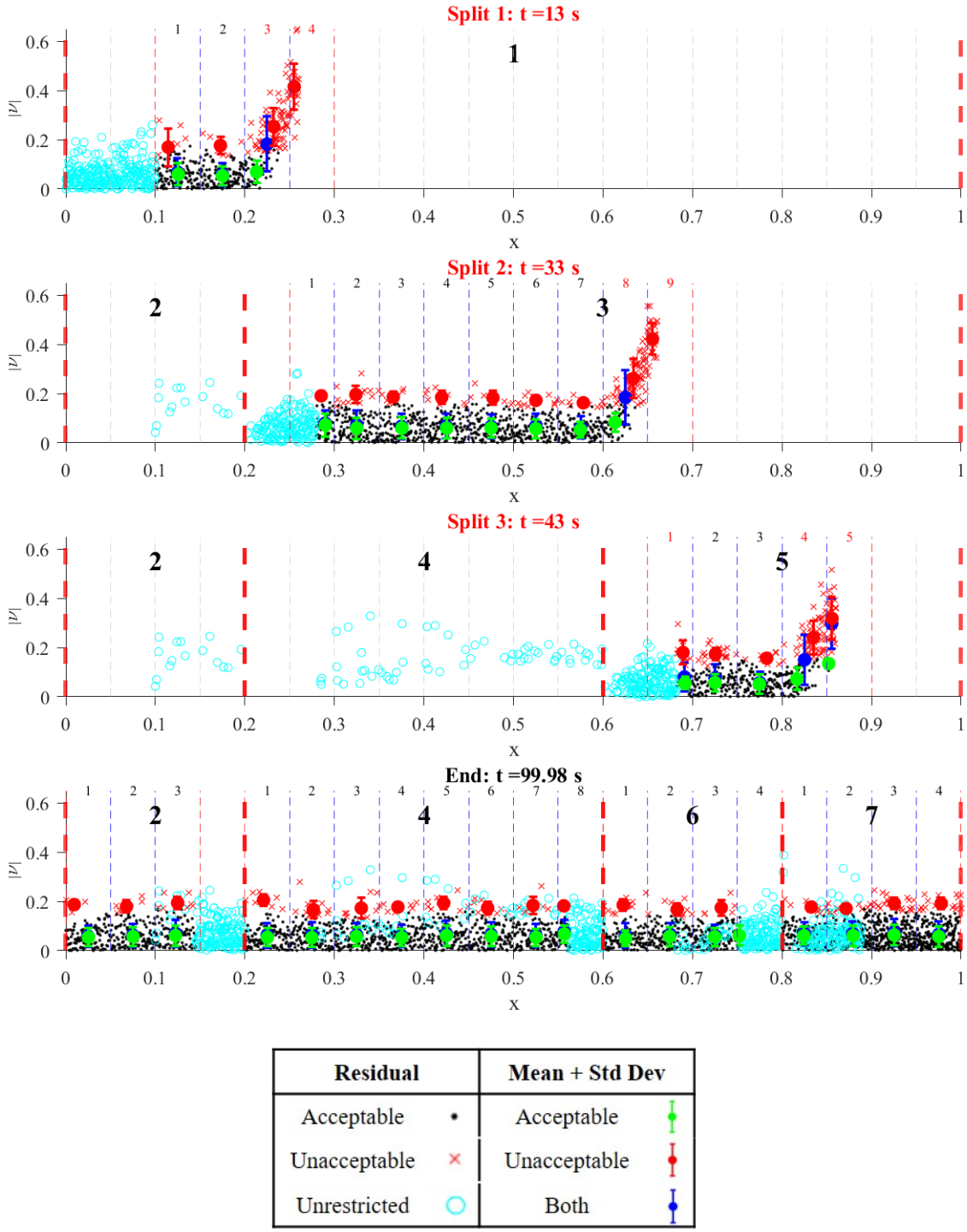
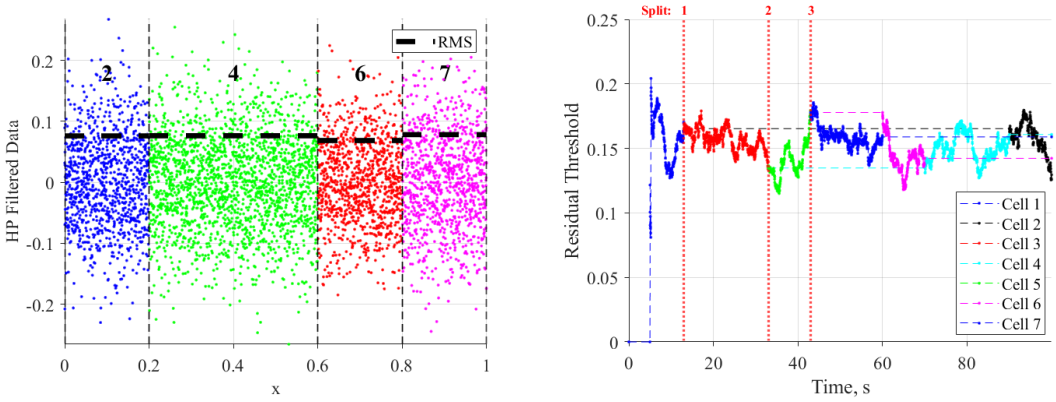


Figure 4.4: Binned local residual characterization for piecewise linear baseline case.

The residual characterization is used to inform the cell splits, and this pro-

cedure can be better understood by describing how the residual threshold is computed, since that is what determines the cutoff between acceptable and unacceptable residuals. The residual threshold distinguishes between residuals associated with acceptable modeling error due to noise, and those that are indicative of behavior inconsistent with the current local model. The modeling data have constant magnitude Gaussian white noise across the range of x as shown in the high-pass filtered modeling data in Fig. 4.5(a). The RMS of the noise in each cell is shown with a black dashed line, with a global RMS of 0.075. A moving window of the RMS of the noise in each cell, multiplied by the residual threshold factor of 2, is shown in Fig. 4.5(b). Given the constant magnitude noise estimate, the residual threshold across all cells is shown to be around 0.15. Variations in the residual thresholds are due to the relatively small noise RMS window of 2 seconds. The residual threshold is a time-varying value in each cell that is dependent on the noise estimates for that individual cell. Throughout time, if a residual point lies above the current residual threshold for the cell that it belongs to, it is characterized as unacceptable; otherwise, it is considered acceptable. Since each measurement belongs only to a single cell, only the threshold for at most one cell is updated at each point in time, with these points connected with a dashed line.



(a) High-pass filtered modeling data

(b) Residual threshold

Figure 4.5: Residual threshold characteristics for piecewise linear baseline case.

Finally, once the cell structure decision making is understood, the final models are presented. The time histories of bias y_0 and slope y_x parameters across all cells are shown in Fig. 4.6. Notice the sharp jumps following the splits when the child cell parameters are initialized with high uncertainty on the parent cell information, as well as by incorporating the parent cell's unacceptable residuals immediately. The initialized uncertainty in the dispersion matrix is also responsible for the relatively quick convergence of each child cell model's parameters. Since each data point only belongs to a single cell at each point in time, a dashed line is used to extend the estimates from other cells while they are not being updated. The final parameter estimates are shown in Fig. 4.7 for the local linear models, along with 95% confidence intervals. The true parameter values from Eq. (4.1) are also shown, indicating acceptable fits within the uncertainty.

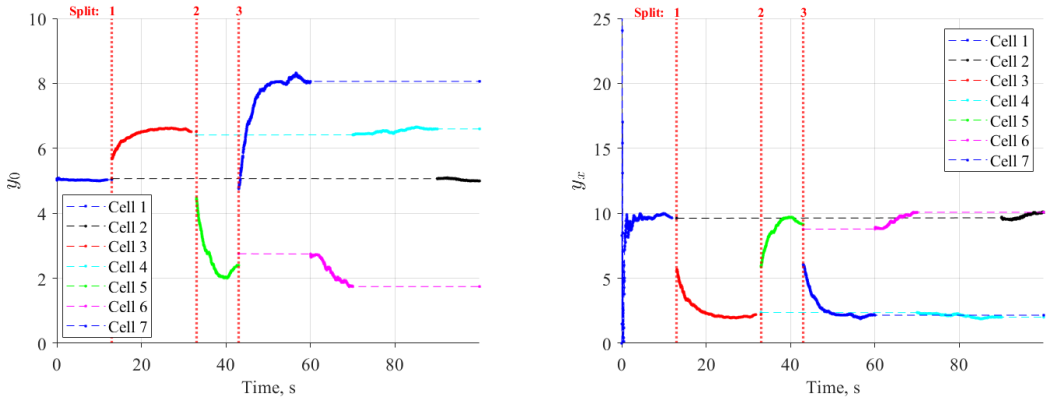


Figure 4.6: Time history of parameter estimates across all cells for piecewise linear baseline case.

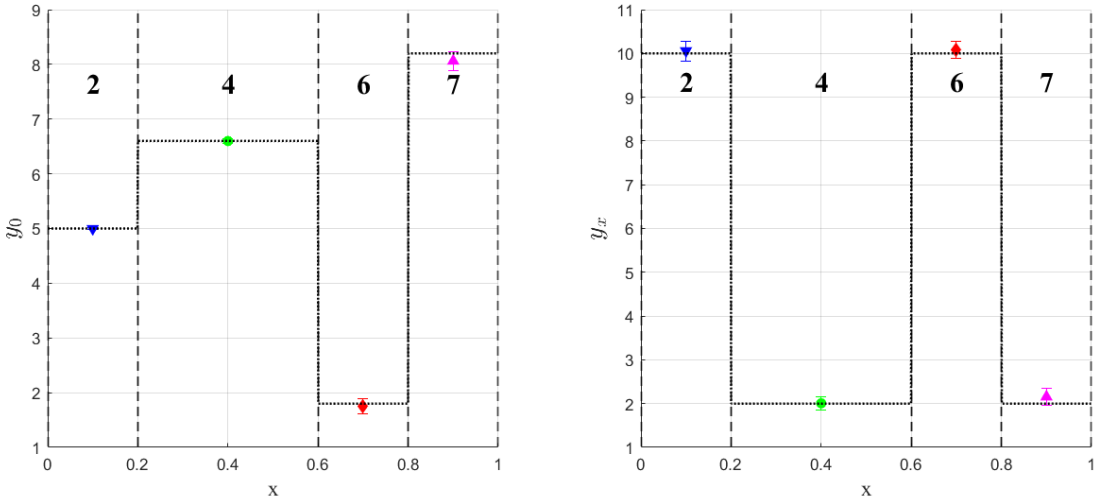


Figure 4.7: Local model parameter estimates across all cells for piecewise linear baseline case.

4.1.2 Minimum cell width

This example demonstrates how the minimum cell width specification can impact the resulting cell split locations. The piecewise linear function was used to run 25 simulation test cases with a minimum cell width of 0.05, and the resulting split locations and model fits are shown for all cases in Fig. 4.8. All possible split

locations are shown as dashed gray vertical lines. Every test case was successful in determining the correct split locations at $x = 0.2$, $x = 0.6$, and $x = 0.8$, as well as providing good local model fits to the data.

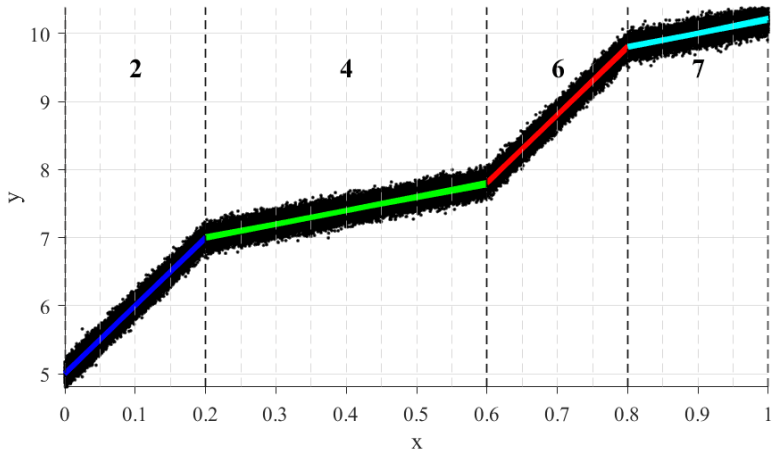


Figure 4.8: Local model fit for piecewise linear model with 25 simulation cases and a minimum cell width of 0.05.

When the minimum cell width was reduced to 0.025, the resulting cell structures are shown in Fig. 4.9 for 25 simulations, with the chosen splits indicated, along with how many runs resulted in each split location. Although the local model fits are still acceptable in each cell, the increased allowable split resolution makes it more difficult to distinguish the true split locations through the noise content in the data. If split location precision is highly prioritized, it is important that the minimum cell width is wide enough so that the discrete locations are not hidden by noise. It is also important that the proposed locations lie at those true split points, which may not be known in advance. Specifying a reasonably low maximum number of cell bins can help ensure that initially wider split locations are considered. However, this is an extreme example of a piecewise linear function, whereas for more practical

higher order data sets, the "true" split location is less straightforward.

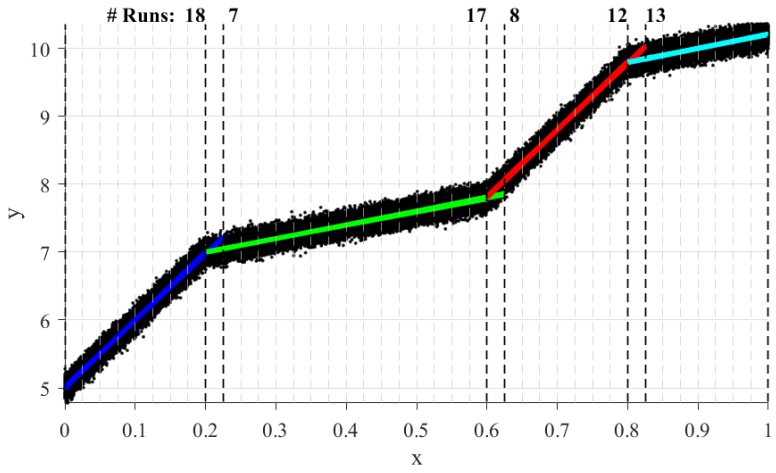
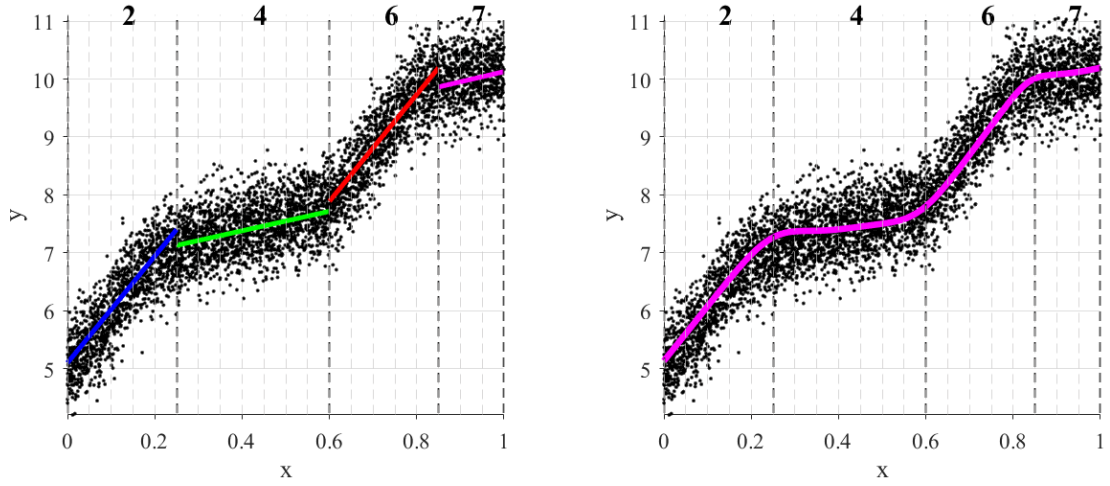


Figure 4.9: Local model fit for piecewise linear model with 25 simulation cases and a minimum cell width of 0.025.

4.1.3 Modeling data noise sensitivity

As demonstrated in the previous example, the noise content in the data may have a small influence on the resulting split locations, but the SPLITR method results are not overly susceptible to overfitting the noise in the data, considering that a sufficient estimate of the noise is obtained. This is because the residual threshold, which determines the splits, is based on an estimate of the noise content in the data. When the SNR for the modeling data was reduced to 4, the resulting split locations and piecewise linear model fits are shown in Fig. 4.10(a), with the global smoothed model in Fig. 4.10(b). Although there is a significant increase in noise magnitude compared to the baseline case, and the residuals are accordingly much larger, there are still only 4 cells placed in reasonable locations. In this case, the global smoothed model plays a role in outputting a more useful final model.

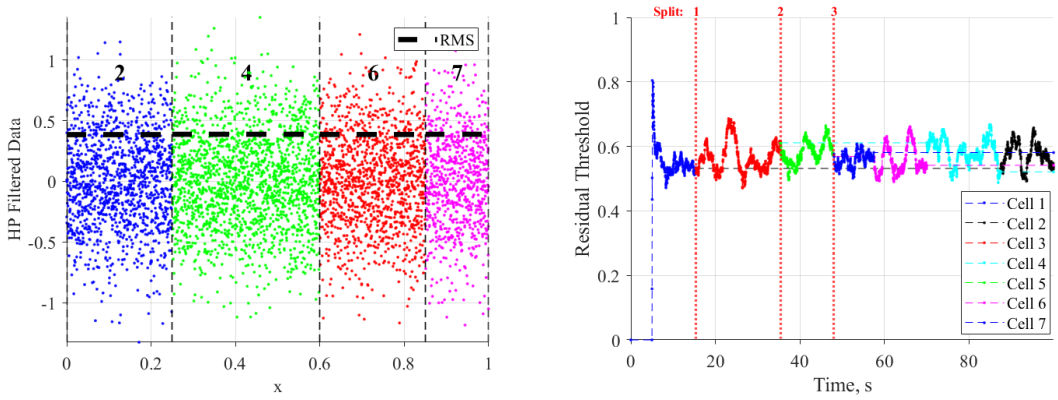
The high-pass filtered modeling data are shown in Fig. 4.11(a) with an RMS of 0.386, and the residual threshold over time is shown in Fig. 4.11(b) with $\lambda_\nu = 1.5$. Compared to the baseline cases of noise and residual threshold shown in Fig. 4.5(a) and Fig. 4.5(b), respectively, notice that it was more difficult to trigger a split in this case given the low SNR, and correspondingly large residual threshold. This is an important characteristic of the SPLITR method to prevent excessive splits to overfit data with significant noise.



(a) Local model fits

(b) Global smoothed model fit

Figure 4.10: Local and global model fits for piecewise linear data noise sensitivity test case with SNR = 4.



(a) High-pass filtered modeling data

(b) Residual threshold

Figure 4.11: Residual threshold characteristics for piecewise linear data noise sensitivity test case with SNR = 4.

4.2 1D quadratic data

The next set of examples looks at 1D quadratic data to demonstrate how higher order functions may be modeled as a collection of linear components, while relying on the global smoothing to tie the local models together. The quadratic test data are generated through the function shown in Eq. (4.2) with $-5 \leq x \leq 5$, and a minimum cell width of 0.25.

$$y = 0.5x^2 \tag{4.2}$$

This section shows test data results for a baseline test case, as well as to demonstrate the sensitivity of the results to the filter window, the smoothness factor, the residual threshold factor, varying noise magnitude, varying noise through multiple simulations, and model adaptation.

In each example, 2 passes through the input space were completed, with 2500

evenly spaced points during each pass. In contrast to the baseline piecewise linear test case in which each piecewise segment changes drastically from the previous one, a local linear model would be constantly varying for quadratic data, with the child cell models not significantly different from the parent models. Therefore, this example used a cell initialization proportion of 1, as discussed in Section 3.5.

4.2.1 Baseline case

This baseline quadratic test case shows how the SPLITR method may be used to model higher order test data. Figure 4.12 displays the time history of the modeling data. Gaussian white noise with an RMS of 0.21 was added to the output data for an SNR of 18. Similar to the piecewise linear case, the splits occur during the first pass of the x-data, while the second pass is devoted to updating the parameters.

The cell structure development is shown in Fig. 4.13 with relatively evenly spaced partitions to characterize this higher order function with linear segments. The final split locations and local linear models are presented in Fig. 4.14(a), with the global weighted model in Fig. 4.14(b). The Gaussian validity functions that provide the global weighting are shown in Fig. 4.15(a), with the normalized validity functions in Fig. 4.15(b). Recall that the Gaussian weighting functions are centered in each cell, with a standard deviation that is proportional to the cell width. As a result, the cells with smaller widths have reduced validity as indicated by the magnitude of w and the more limited reach of those functions, whereas the larger cells have a stronger influence on the global model output. The high-pass filtered

output data in Fig. 4.16(a) show the constant magnitude Gaussian white noise in the output data. This is used to inform the residual threshold across the cells, as presented in Fig. 4.16(b) with $\lambda_\nu = 2$.

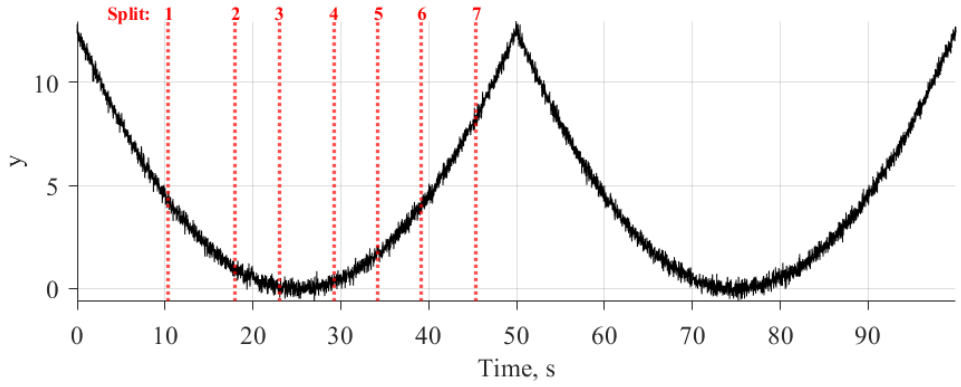


Figure 4.12: Time history of response variable for quadratic baseline case.

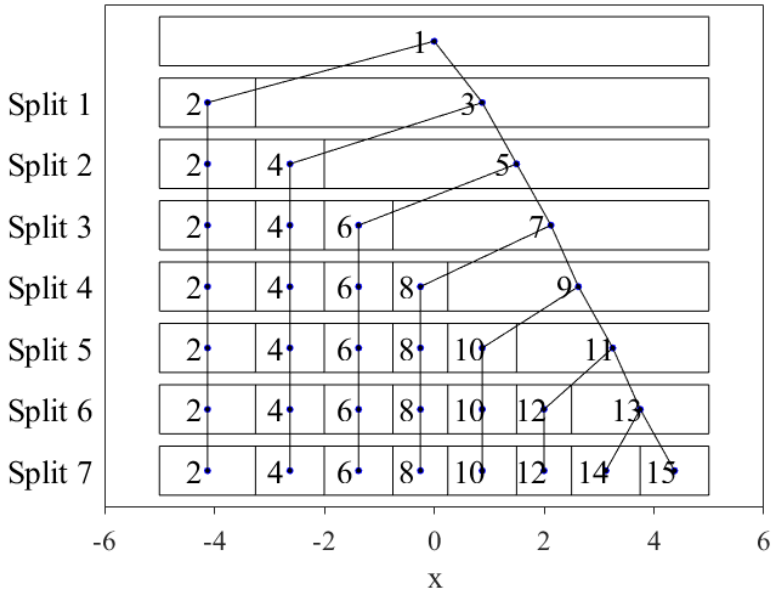
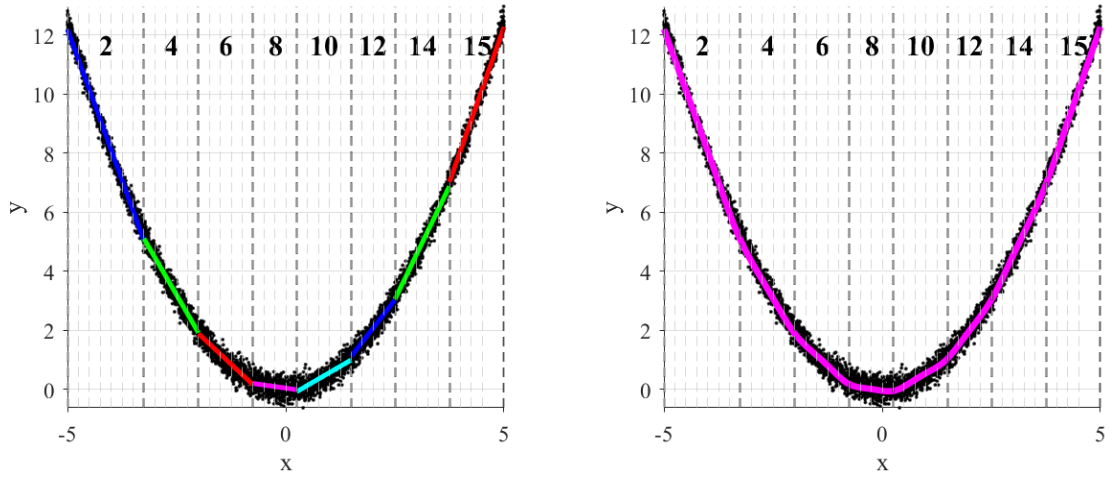
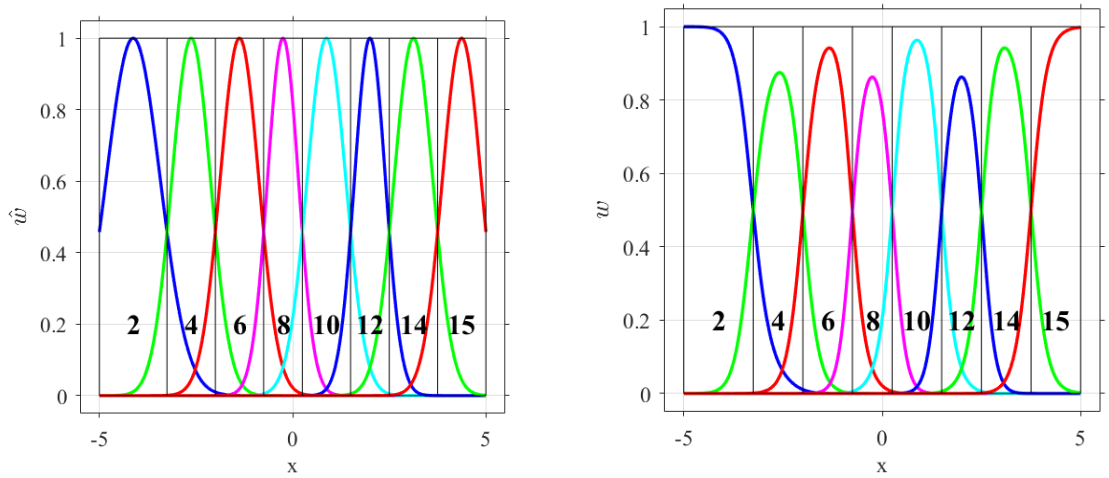


Figure 4.13: Cell structure evolution for quadratic baseline case.



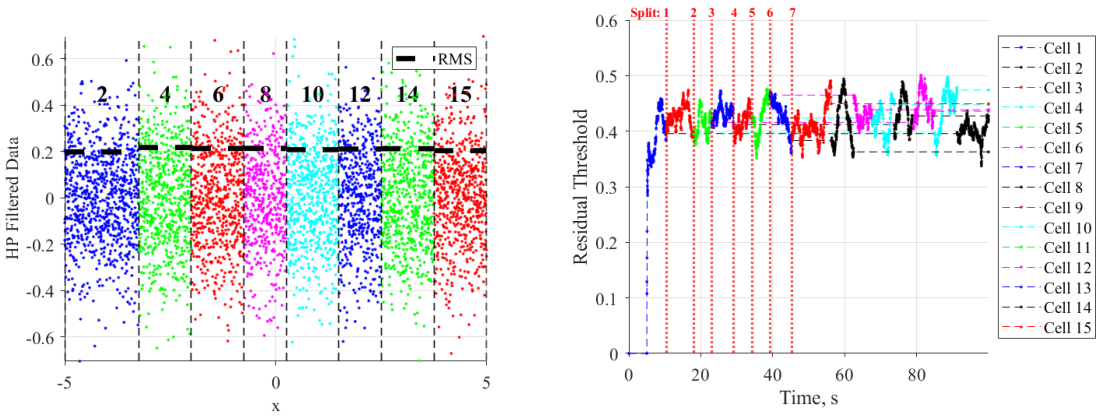
(a) Local model fits (b) Global smoothed model fit

Figure 4.14: Local and global model fits for quadratic baseline case.



(a) Validity functions (b) Normalized validity functions

Figure 4.15: Validity functions for quadratic baseline case.



(a) High-pass filtered modeling data

(b) Residual threshold

Figure 4.16: Residual threshold characteristics for quadratic baseline case.

The parameter estimates for the final cells are shown in Fig. 4.17, along with the true estimates obtained from analytical derivatives. Each local model parameter matches the true value within the estimated uncertainty, except for the parameters in cell 15, where the true values lie just outside the 95% confidence intervals. While the forgetting factor is useful to allow the child cells to more rapidly adjust from the parent models, it may also cause information to be unnecessarily lost once the initial adjustment has occurred.

The time history evolution of the parameters is also shown in Fig. 4.18. It is apparent that, in contrast to the piecewise linear example that exhibited very sharp transitions in the parameters between cell models, for this higher order data the child cell models require less of an adjustment from the parent models.

Finally, Fig. 4.19 shows the global parameter estimates for each of the parameters that are weighted by the normalized Gaussian validity functions. Since all of the local models have the same model structure, the effective global parameter estimates can be visualized individually as the weighted combination of each of the

local parameter estimates. While the true parameter estimates for y_0 and y_x across x should be parabolic and a straight line, respectively, the weighting introduces oscillatory behavior between cells. This is also apparent in the global weighted model output in Fig. 4.14(b), which is obtained as a weighted combination of the local model outputs. This characteristic could be improved with a more appropriate weighting that is customized for a given data set and cell structure.

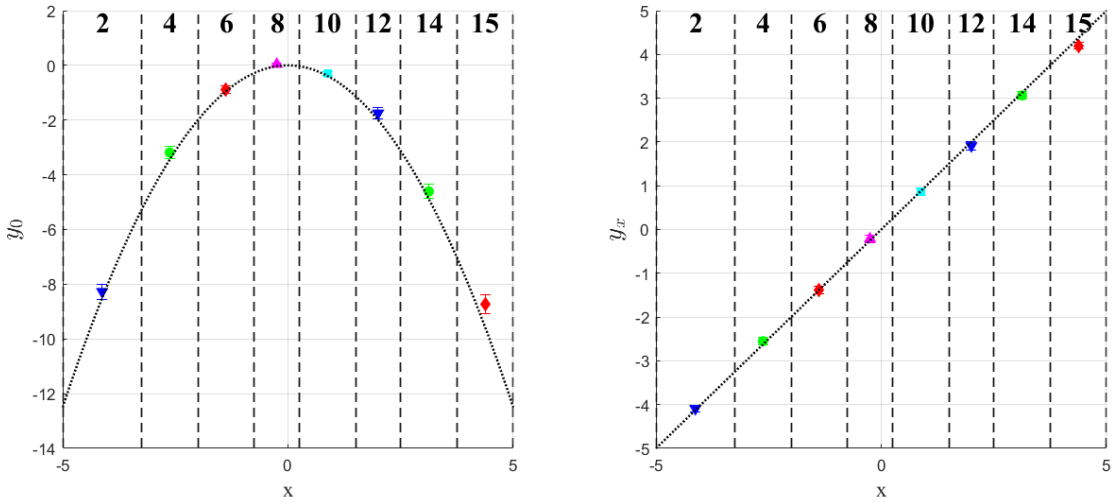


Figure 4.17: Local model parameter estimates across all cells for quadratic baseline case.

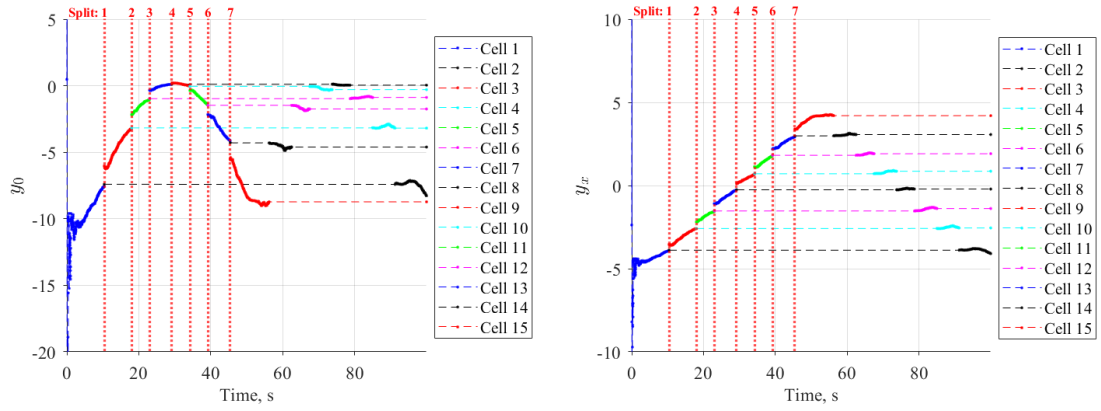


Figure 4.18: Time history of parameter estimates across all cells for quadratic baseline case.

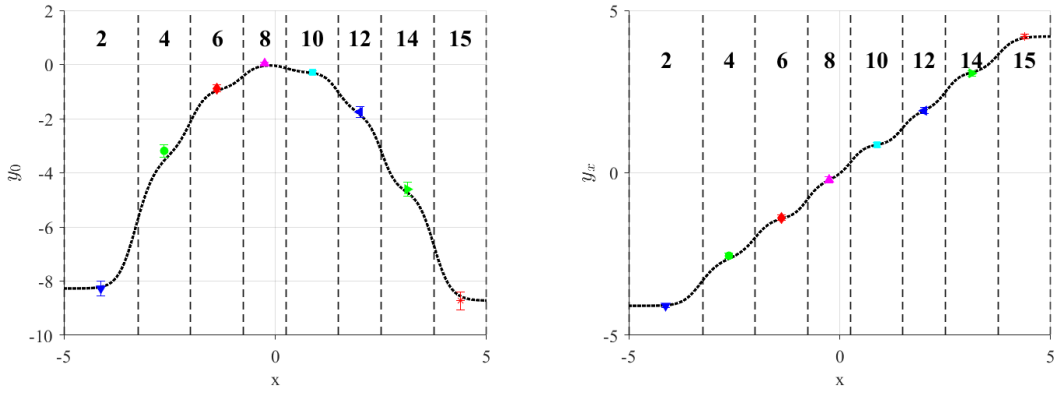


Figure 4.19: Local and global parameter estimates for quadratic baseline case.

4.2.2 Filter window

This example shows the effect of the filter window specification. In the baseline quadratic test case, the filter window was set to 2 seconds, and as a result, the residual threshold factor shown in Fig. 4.16(b) fluctuates significantly about the true point. If instead the filter window was set to infinite time, the resulting cell structure development is shown in Fig. 4.20, with the model fits in Fig. 4.21, and with the results not differing significantly from the baseline case. The high-pass filtered noise data are displayed in Fig. 4.22(a), and the residual threshold is shown in Fig. 4.22(b). This example exhibits a much more stable estimate of where the residual threshold lies across all cells, given the constant magnitude Gaussian white noise added to the data. If the noise characteristics are expected to be constant across the entire data set, a larger filter window will ensure a more stable residual threshold, and therefore a more consistent characterization of residuals across the cells.

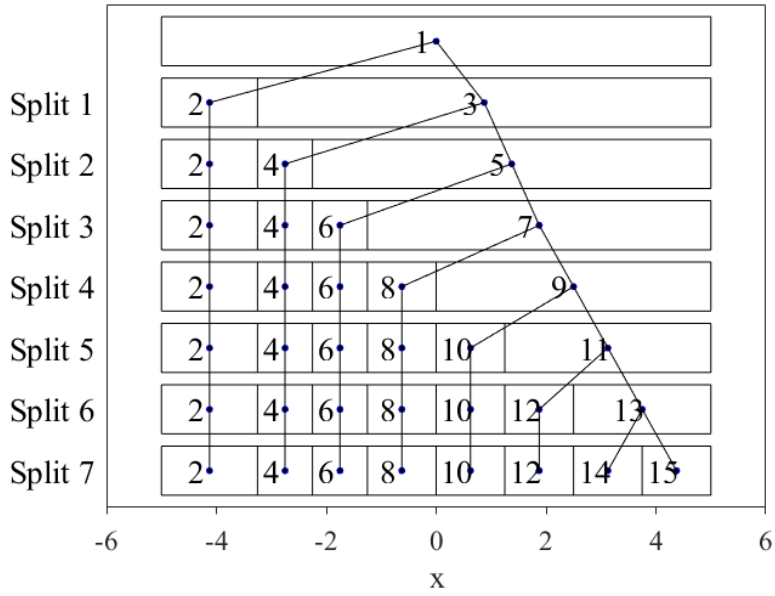
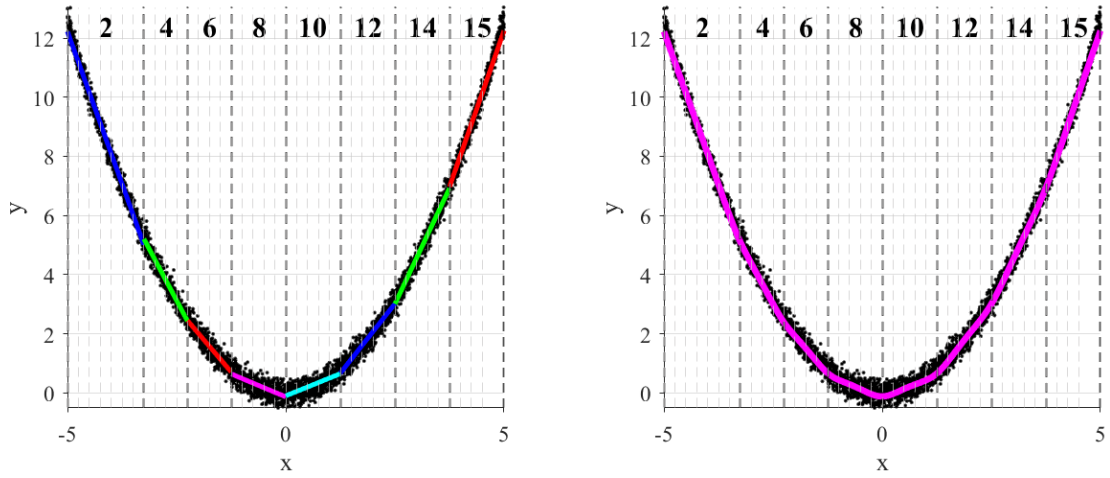


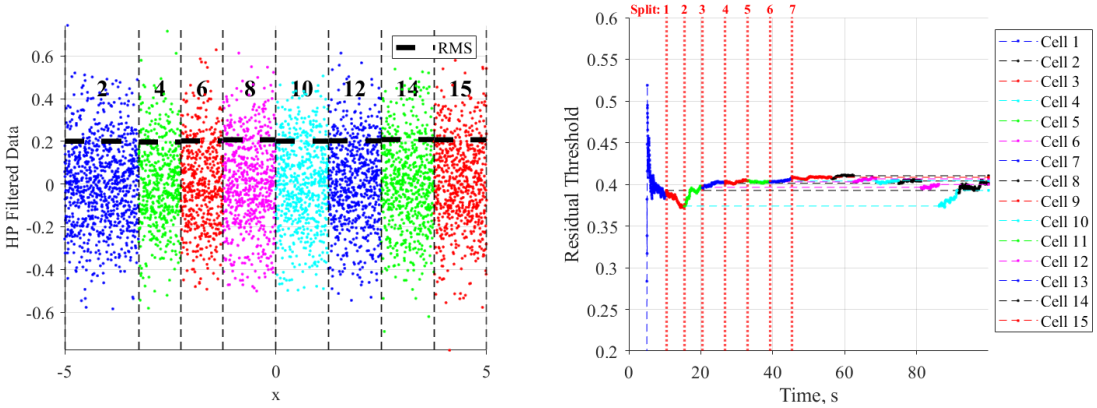
Figure 4.20: Cell structure evolution for quadratic model with filter window variation.



(a) Local model fits

(b) Global smoothed model

Figure 4.21: Local and global model fits for quadratic model with filter window variation.



(a) High-pass filtered modeling data

(b) Residual threshold

Figure 4.22: Residual threshold characteristics for quadratic model with filter window variation.

4.2.3 Smoothness factor

This example shows the impact of the global weighting smoothness factor λ_s, which influences how the local models are combined. For the baseline case in Fig. 4.14(b) with $\lambda_s = 1$, notice that there are apparent waves in the global model fit, particularly in the regions crossing over the cell boundaries between the local models.

If $\lambda_s = 0.25$ instead for the same baseline test data, the global smoothed model is shown in Fig. 4.23 with even more sharp variations in slope across the cell boundaries. The validity functions in Fig. 4.24 are also much more narrow compared to the baseline validity functions in Fig. 4.15, and appear almost as boxcar functions. This could be used for a case in which the model is expected to exhibit highly localized behavior and where a smooth transition is less critical, such as in the piecewise linear case in Section 4.1.1.

When instead $\lambda_s = 1.5$, the resulting global model is shown in Fig. 4.25. While the global output is much smoother in this case, there is added error particularly around $x = 0$ where the local models far outside this region are influencing the global model output at that point. This is confirmed in Fig. 4.26 which shows the significant overlap in validity functions across the cells, with the inner cells affected most strongly. This characteristic could be highly desirable if, for example, smooth derivatives of the model are required for purposes such as control law design. The unique composition of the SPLITR model as a combination of smaller parts also indicates that the best representation of the data may not be a model with the minimum error if there are other considerations such as smoothness. There may be tradeoffs between obtaining a good local model fit and a continuous global smoothed model. One advantage of the SPLITR method in this context is that the local models are estimated independently of the adjacent cells and of the validity functions, which offers flexibility in designing and customizing the weighting functions for a specific application.

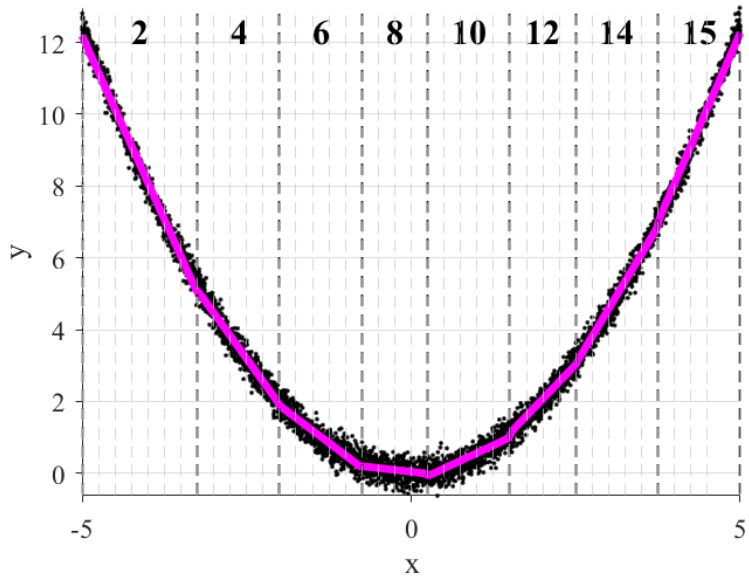
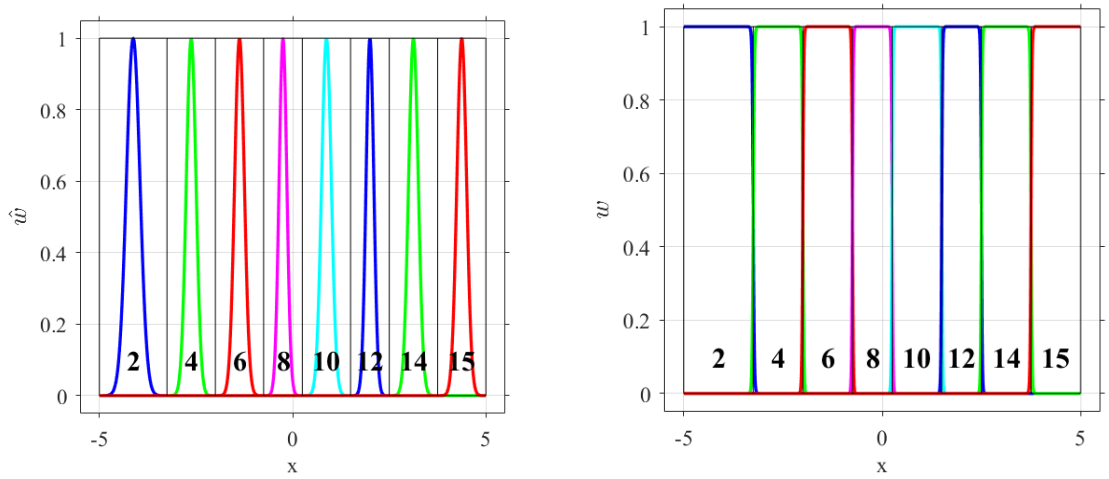


Figure 4.23: Global smoothed model fit for quadratic test case with smoothness factor $\lambda_s = 0.25$.



(a) Validity functions

(b) Normalized validity functions

Figure 4.24: Validity functions for quadratic test case with smoothness factor $\lambda_s = 0.25$.

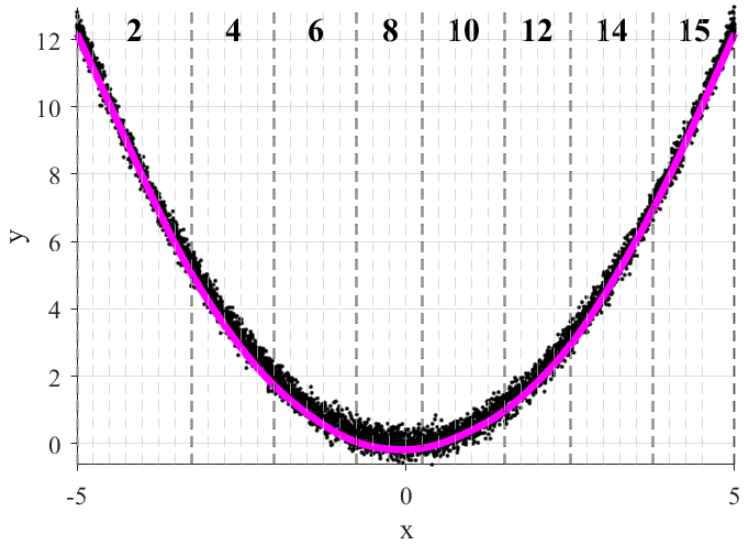
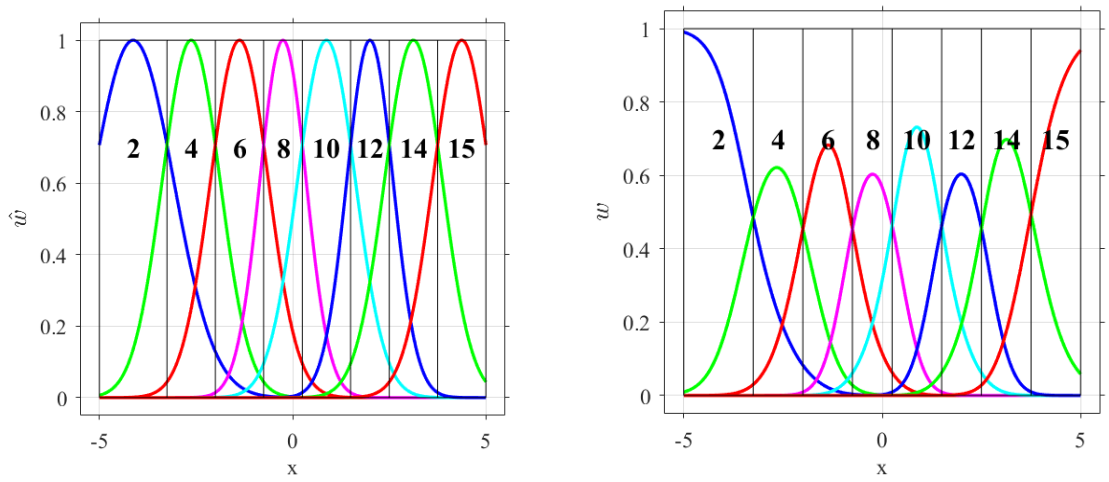


Figure 4.25: Global smoothed model fit for quadratic test case with smoothness factor $\lambda_s = 1.5$.



(a) Validity functions

(b) Normalized validity functions

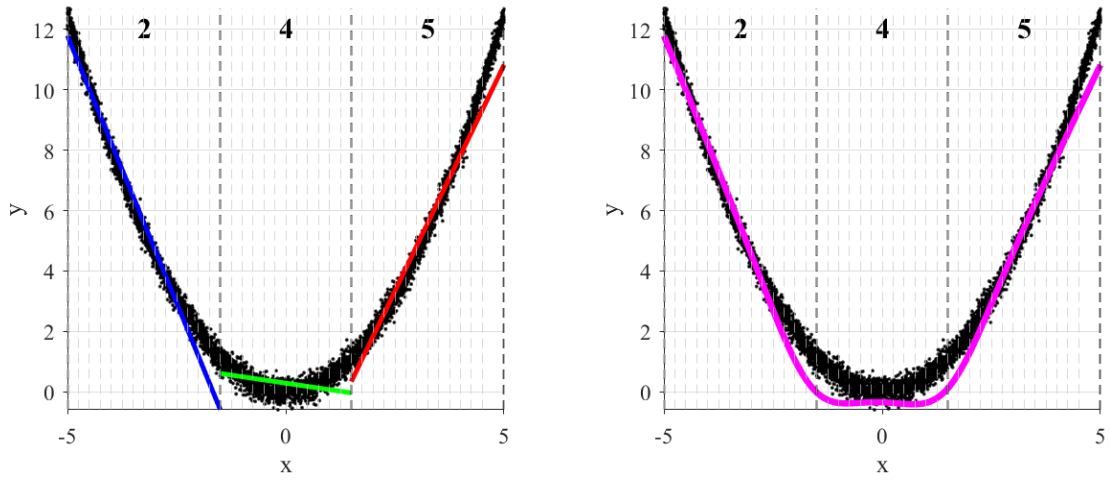
Figure 4.26: Validity functions for quadratic test case with smoothness factor $\lambda_s = 1.5$.

4.2.4 Residual threshold factor

The next example demonstrates the impact of changing the residual threshold factor, λ_ν. For quadratic test data generated using the same specifications as in

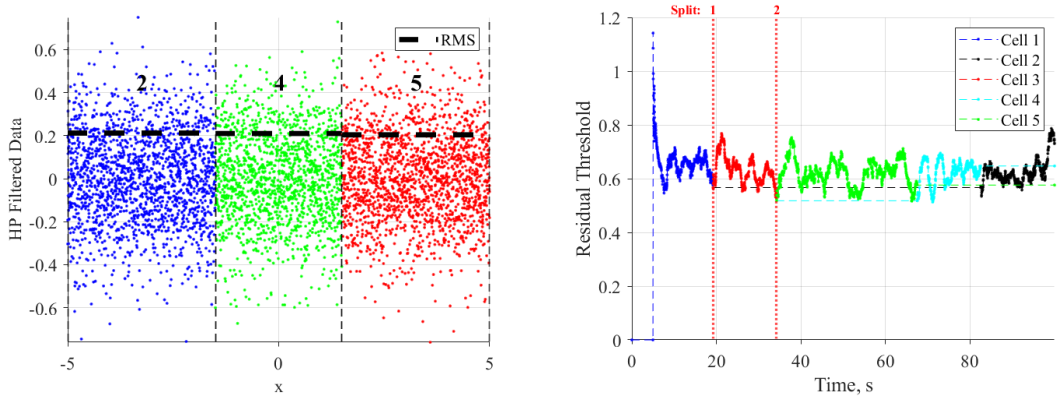
the baseline case, λ_ν was raised from 2 to 3, and the resulting local and global model fits are shown in Fig. 4.27. Only 2 splits were performed, resulting in an altogether poor model fit. Despite the RMS of the noise in this case in Fig. 4.28(a) exhibiting similar character to the baseline quadratic test case shown in Fig. 4.16(a), the residual threshold for this case in Fig. 4.28(b) is larger than that shown for the baseline case in Fig. 4.16(b). As a result, the residual characterization in Fig. 4.29 shows residuals characterized as acceptable for a wider range of nonlinear behavior, thus resulting in fewer splits allowed.

The residual threshold factor can be considered the most impactful SPLITR algorithm user specification due to its direct influence on the residual threshold, and therefore on the split determinations. If λ_ν is too small, then excessive splits may result, there may not be sufficient data in each cell to accurately estimate the parameters, and the resulting global smoothed model may be overly choppy. If λ_ν is too large, then necessary splits may not be recognized. For cases with known Gaussian white noise such as in this chapter, $\lambda_\nu = 2$, i.e. 2σ is shown to provide a good tradeoff between these extremes. For cases where the noise content is unknown, and/or contains nonuniform frequency distribution and other unmodeled influencing factors such as what is typically found in experimental data, λ_ν must be larger. For the experimental flight test data discussed in the next chapter, $\lambda_\nu = 4$ was used. This was chosen based on studying the NASA E1 flight data, and was also applied to flight data from another aircraft. Future work may automatically adjust λ_ν based on further analysis of the noise content.



(a) Local model fit (b) Global smoothed model fit

Figure 4.27: Local and global model fits for quadratic model with residual threshold factor variation.



(a) High-pass filtered modeling data (b) Residual threshold

Figure 4.28: Residual threshold characteristics for quadratic model with residual threshold factor variation.

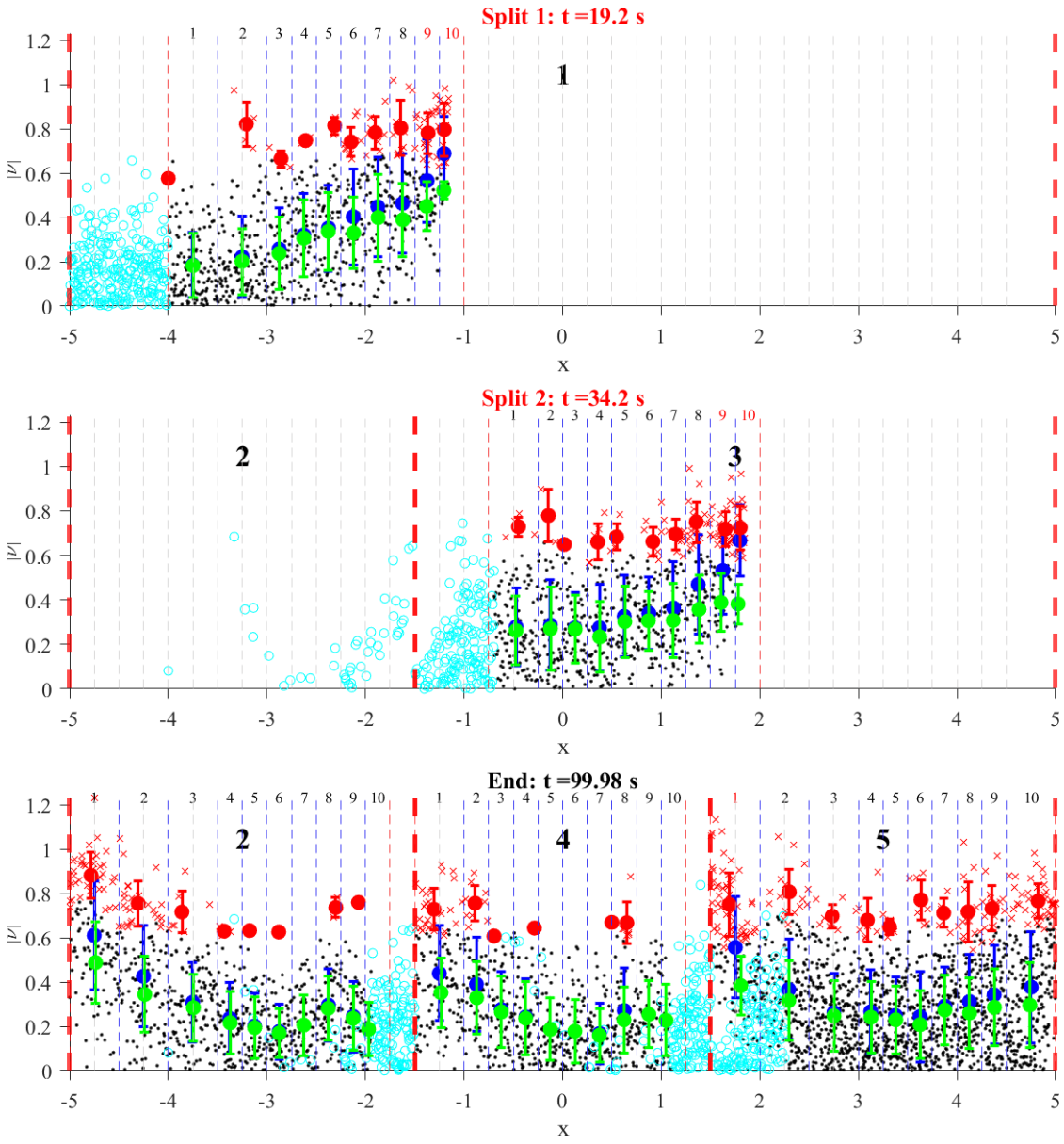


Figure 4.29: Binned local residual characterization for quadratic model with residual threshold factor variation.

4.2.5 Varying noise magnitude

The next test case shows the sensitivity of the SPLITR results to varying magnitude Gaussian white noise in the modeling data that is a function of y. This scenario, while undesirable from a statistical standpoint, is very common in practice, such that the noise magnitude is a function of one or more variables. In aerodynamic modeling, for example, there may be increased noise at higher angles of attack where there are unsteady effects and additional structural vibrations. Since the noise content estimates are customized for each cell in the SPLITR model development, variations in noise magnitude can be accounted for in a localized manner.

In this example, Gaussian white noise was added to the data with the magnitude proportional to y. The time history of modeling data is shown in Fig. 4.30, with 2 passes through the x-data. Note that the splits only occur during the periods of data with reduced noise. The cell structure development is shown in Fig. 4.31, and the local and global model fits in Fig. 4.32. Although there is a significant increase in noise at larger y values, there are in fact fewer splits in those regions as in the less noisy area of $y < 4$ because the residual threshold was adjusted accordingly in an automated way. The plot of the high-pass filtered modeling data in Fig. 4.33(a) shows the proportional noise content, and Fig. 4.33(b) shows the resulting residual threshold that varies significantly in each cell. As a result, a residual that is characterized as unacceptable in cell 6, for example, may be considered acceptable in cell 2, resulting in fewer splits allowed in the regions with higher magnitude noise to prevent overfitting.

Note also that since there is a relatively small RMS window of 2 seconds, the estimate of noise content even in a single cell changes over time. At around $t = 35$ seconds in Fig. 4.30, the final split is performed and the data obtained for the next 30 seconds or so lie in cell 9. However, Fig. 4.33(a) shows that the noise magnitude in cell 9 in fact varies significantly across the range of x. This is reflected in the residual threshold in Fig. 4.33(b) where initially the threshold is smaller, and then increases around $t = 50$ seconds to reflect the largest magnitude noise around $x = 5$, and then reduces once again. A similar trend can be seen in the threshold for cells 1 and 2.

Finally, it is worth acknowledging that the global model fit in Fig. 4.32 is not particularly good around $x = 0$ compared to the local model fits. This is due to the validity functions, shown in Fig. 4.34, which cause the local models in the wider cells 2 and 9 to influence the global model output in the narrower cells toward the middle.

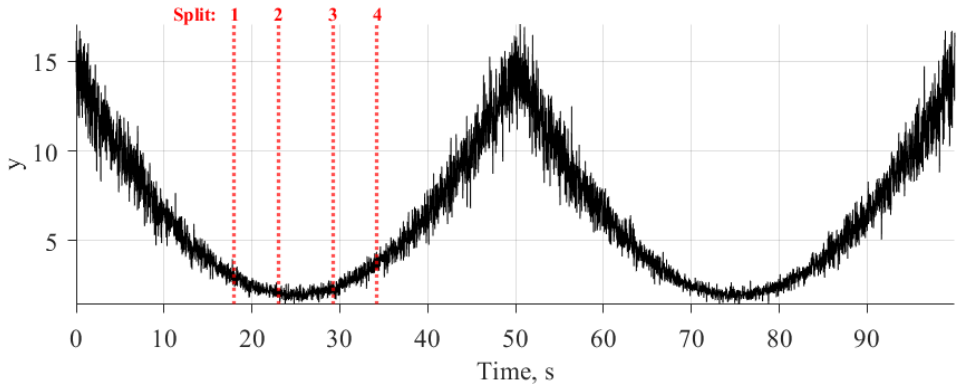


Figure 4.30: Time history of response variable for quadratic model with varying noise magnitude.

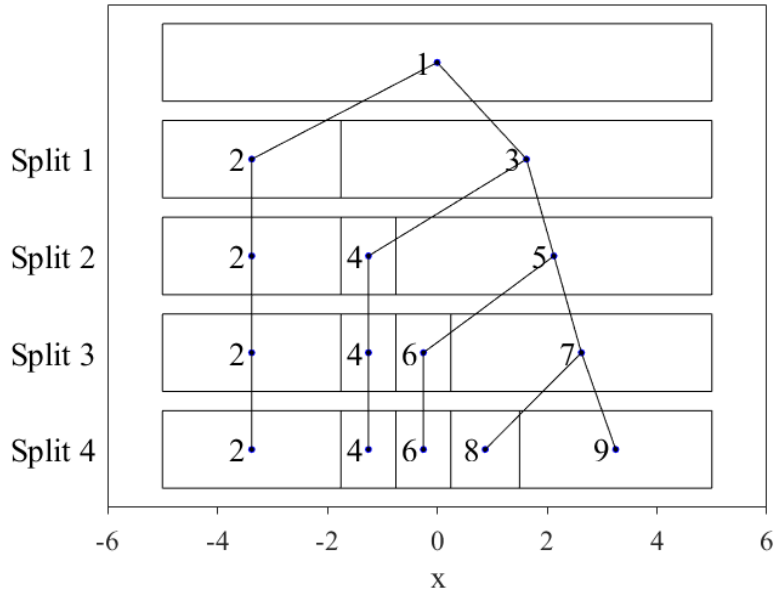
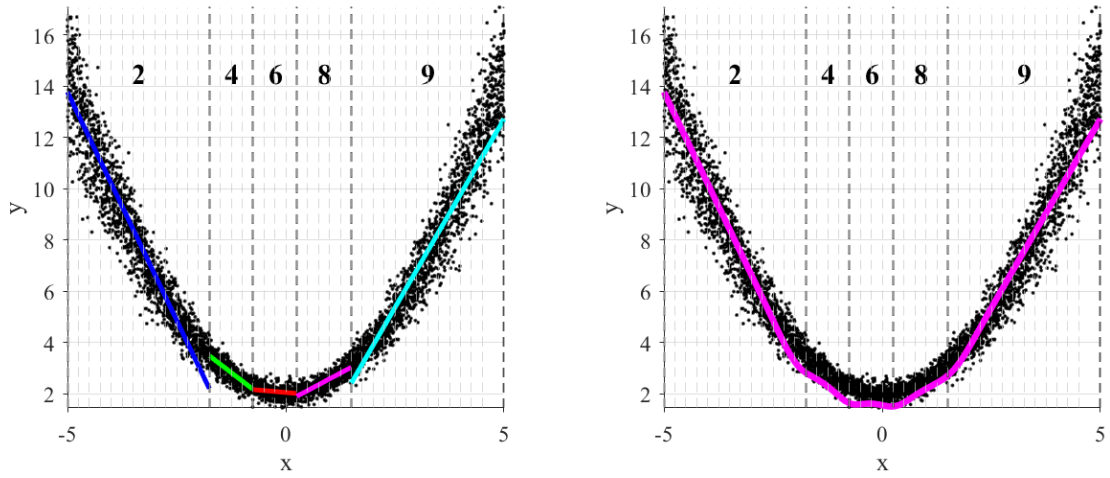


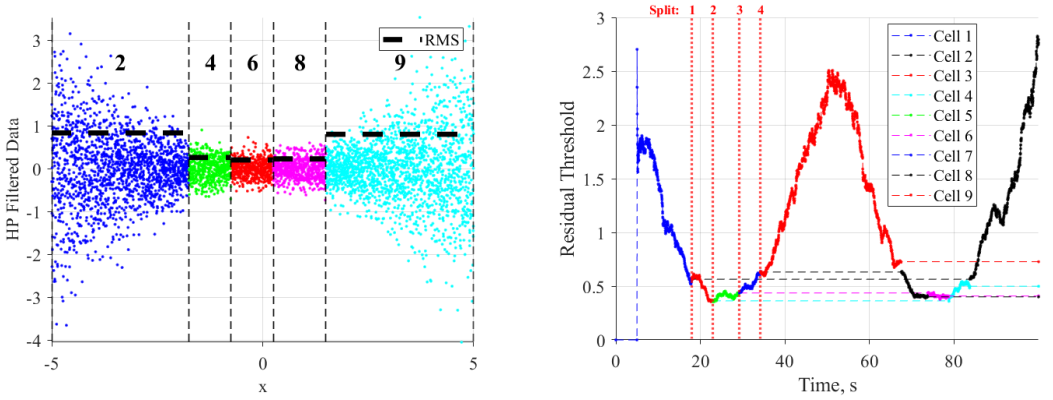
Figure 4.31: Cell structure evolution for quadratic model with varying noise magnitude.



(a) Local model fit

(b) Global smoothed model fit

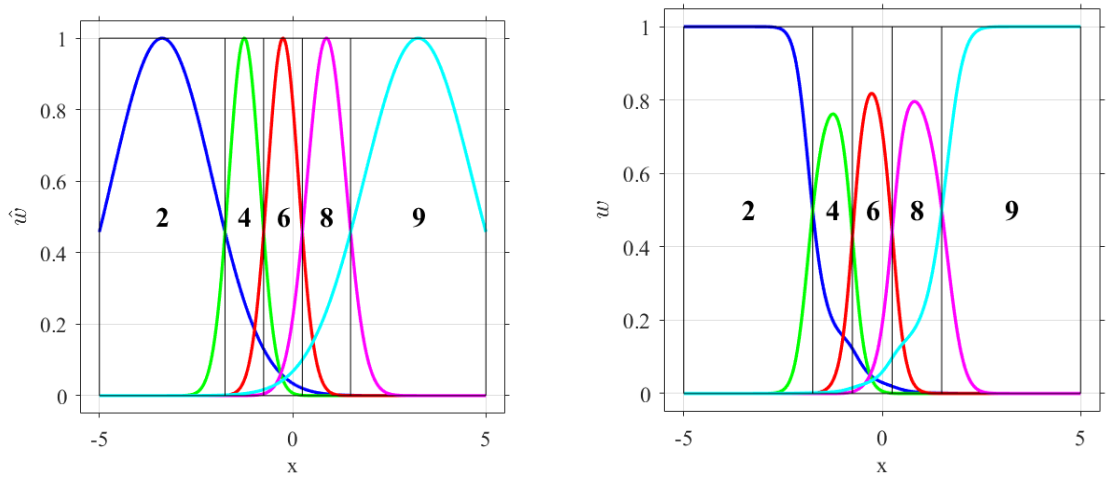
Figure 4.32: Local and global model fits for quadratic model with varying noise magnitude.



(a) High-pass filtered modeling data

(b) Residual threshold

Figure 4.33: Residual threshold characteristics for quadratic model with varying noise magnitude.



(a) Validity functions

(b) Normalized validity functions

Figure 4.34: Validity functions for quadratic model with varying noise magnitude.

4.2.6 Multiple simulations

The quadratic baseline case was run 25 times with constant magnitude Gaussian white noise added to the data to show the reliability of the SPLITR results across variations in the noise content of a given data set. The smoothed global

model fits for all cases are overlaid on all of the test data in Fig. 4.35, and show good quadratic fits with variations in the waves seen across the cell boundaries. Figure 4.36 shows the local slope parameter estimates for all test cases. Since there was a relatively small minimum cell width of 0.25 and there is no "true" breakpoint location for fitting a straight line to a curved one, each test case placed the local models at slightly different points based on the noise in each simulation. Nonetheless, the parameter estimates lie on or near the true values, and the groupings of colors show the clustering of split locations across the different simulations.

This example demonstrates the reliability of the SPLITR modeling results despite variations in the constant magnitude random noise added to the data. In particular, it shows that despite the split locations being chosen at slightly different points along the range of x, the slope estimate in each local cell was adjusted to model the data contained in that region. While this method focuses on obtaining sufficient *local* model estimates as shown in Fig. 4.36, it also demonstrates that the *global* smoothed models in Fig. 4.35 are able to consistently combine the piecewise models to obtain the quadratic shape.

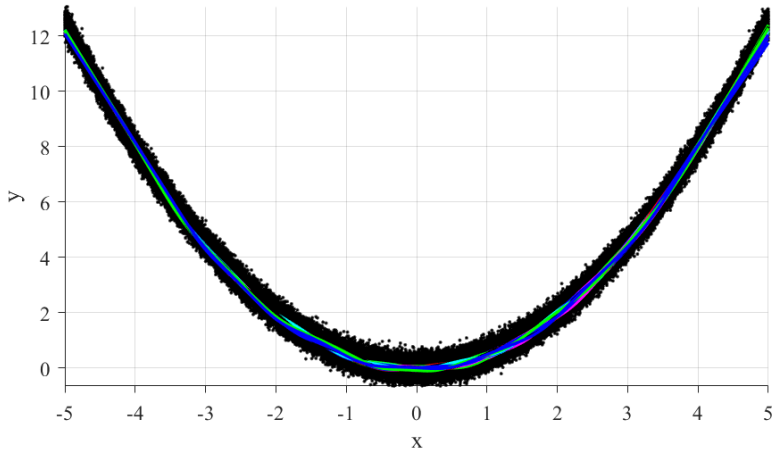


Figure 4.35: Global model fits for multiple simulations with quadratic data.

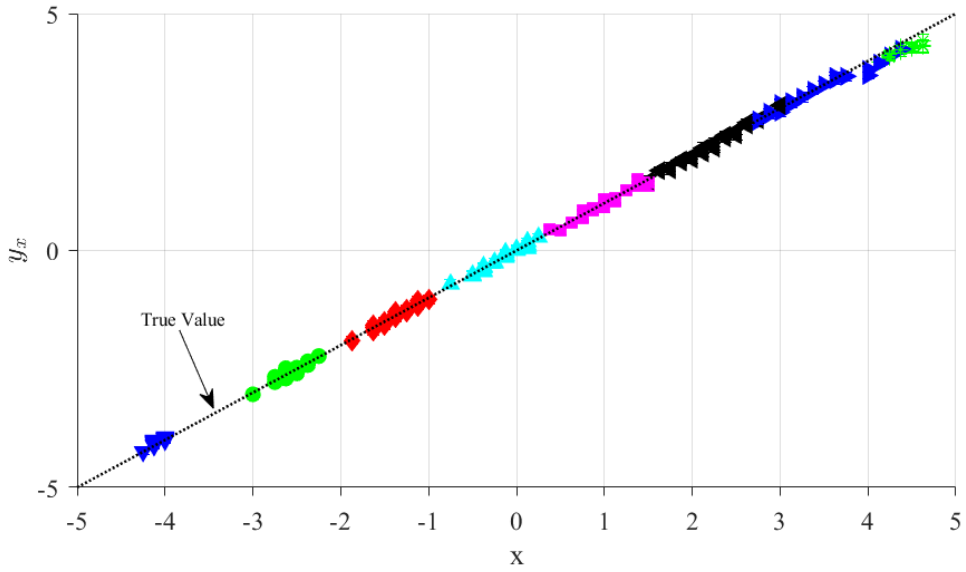


Figure 4.36: Local parameter estimates for multiple simulations with quadratic data.

4.2.7 Model adaptation

This example shows how the SPLITR model can adapt to a function that is modified during the model development. An advantage of the SPLITR model is that if the system were to adapt or sustain damage that would affect only a

small part of the input space, then only the relevant localized models would need to be adjusted, whereas the rest of the models could remain intact. This example, however, investigates a significant global model change, as if the system were to become unstable.

The initial baseline configuration is a quadratic model as given in Eq. (4.2). Data were generated with an SNR of around 18 in this case. Two passes of the x-data were completed from $x = -5$ to $x = 5$ with all of the baseline user specifications, and the resulting local and global models are shown in Fig. 4.37.

A sudden system adaptation is then simulated by generating the next set of data from a new function as shown in Eq. (4.3).

$$y = 25 - 0.5x^2 \tag{4.3}$$

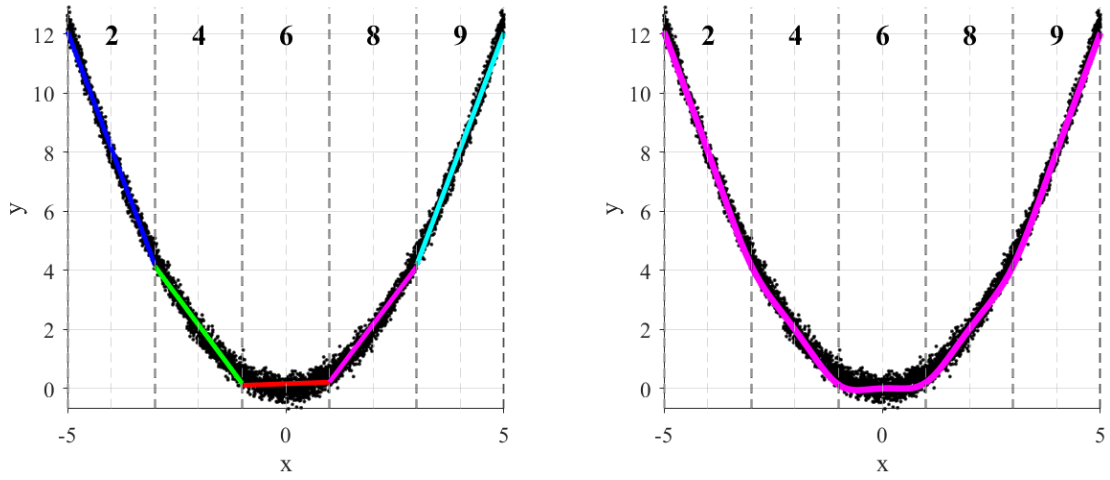
The only other change required during the adaptation phase was that the child cells are initialized with a much larger level of uncertainty to allow the models to adapt more quickly from the no longer relevant parent cell parameter initializations. Otherwise, it would require a significant amount of new data to overcome the certainty from the parent models. This modified specification of initializing the dispersion matrices with high uncertainty was performed throughout the entire simulation, not just during the adaptation.

Two additional passes of the x-data were then completed with data from the new quadratic function. The final local and global model fits are shown in Fig. 4.38, where the model effectively "forgot" about the initial model and fully adapted to the

new function. This was accomplished through the ability to add partitions where the existing model is not performing well. Note that splitting the model is the only tool that the SPLITR method has to use if the existing local model is not performing well. As a result, for cases of large adaptation such as in this example, it is difficult to balance the tradeoffs between forgetting too much prior information and adapting quickly to new information. For this reason, in this example the minimum cell width was set to 1, instead of 0.25 as it was for the previous quadratic cases. This was to ensure that the model is not overfit with excessive partitions that may occur during the adaptation due to the poor initial model fit, and that the cells are wide enough to obtain data with a high enough SNR. Furthermore, the initial resolution of 0.25 was to enable the model to choose the split locations from among a wider range of possibilities, but not intending to allow a split at every 0.25 increment. Nevertheless, when this simulation was run with a minimum width of 0.25, the results were still reasonable.

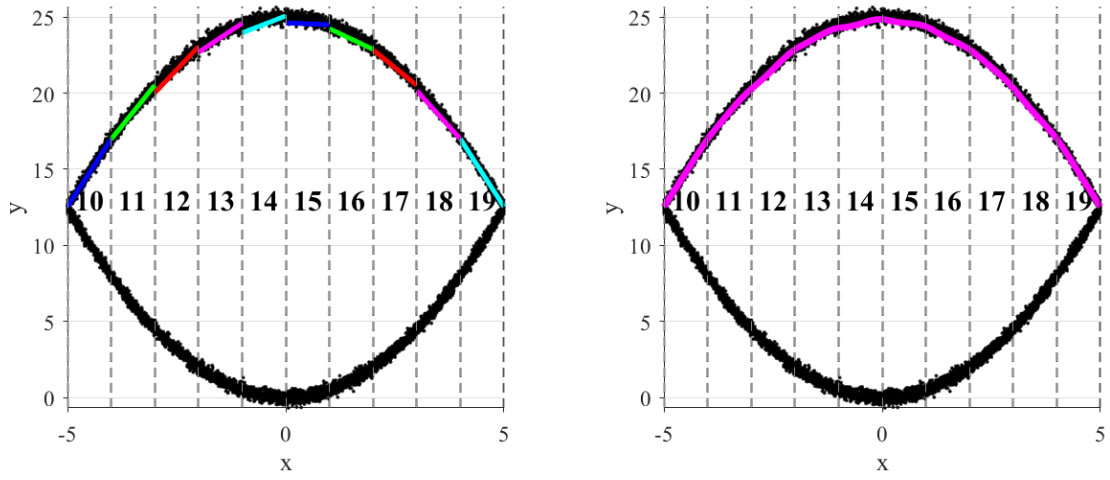
The cell structure evolution throughout the combined data is shown in Fig. 4.39, with the combined time history of modeling data in Fig. 4.40. Although the complexity of the quadratic function did not change for the new configuration, new cells were still added to capture the changes. If the adapted configuration was instead the baseline, only the 4 original cells would be needed for the SPLITR model. The additional splits as utilized in this example are meant to be a tool that can be used to hasten the adaptation. For the final cell structure, methods such as pruning may be appropriate in future work to effectively combine adjacent cells for which the parameters are not sufficiently different from each other, to obtain a simpler final

model. The SPLITR logic would need to be further developed to more effectively and autonomously detect a fault or model adaptation, and to adapt the modeling accordingly.



(a) Local model fit (b) Global smoothed model fit

Figure 4.37: Local and global model fits for quadratic model with model adaptation using the first set of data.



(a) Local model fit (b) Global smoothed model fit

Figure 4.38: Local and global model fits for quadratic model with model adaptation using the full set of data.

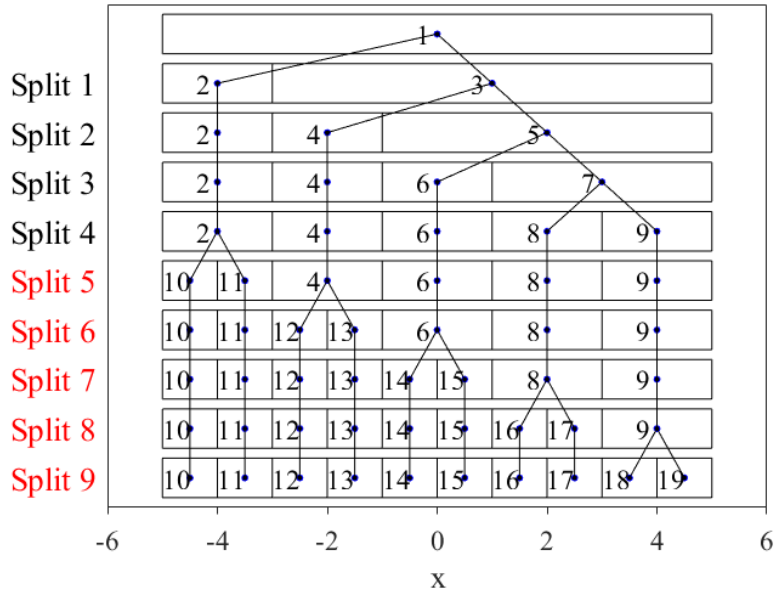


Figure 4.39: Cell structure evolution for quadratic model with model adaptation.

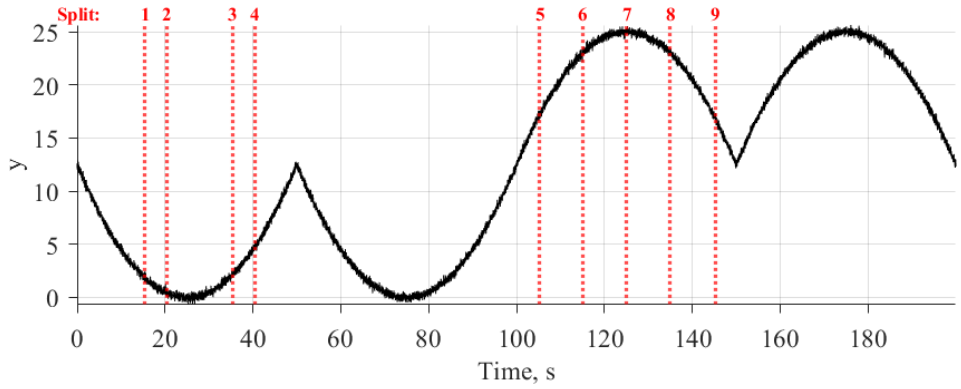


Figure 4.40: Time history of response variable for quadratic model with model adaptation.

4.3 1D cubic data

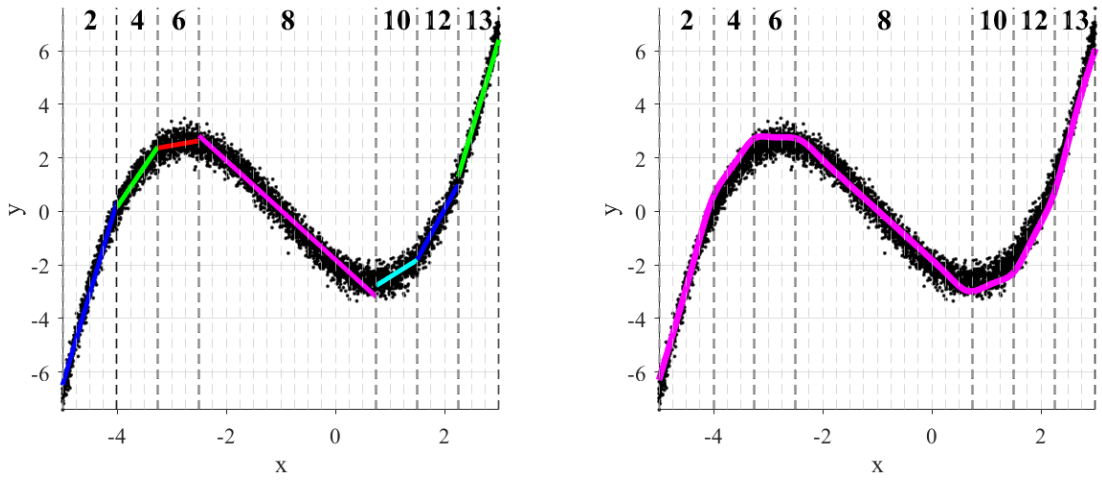
The final set of 1D test cases looks at data generated from a cubic function defined in Eq. (4.4) from $x = -5$ to $x = 3$.

$$y = 0.25(x^3 + 3x^2 - 6x - 8) \tag{4.4}$$

A baseline test case will be shown for data simulated with constant magnitude Gaussian white noise added to the data, and then a second example with varying magnitude noise. Data were generated with 2 passes of the input space and 2500 evenly spaced points across the range of x in each pass.

4.3.1 Baseline case

For the baseline cubic test case, Gaussian white noise with an RMS of 0.277 was added to the data, resulting in an SNR of 10. The final split locations and local and global models are shown in Fig. 4.41, with the high-pass filtered modeling data in Fig. 4.42. Aside from several sharp points in the smoothed model that result from the Gaussian global weighting, the SPLITR model captures this higher order data well with piecewise local linear models by offering increased resolution around the highly curved regions. As discussed in Section 4.2.3, additional smoothing for this higher order case could improve the global model fit and reduce the sharpness of the breakpoints.



(a) Local model fit

(b) Global smoothed model fit

Figure 4.41: Local and global model fits for cubic baseline case.

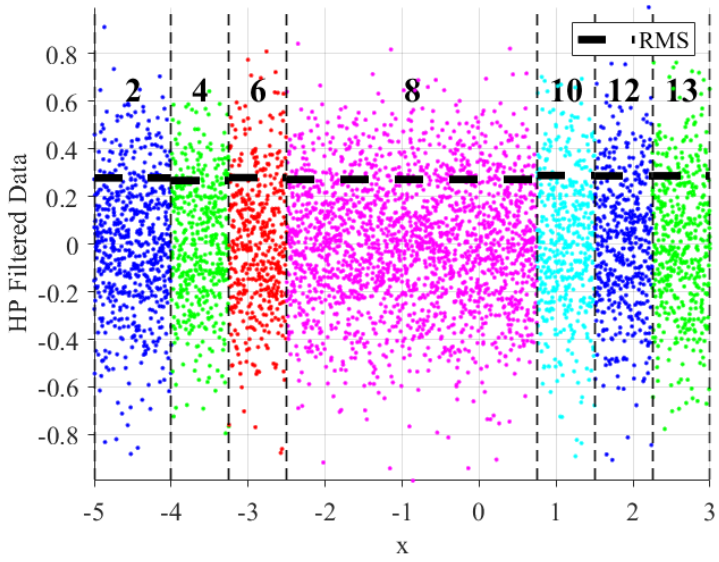
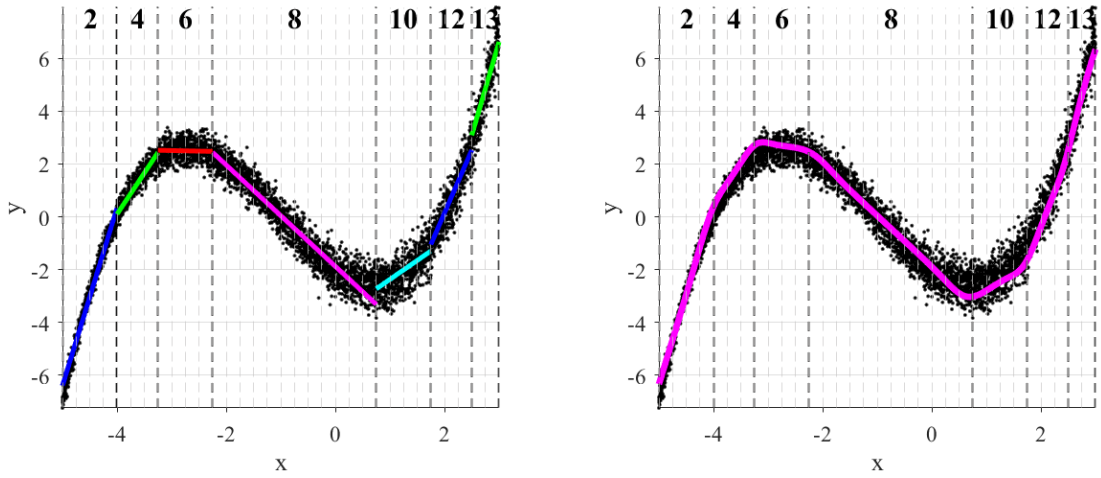


Figure 4.42: High-pass filtered modeling data for cubic baseline case.

4.3.2 Varying noise magnitude

In this example, the magnitude of the Gaussian white noise added to the data was varied linearly across x, with minimum magnitude at $x = -5$ and maximum

magnitude at $x = 3$. The resulting local and global model fits are shown in Fig. 4.43, with the high-pass filtered modeling data in Fig. 4.44. This example demonstrates that despite the varying noise magnitude compared to the previous case, and the fact that the residuals will be larger toward higher x in this case, the resulting split locations and models are very similar, indicating the insensitivity of the splitting logic to the random noise variation. If the noise magnitude was significantly larger in a particular region, it is possible that splits may not occur there, as the residual threshold would be modified accordingly.



(a) Local model fit

(b) Global smoothed model fit

Figure 4.43: Local and global model fits for cubic test case with varying noise magnitude.

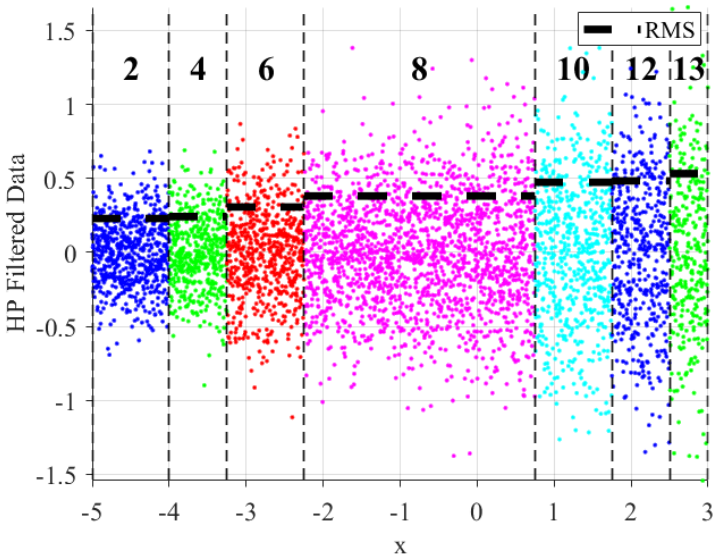


Figure 4.44: High-pass filtered modeling data for cubic test case with varying noise magnitude.

4.4 2D test data

This section shows SPLITR modeling results for test data that is a function of two dimensions, $z = f(x, y)$. The SPLITR splitting logic, as discussed in Chapter 3, stores the residual information across each specified PV dimension, and can therefore choose both the relevant split location *and* dimension based on the residual evidence contained in the bins. This will be demonstrated through three test cases presented.

4.4.1 2D quadratic data, case 1

This test case uses data generated from the function in Eq. (4.5), and looks at a quadratic contribution from only the y dimension.

$$z = x - (y + 20)^2 \tag{4.5}$$

The input space was defined over $-100 \leq x \leq 0$, and $-100 \leq y \leq 0$. There were 41 evenly-spaced x and y data points generated, and then a mesh was formed, resulting in 1681 points throughout the two-dimensional input space. These points were organized into an array to traverse the input space by holding the y-dimension constant and traveling back and forth across the range of x-data, with the resulting time history shown in Fig. 4.45. Given the relatively small increment of EV data, a second pass through the input space to update the parameters was not performed, nor necessary. Only a very small amount of Gaussian white noise was added to the output data in this case, which resulted in a large number of splits because the residual threshold was relatively low. Note that since the SPLITR model is developed in real time as the data are received, the manner in which the input space is traversed may have a significant impact on the resulting split locations, and by extension, the model fit. The baseline residual threshold factor of 2 was used for this case.

Although allowable PVs included both x and y, the SPLITR model resulted in several splits across only the y-dimension, as shown in Fig. 4.46. This is consistent with the presence of nonlinear behavior in the output data as a function of y. Figure 4.47 shows the modeling data along with the SPLITR model fit, as well as the split locations that divide the quadratic y dependency into many cells. Finally, Fig. 4.48 shows validation data at $x = -100$, as well as the SPLITR model fit and split

locations. With $R^2 \approx 1$, the SPLITR model captures the data extremely well for this test case.

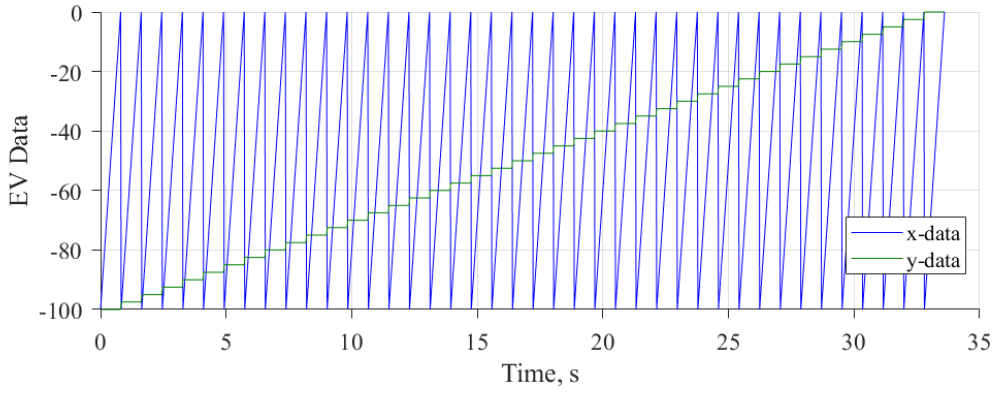


Figure 4.45: Time history of input data for 2D quadratic case 1.

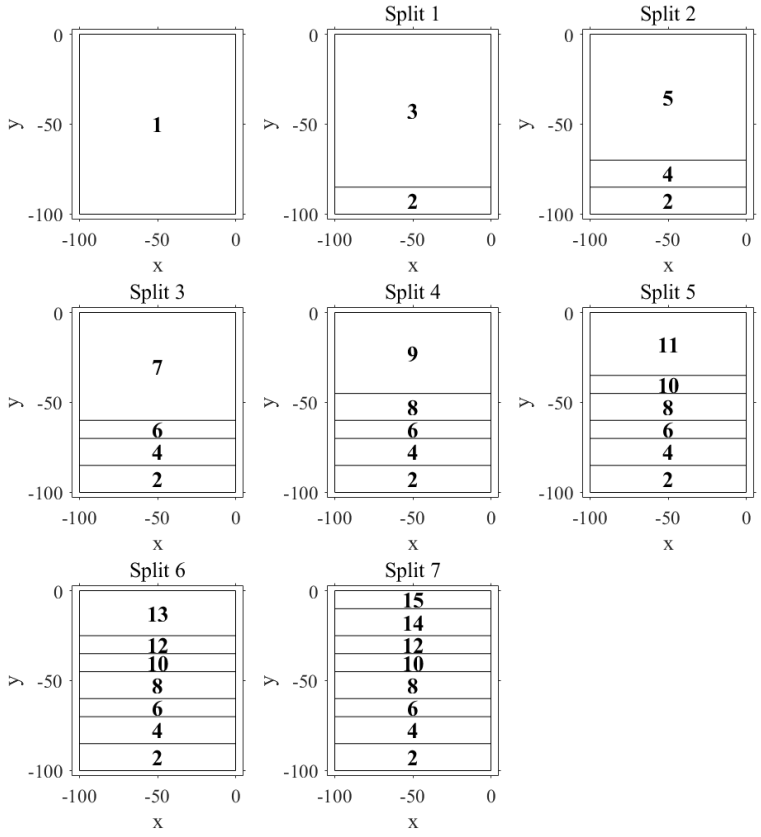


Figure 4.46: Cell structure evolution for 2D quadratic case 1.

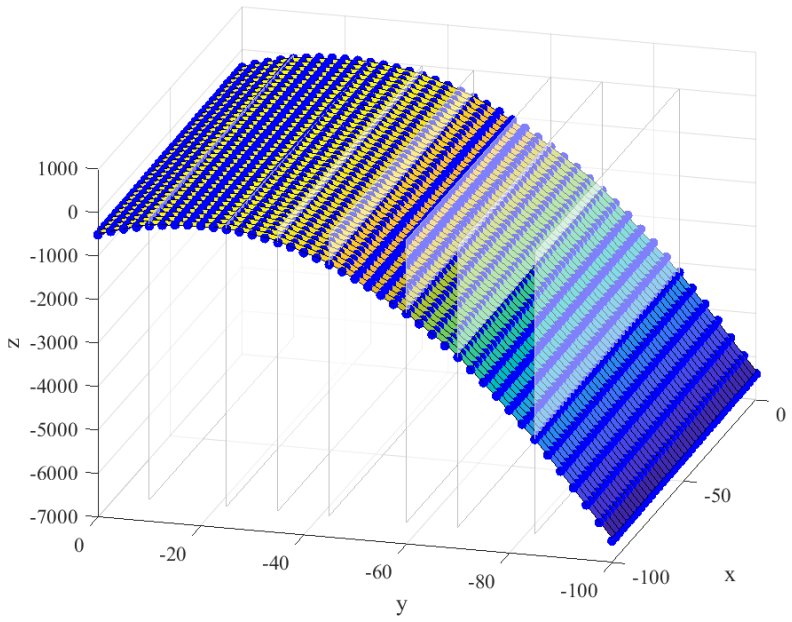


Figure 4.47: Global model fit for 2D quadratic case 1.

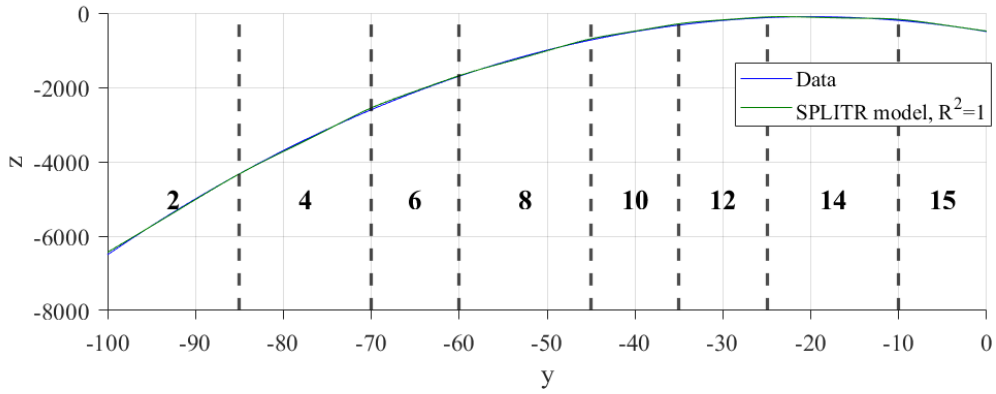


Figure 4.48: Validation test data and model fit for 2D quadratic case 1.

4.4.2 2D quadratic data, case 2

This second 2D test case looks at test data from the function in Eq. (4.6), with $-5 \leq x \leq 5$, and $-5 \leq y \leq 5$. This case looks at modeling the interaction effect between x and y.

$$z = xy - 10x \tag{4.6}$$

Gaussian white noise was added to the output data, with standard deviation equal to two-percent of the RMS of the response variable. The input space of 1872 data points at 0.5 increments was covered twice in both a forward and backward pass as shown in the time history in Fig. 4.49. In contrast to the previous example, in this case the input space was traversed in a box-like formation as shown in Fig. 4.50. The outer box was first drawn by holding y constant and covering the x range along the bottom of the plot, and then proceeding up the right side, across the top, and down the left side. The remaining boxes are then filled in, beginning with the bottom left corner.

The cell structure evolution is depicted in Fig. 4.51, which shows each cell structure from the initial configuration through the fourth split. The final cell structure has 3 splits in the x-dimension and 1 in the y-dimension.

The SPLITR model is pictured in Fig. 4.52, showing the partition planes, model surface, and modeling data. Note that there are some waves in the surface that result from the Gaussian weighting, and that the fit is poorer at the edges of the input space. The model fit can also be more easily visualized in Fig. 4.53 which shows the modeling data and model output, with $R^2 = 0.996$.

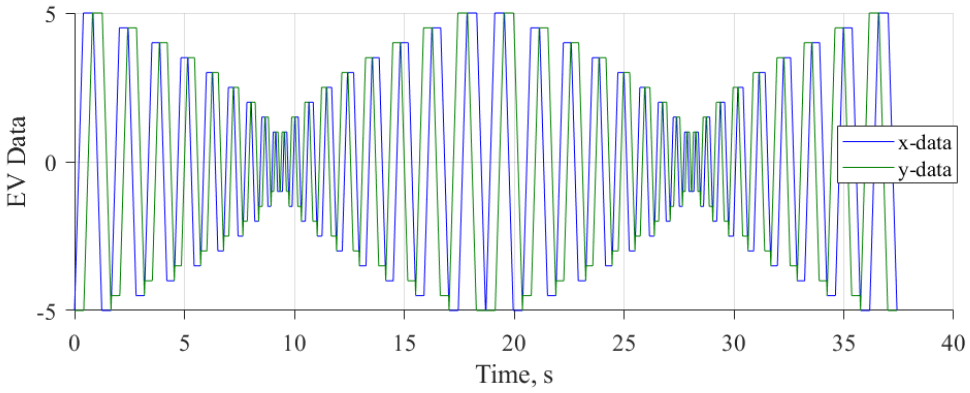


Figure 4.49: Time history of input data for 2D quadratic case 2.

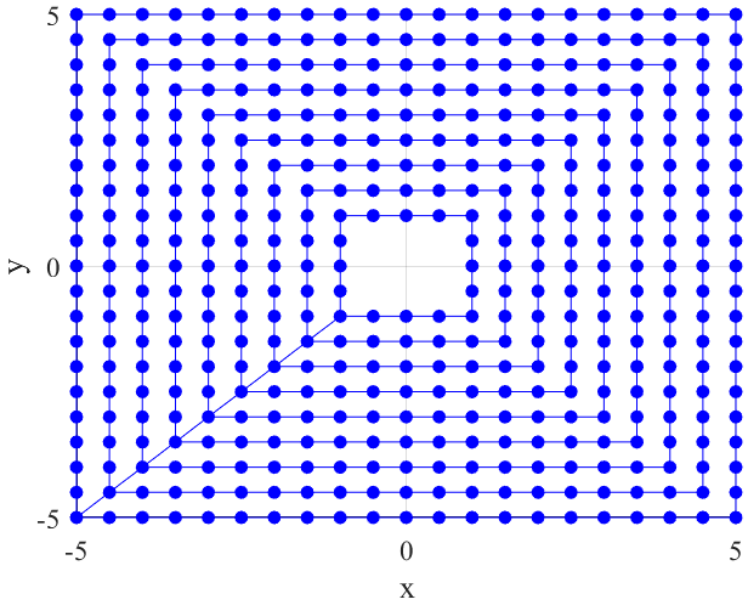


Figure 4.50: Test data for 2D quadratic case 2.

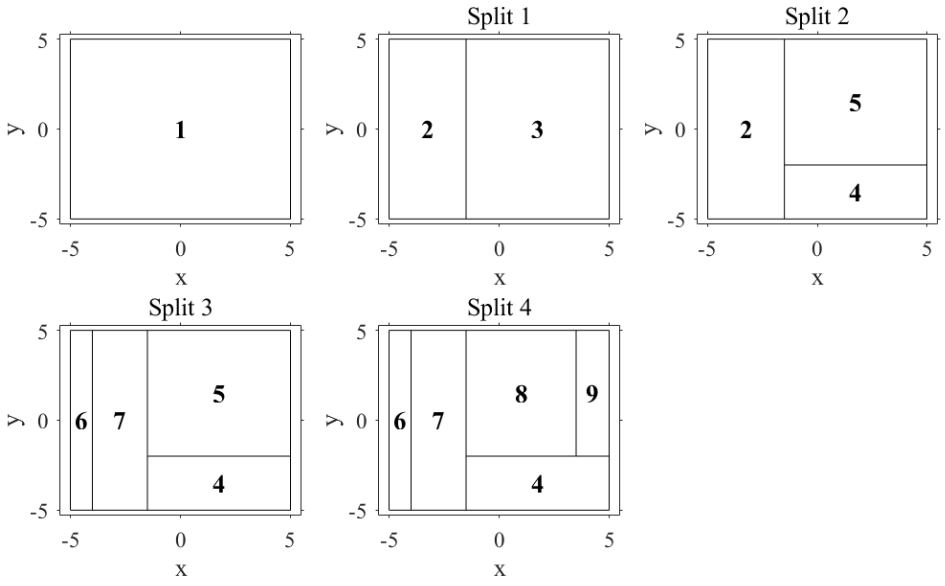


Figure 4.51: Cell structure evolution for 2D quadratic case 2.

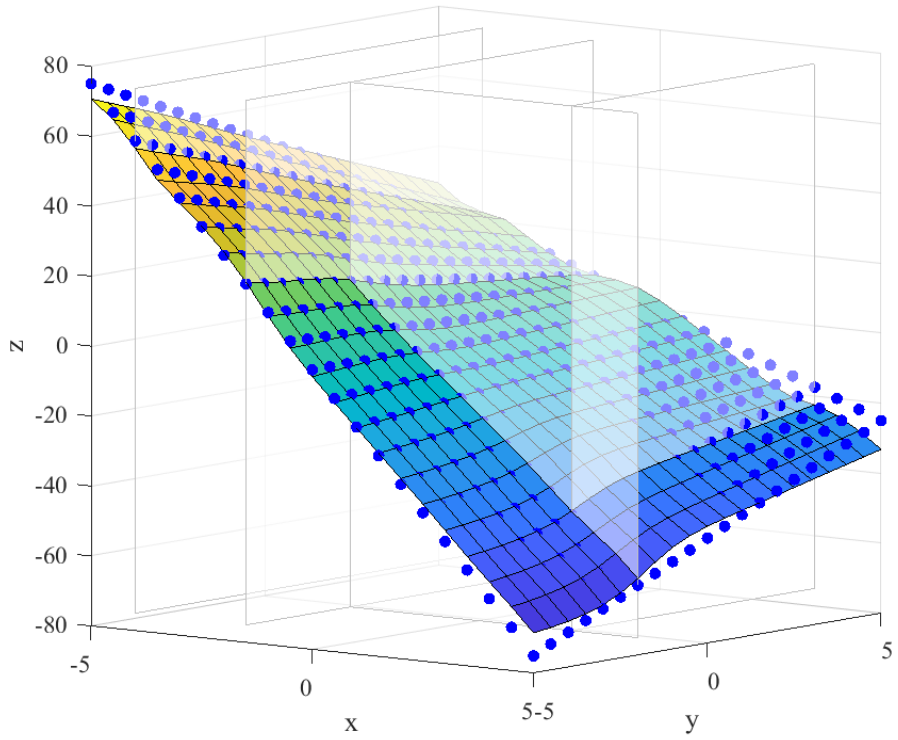


Figure 4.52: Modeling data and SPLITR model fit for 2D quadratic case 2.

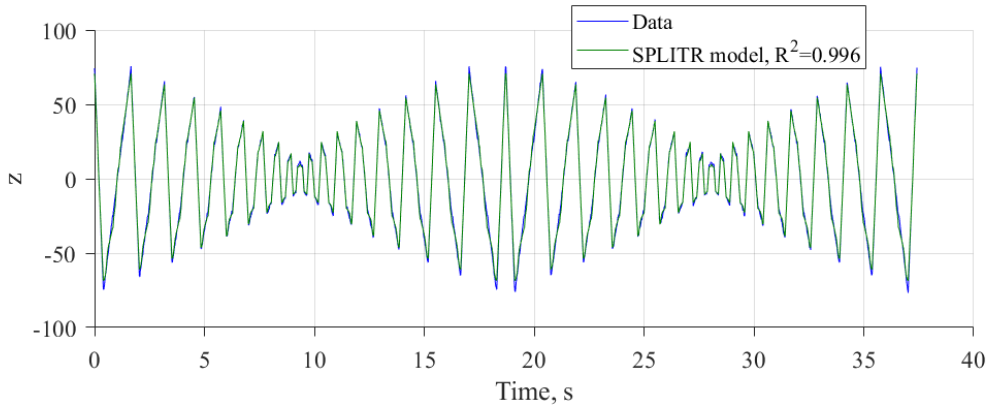


Figure 4.53: Test data and model fit for 2D quadratic case 2.

4.4.3 2D quadratic data, case 3

This final 2D quadratic test case builds a model using data generated from the function in Eq. (4.7), with $-5 \leq x \leq 5$, and $-5 \leq y \leq 5$. This case looks at a quadratic influence in the x dimension, as well as an interaction effect.

$$z = xy - 0.05x^2 \tag{4.7}$$

Two-percent Gaussian white noise was added to the output data, similar to the previous case. Due to the large amount of curvature in this case, the input space of 1872 data points at 0.5 increments was covered eight times through a combination of forward and backward passes as shown in the time history in Fig. 4.54. The input space was traversed in the same box-like formation as in the previous example.

This example is a case where the modeling data show significant curvature, and a piecewise linear composition of highly nonlinear data results in a large number of splits, as shown in Fig. 4.55. The model fit using validation data spanning the full

range of the input space is presented in Fig. 4.56, which shows the highly nonlinear nature of the modeling data across the y-dimension, the resulting model surface, and the split locations. The waves in the model surface associated with the global weighting are also apparent, as well as a relatively poor fit at the corners. Finally, Fig. 4.57 shows the model fit across only 2 passes of the input space, with a total R^2 of 0.99. It is clear there too that the model fit is poorest at the peaks, or edges.

While axes-oblique partitioning methods might be more conducive to modeling highly nonlinear functions, this example demonstrates that a large number of axes-orthogonal partitions can still provide a reasonable model fit.

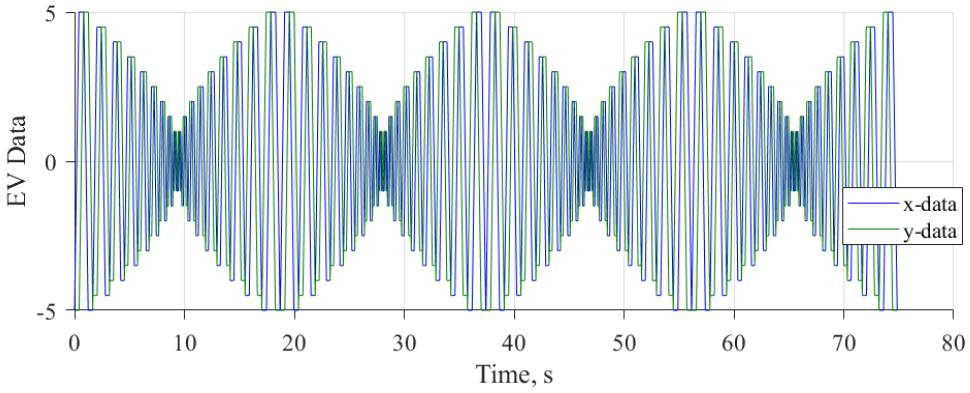


Figure 4.54: Time history of input data for 2D quadratic case 3.

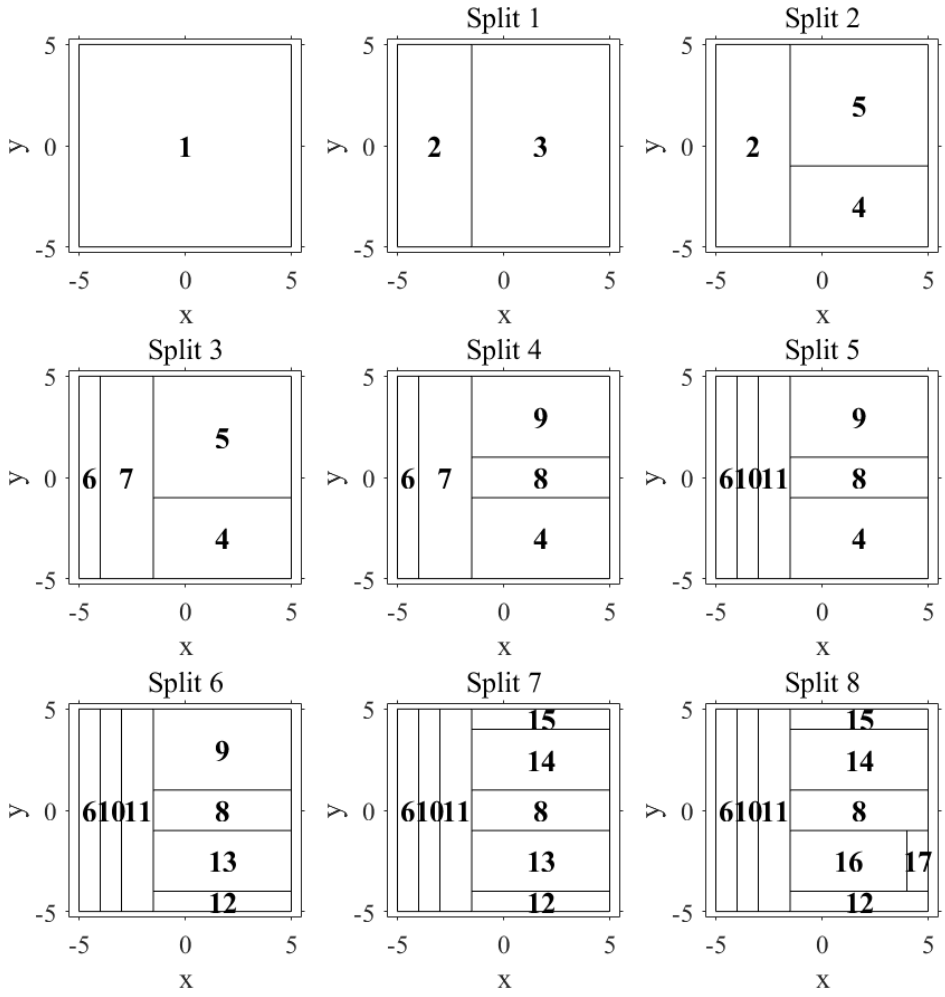


Figure 4.55: Cell structure evolution for 2D quadratic case 3.

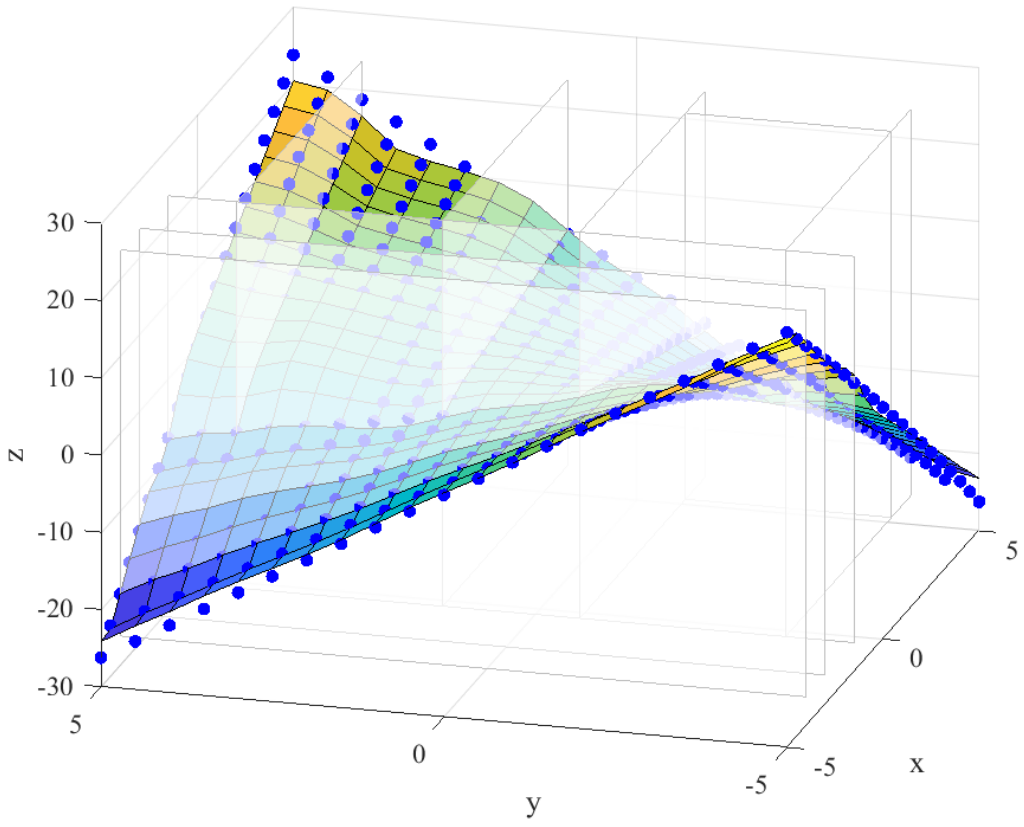


Figure 4.56: Global model fit with validation data for 2D quadratic case 3.

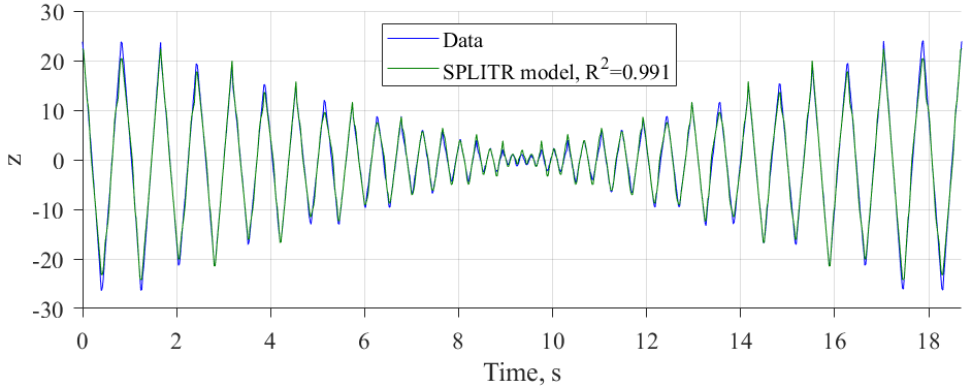


Figure 4.57: Validation test data and model fit for 2D quadratic case 3.

4.5 Summary

While Chapter 3 presented the theory behind the novel SPLITR modeling approach, Chapter 4 provided an in-depth study and illustration through a series of simple examples that both described the algorithm logic, and explored the sensitivity to specified parameters or data characteristics.

The 1D piecewise linear examples offered straightforward data sets to describe in detail the model development process. The minimum cell width parameter was shown to impact the precise allowable split locations. In practice, this means that the user-specified minimum cell width must offer sufficient resolution to capture discrete changes in the model at particular locations. For the piecewise linear test data, this could be extremely important. However, most systems to be modeled exhibit higher order behavior, and the ensuing examples showed that the precise location of the splits was less important, as there are many possible breakpoints that could still produce a sufficient global model. The SPLITR method robustness to high-magnitude Gaussian white noise was also demonstrated.

The 1D quadratic test cases showed the SPLITR modeling approach effectively applied to higher order data. The filter window size was shown to play a role in the convergence of the residual threshold, but not in the resulting model fit. The smoothness parameter λ_s influenced the overlap between the local models, while the residual threshold factor λ_ν directly impacted the residual characterization, and therefore the tendency to split. The SPLITR method's robustness to varying magnitude Gaussian white noise was also demonstrated. Consistent acceptable global

model fits were shown for multiple simulations with random noise. A final 1D quadratic test case demonstrated how the SPLITR model can adapt to a modified function, which can be useful for an adaptive system with changing behavior.

The 1D cubic data examples were used to show results for a function with increased curvature compared to the quadratic cases, and the SPLITR models were able to capture the data with similar results when both constant magnitude and varying magnitude Gaussian white noise were added.

Finally, results were shown with 2D quadratic test data to demonstrate how the SPLITR method can be used to correctly identify the single dimension with nonlinear dependencies, as well as offer 2D splits when the nonlinear function requires it. The first test case showed 1D splits in quadratic data where the nonlinear dependencies were restricted to one dimension. The second case showed a more complex 2D example using data generated from a function with an interaction term and a linear term, and the final example showed data from a function with an interaction term and a quadratic term. These cases resulted in a large number of axes-orthogonal splits, and model fits that were altogether acceptable, yet with notable deficiencies in certain areas of the input space.

Each test case presented in this chapter was devoted to showcasing how SPLITR can be used to model data with specified dependencies, such as linear, quadratic, cubic, and interaction effects. Nonlinear polynomial models for physical systems tend to include a combination of one or more of those sets of modeling terms. Collectively, these examples show how SPLITR could be used to capture the nonlinearities associated with the higher order and multivariate terms by partitioning the input

space into smaller components that can be approximated as linear. This simple representation can be applied even to complex functions, although the number of cells required may be large.

With these example test cases offering clarity and insight into the SPLITR method, the next chapter will explore applying SPLITR to more complex physical systems for aerodynamic modeling. Applications include using simulated test data of the F-16, and experimental data from the NASA E1 and T-2 aircraft.

Chapter 5

SPLITR Applied to Aerodynamic Modeling with Simulated and Experimental Flight Data

After the SPLITR modeling approach was introduced in Chapter 3, Chapter 4 demonstrated how the method operates, as well as the sensitivity of the results to various input parameters, through a series of test cases using generic simulated test data. It also showed how higher order nonlinear dependencies and interaction terms could be captured with several local linear models. While the SPLITR method is practical for any data set, it was motivated by, and developed within, the context of aerodynamic modeling for the NASA L2F concept. This chapter will therefore expand the test case complexity from that which was presented in Chapter 4 to applied examples of aerodynamic modeling using both simulated and experimental flight data.

First, the modeling process generally discussed in Chapters 2 and 3 will be

specifically applied to aerodynamic modeling for the aircraft applications discussed in this chapter. Then an F-16 simulation will be introduced, and SPLITR modeling results will be shown using simulated flight data. The NASA E1 and T-2 Generic Transport Model (GTM) test aircraft will then be described, and the SPLITR results will be presented for the experimental flight test data obtained from each of these test vehicles. These conventional aircraft will be useful to allow for interpretation and validation of the results together with experience-based expectations. Each example highlights various practical aspects and benefits of the SPLITR modeling capabilities, while collectively the test cases described in this chapter showcase the utility and versatility of the SPLITR modeling method for experimental applications. In particular, this chapter demonstrates how SPLITR can be practically, easily, and successfully used to develop aerodynamic models using real experimental data sets that have low SNRs.

5.1 Aerodynamic modeling problem setup

The least-squares parameter estimation process used in this work is described in Sections 2.3.4 and 3.2.4. Up until this point, the parameter estimation has been presented in a generalized way that is agnostic to a particular function or system to be modeled. This section shows how the response variables are calculated for the aircraft modeling examples in this chapter.

For conventional fixed-wing aerodynamic modeling, the nondimensional aerodynamic force and moment coefficients are calculated using the following equations,

which are defined in the aircraft body axes and retain the full nonlinear aircraft motion, while ignoring propulsion gyroscopic terms [21].

$$C_X = \frac{ma_x - T}{\bar{q}S} \tag{5.1}$$

$$C_Y = \frac{ma_y}{\bar{q}S} \tag{5.2}$$

$$C_Z = \frac{ma_z}{\bar{q}S} \tag{5.3}$$

$$C_l = \frac{I_x}{\bar{q}Sb}\left[\dot{p} - \frac{I_{xz}}{I_x}(pq + \dot{r}) + \frac{(I_z - I_y)}{I_x}qr\right] \tag{5.4}$$

$$C_m = \frac{I_y}{\bar{q}S\bar{c}}\left[\dot{q} + \frac{(I_x - I_z)}{I_y}pr + \frac{I_{xz}}{I_y}(p^2 - r^2)\right] \tag{5.5}$$

$$C_n = \frac{I_z}{\bar{q}Sb}\left[\dot{r} - \frac{I_{xz}}{I_z}(\dot{p} - qr) + \frac{(I_y - I_x)}{I_z}pq\right] \tag{5.6}$$

The force equations for C_X and C_Z can alternatively be expressed in the wind axes as

$$C_L = -C_Z \cos\alpha + C_X \sin\alpha \tag{5.7}$$

$$C_D = -C_X \cos\alpha - C_Z \sin\alpha \tag{5.8}$$

These dependent variables can be calculated in flight using body-axis measurements of angular rates and translational accelerations, along with dynamic pressure measurements and geometry and mass properties. For the E1 flight test data in this chapter, a smoothed local differentiation method was used to compute the angular accelerations in flight using a fixed-lag smoother with a two time sample delay [14, 21]. When the thrust in Eq. (5.1) is not measured, C_X is commonly estimated to include the lumped body x-force as well as the thrust force, as was done in this work [14]. Note that Eqs. (5.1 – 5.6) are the typical representation for rigid-body fixed-wing aerodynamics, but some of the modeling considerations in Section 1.2.1 could require modified representations. For example, in rotorcraft system identification the aerodynamic models are more often expressed using body-axis dimensional forces and moments, so Eqs. (5.1 – 5.6) are modified as in Eq. (1.3). Or if there are significant and coupled aeroelastic properties, then additional sensor data would be required to estimate both the aerodynamic and structural modes.

The response variables are expressed in terms of explanatory variables measured in flight, such as air flow angles (α, β), body-axis angular rates (p, q, r), control surface deflections (δ), and an explanatory variable for thrust, such as advance ratio (J). Section 3.2.1 describes in more detail the model variables, both the EVs and the PVs, for aerodynamic modeling applications.

With the emphasis on model interpretability in this work, the estimated parameters can be considered as stability and control derivatives that describe the local aerodynamics in each cell, and if each cell has the same model structure, these influence coefficients can be compared across the global flight envelope.

5.2 F-16 simulation data

This section demonstrates the SPLITR modeling approach through a simplified aircraft example using data from an F-16 simulation found in SIDPAC [21], and draws together several sample plots used for clarification throughout Chapter 3. The nominal geometry and mass properties of the F-16 aircraft used in the nonlinear simulation are summarized in Table 5.1. The static and dynamic data used to build the aerodynamic database to support the simulation were obtained from wind tunnel tests. A detailed description of the simulation can be found in Ref. [21].

Symbol	Value	Unit
$\bar{c}$	11.32	ft
b	30	ft
S	300	ft^2
m	647.2	slug
I_x	9,496	slug-ft^2
I_y	55,814	slug-ft^2
I_z	63,100	slug-ft^2
I_{xz}	982	slug-ft^2

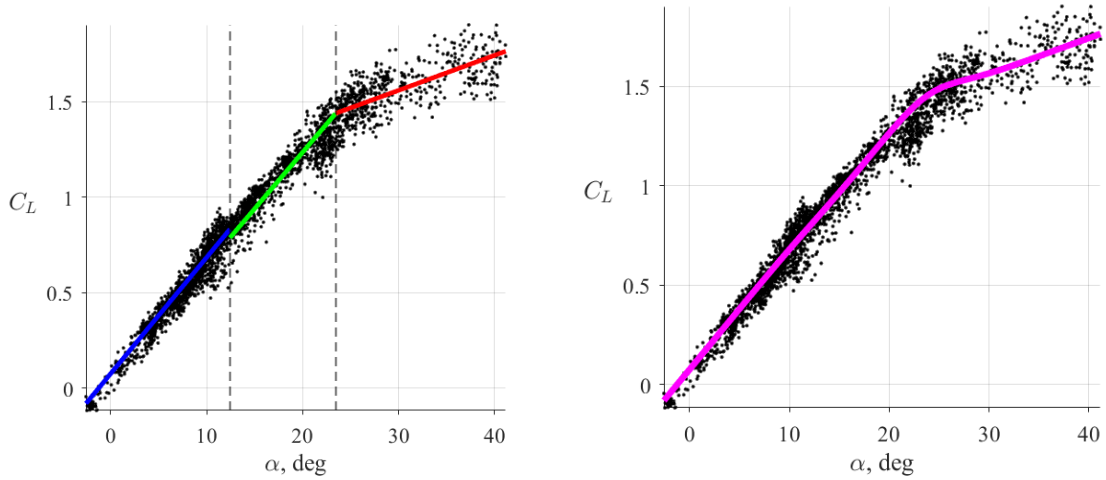
Table 5.1: F-16 geometry and mass properties.

The simulated flight data consisted of a maneuver that began at a trimmed angle of attack of 4 deg at 25,000 ft. The nominal elevator input was slowly varied along with simultaneous manual perturbations intended to excite the longitudinal dynamics across a wide range of α, as shown in Fig. 3.2. Effectively, two passes of the α space are performed across the duration of the maneuver. Gaussian white noise was added to the EVs with standard deviation set at two-percent of the magnitude in the data. As a result, the noise content in the response variable C_L increases with

α in a way that is similar to what is seen in flight, since there tends to be higher noise or unmodeled dynamics at higher α.

Using this simulation data, a single linear model was estimated for $C_L = f(\alpha)$, and the resulting residuals in Fig. 3.3(b) clearly show that deterministic modeling information is not captured, which is the nonlinear behavior of C_L at high α. When the SPLITR approach was used to estimate a model with $\phi = [\alpha]$, α was divided into three cells, as shown in Fig. 3.2, and the corresponding residuals in Fig. 3.3(a) are improved. The associated piecewise linear models shown in Fig. 5.1(a) represent the modeling data in each cell well, and the weighted global nonlinear model is shown in Fig. 5.1(b), with $R^2 = 0.98$.

This simplified 1D example demonstrates how nonlinear flight data can be successfully modeled with several piecewise linear functions using the SPLITR approach, that the resulting model captures the local and global aerodynamics well, and that the partitions placed along the familiar lift curve are consistent with a physical interpretation of the various α regimes. However, aerodynamic forces and moments are generally functions of multiple explanatory variables, and they are more difficult to estimate from flight data that contain unknown errors, including instrument noise, additional vehicle dynamics, and other disturbances. This F-16 example provides a useful visualization of the model fits, whereas the next section will apply the SPLITR approach to experimental flight data obtained from the E1 aircraft.



(a) Local model fits

(b) Global smoothed model fit

Figure 5.1: SPLITR model fits of $C_L = f(\alpha)$ for F-16 data.

5.3 Practical aspects of modeling flight data

The SPLITR algorithm is intended to be an automated modeling method that can be applied to various aircraft, so this section will summarize the algorithm parameters and other specifications that still must be user-prescribed.

It is important to note that aside from λ_ν, all of the baseline user input specifications and guidance described in Section 3.5, and used for the baseline simulation cases in Chapter 4, were used for modeling the E1 and T-2 test data. Although SPLITR contains several user inputs to allow increased flexibility where more insight is available into the system, in general the modeling process was found to be insensitive to small variations across the input specifications and throughout different sets of flight data. The most impactful parameter was found to be the residual threshold factor λ_ν, which is related to the cutoff between acceptable and unac-

ceptable residuals, and influences the split determinations. A value too large will prevent any splits from occurring, and a value too small will result in an overly complex model. In the simulated test data examples in Chapter 4, $\lambda_\nu = 2$ was sufficient for many data sets with small amplitude Gaussian noise. For flight test data that contain unknown noise and other unmodeled information content, the threshold needs to be raised to ensure that excessive splitting does not occur. For these experimental data examples, λ_ν was set to 4, which effectively places the residual threshold at 4σ. Since the E1 and T-2 are conventional aircraft, the expectations are that the aerodynamics are linear throughout much of the flight envelope. Therefore, the models across neighboring cells are not expected to be significantly different, and so the cell initialization proportion was set to 1, similar as for the quadratic test case in Chapter 4, which will still allow the flexibility to account for nonlinear regions such as stall. This is in contrast to the piecewise linear case where the child cell models were expected to be drastically different from the parent cell models. In these results, the maximum number of cells was not limited in order to demonstrate the effective model complexity management.

For scaled aircraft, the dynamic modes are scaled according to the geometry properties as

$$f_{model} = \frac{f_{aircraft}}{\sqrt{s}} \tag{5.9}$$

where s is the scale factor of the model [21]. General aviation aircraft typically exhibit rigid-body dynamics of less than 1 Hz, so for the scaled aircraft used in this

work, the dynamics were estimated to be less than 2 Hz. Therefore, a fourth-order high-pass Butterworth filter was used with a cutoff frequency of 3 Hz to estimate the noise, similar to what was used in Chapter 4 [14].

Other information that must be defined in advance for the modeling approach discussed in Section 5.1 includes the aircraft mass, geometry, and inertia properties to nondimensionalize the forces and moments. For each model, the regressors, available PVs, and expected ranges of PVs must also be specified.

In this work, complex nonlinear behavior in the response variables is addressed by partitioning the flight data to capture the simpler aerodynamics locally. However, this approach can be viewed more generally such that any deterministic behavior in the residuals as a function of the PVs outside of the residual threshold will be treated the same by triggering a split. The regressors are specified in advance, separate from the cell structure determination. So if an inadequate regressor pool is chosen, either one that contains irrelevant or strongly correlated regressors, or one that is missing important terms, then the algorithm may try to compensate for the insufficient modeling information by splitting unnecessarily. These successive splits may therefore not actually improve the model fit.

As discussed in Section 3.2.2, only linear regressors were chosen based on the flight data, which results in an LMN consisting of local linear models. Higher-order and multivariate terms can be included as well in each cell, which would yield a network of local nonlinear models. This would likely reduce the model complexity by resulting in fewer cells, but could also pose identifiability problems for each cell, unless local model structure determination techniques were applied.

In these results, the modeling process was not performed onboard the aircraft, but rather using data obtained from the test flights that were post processed in a real-time simulation on a laptop computer. Since the flight computers recorded data at 50 Hz, the data processing sequence operated at 50 Hz as well. The cell splitting procedure ran at 5 Hz, which was beyond the expected range of rigid body dynamics for the test aircraft. It could also have been set to run much slower to reduce the computational load, since the splits do not occur frequently. The flight data used in these results were obtained from several flights during which the pilot performed low-α flight operations as well as high-α stall maneuvers. The pilot inputs were overlaid with excitation inputs added to the control surface commands. The goal was to obtain global data with high information content across a wide range of flight variables.

5.4 E1 flight test data

First, the E1 test aircraft will be described, and then the SPLITR results will be shown through in-depth case studies that describe the model development and results for C_m and C_X.

5.4.1 E1 test aircraft

The test vehicle used in this work was a 40% scale Extra 330SC radio-controlled battery-powered aircraft, designated as E1 and shown in Fig. 5.2. The cruise speed for E1 was approximately 50 kts, and the nominal altitude throughout the flights

used in these results was 600 ft above ground level. The control surfaces are ailerons, flaps, elevator, and rudder. The geometry and mass properties for the flight vehicle are summarized in Table 5.2, and were determined through ground-based tests. The aircraft was instrumented with an inertial navigation system (INS) that provided 3-axis translational accelerations, angular rates, Euler angles, GPS position, and velocity. An air flow angle vane mounted on a boom on the right wingtip measured the angle of attack and sideslip angle, a pitot-static tube provided static and dynamic pressure measurements, encoders measured each of the control surface deflections, and a Hall-effect sensor measured the propeller speed. These measurements were used to calculate the aerodynamic forces and moments given in Eqs. (5.1 –5.6), as well as for the explanatory variables used for modeling. In particular, the calculation for advance ratio is given as

$$J = \frac{V}{\Omega d} \tag{5.10}$$

where V is the airspeed, Ω is the propeller angular rate, and d is the propeller diameter. Additional information about the E1 test vehicle can be found in Refs. [6, 13].

Figure 5.2: E1 test aircraft (credit: NASA Langley Research Center).

Symbol	Value	Unit
$\bar{c}$	1.97	ft
b	10.17	ft
S	19.26	ft^2
m	1.910	slug
I_x	2.964	slug-ft^2
I_y	8.776	slug-ft^2
I_z	11.716	slug-ft^2
I_{xz}	0.750	slug-ft^2

Table 5.2: E1 geometry and mass properties.

5.4.2 Case study of C_m model

For the case of modeling C_m, the model structure was defined as

$$C_m = C_{m_0} + C_{m_\alpha}\alpha + C_{m_q}\frac{q\bar{c}}{2V_0} + C_{m_{\delta_e}}\delta_e + C_{m_J}J \tag{5.11}$$

The PV was specified as α with a minimum cell width of 1 deg, since it is expected that the nonlinearities in C_m are based on α.

Figure 5.3 shows the time history of the response variable C_m and the corresponding PV α over a period of flight test data used for modeling. The modeling

data of C_m were estimated to have an SNR of 5.2. Note that the nonlinear dependency of C_m on α is evident as the splits occur during the high-α maneuvers as the flight envelope is expanded and new local models are required. There were 4 large-amplitude stall maneuvers performed throughout the duration of the modeling data, with 2 splits triggered. The remaining flight data were used to update the resulting cells.

In flight test data, particularly when the PTIs are active across all axes simultaneously, there are variations in all of the flight variables at once. Visually discerning where the nonlinear breakpoints may be placed as a function of the known PVs can therefore be unclear, as shown by the plot of C_m vs. α in Fig. 5.4. This is in contrast to the simplified example using the F-16 data, and implies that the algorithm logic must be relied upon to determine the appropriate breakpoints.

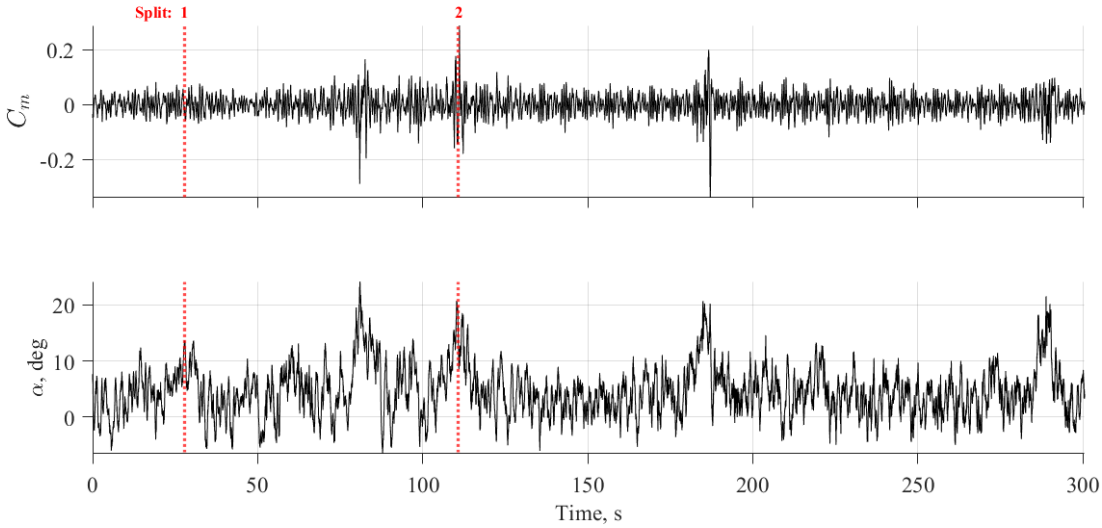


Figure 5.3: Time histories of response variable and PV modeling data for E1 C_m model.

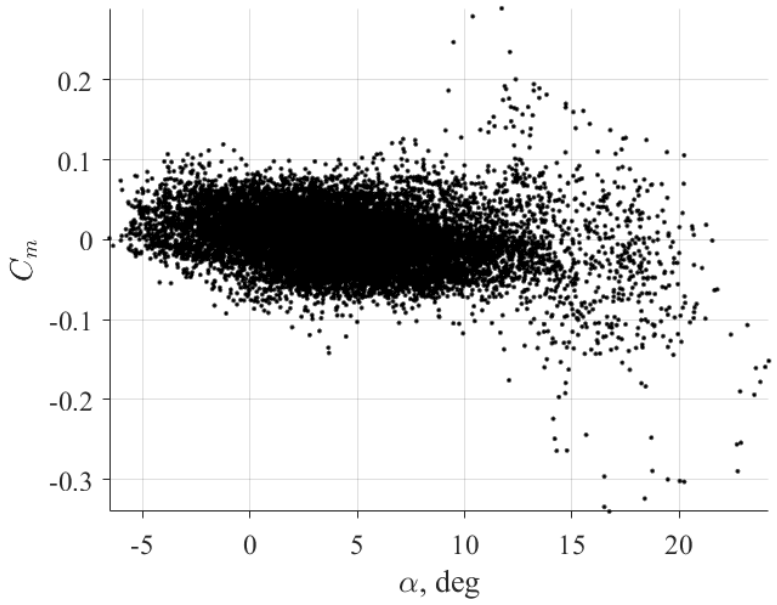


Figure 5.4: C_m vs. α modeling data for E1 C_m model.

The cell structure evolution for C_m parametrized across the range of α is shown in the hierarchical depiction in Fig. 5.5. The first split partitions the high-α from the low-α aerodynamics, while the second split partitions what appears to be the stall region from the post-stall region. The final cell structure shown with three cells is consistent with a physical interpretation of an individual linear model across a confined range of low-, mid-, and high-α data.

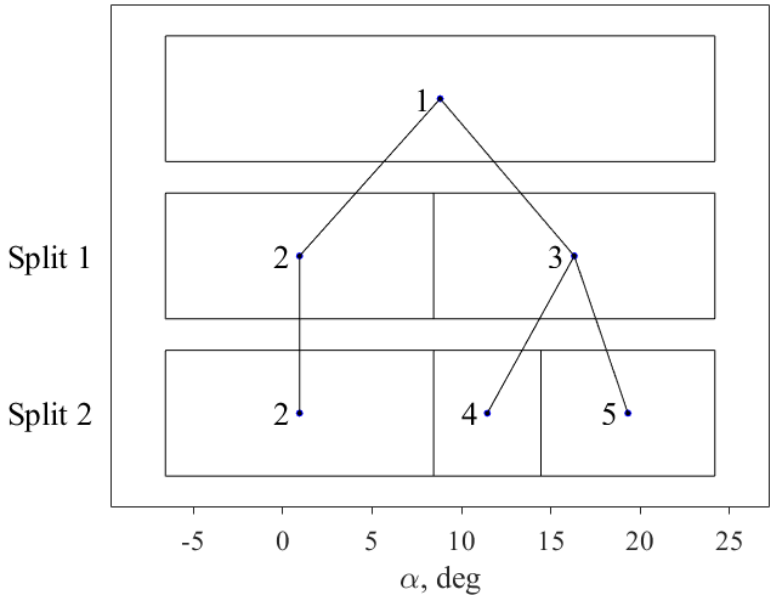


Figure 5.5: Cell structure evolution for E1 C_m model.

A closer look through a depiction of the cell structure decision-making process in Fig. 5.6 clarifies how the cells evolve over time based on the available data. The minimum bin widths of 1 deg used to discretize the possible bin boundaries are shown in light gray.

Beginning with one linear model across the range of α, notice that at low α, although there are numerous residuals that are characterized as unacceptable, their relative magnitude is still small compared to the acceptable residuals, and so the bins pass the status check. At mid α, however, the unacceptable residuals begin to outweigh the acceptable ones, and bins 8-9 are considered failed, so the first split is performed at the left boundary of bin 8. Note that there are not yet enough data points in bin 10 for it to fail. Correspondingly, the second split partitions the mid-α data from the highest-α region before bin 4 of cell 3. Although bins 7–10 in cell 3

also contain numerous unacceptable residuals, the total number of residuals in each bin is below the specified minimum amount, and so these bins are not considered failed yet. After the splits are performed and additional data are gathered across all cells, the model fit in cell 2 appears reasonable, and while there are still large magnitude residuals in cells 4 and 5, no further splits were made. Note also, for example, that the magnitude of the acceptable residuals in cell 5 is much larger than the magnitude for those allowed in cell 2 because the residual threshold is dependent on the noise levels. It is possible that further high-α maneuvers would provide additional information to perform another split in that region, but the model fit at high-α is not necessarily anticipated to be of high quality due to the noise and other unmodeled content.

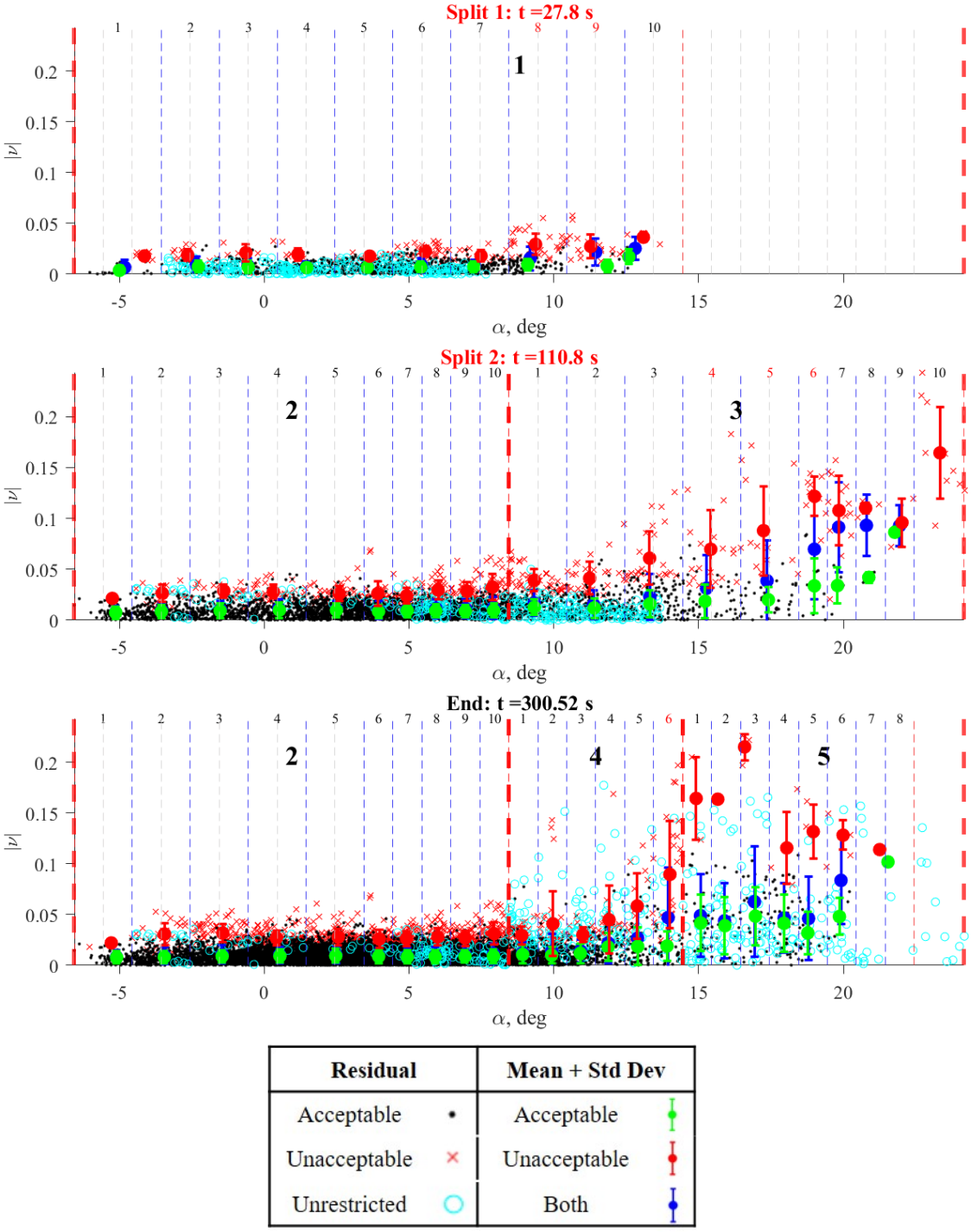
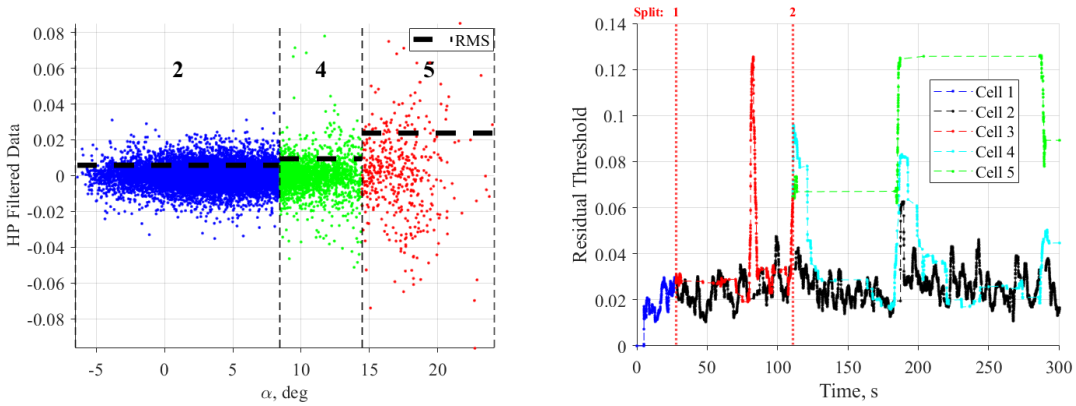


Figure 5.6: Binned local residual characterization for E1 C_m model.

To illustrate the dependency of the split decision making on the residual threshold more clearly, Fig. 5.7(b) shows the residual threshold over time for all

cells. Notice that the noise estimates increase across α as expected, with the high-pass filtered modeling data reflecting this in Fig. 5.7(a). Since the filter window was only specified to contain 2 seconds of data, the residual threshold fluctuates substantially, which is particularly noticeable in cell 2 which contains a large amount of low-α data. However, this small window provides the localized resolution to be able to capture the rapidly changing noise content during the stall maneuver that occurred around 75 seconds, and after which no split was performed. The small window size enabled the residual threshold to quickly adapt to the current conditions. The residual threshold, based on the current assessment of noise content in the data, is the gateway toward both allowing further splits, but also preventing them unnecessarily to avoid splitting due to poor model fit brought about by noise.

One of the pervasive challenges throughout this work has been shown to be balancing the tradeoff between remembering past information and adapting quickly to new data. In particular, the noise content in experimental data can be expected to vary as a function of one or more variables, so it is advantageous for the SPLITR method to adjust the model fit expectations based on rapid adaptability to the current conditions. This will ensure that excessive splitting or overfitting does not occur where the residuals exhibit poor model fit due to increased noise content, and not due to a poor characterization of deterministic modeling information.



(a) High-pass filtered modeling data

(b) Residual threshold

Figure 5.7: Residual threshold characteristics for E1 C_m model.

Figure 5.8 shows the final parameter estimates for each cell, along with their associated 95% (2σ) uncertainty bounds, plotted as a function of α at the center of the associated cells. The model fit error variance in Eq. (3.6) was computed in batch using only the measurements that were actually contained within the range of each cell, as it was done throughout this work. This plot offers further valuable physical insight into the aerodynamics. The static longitudinal stability C_{m_α} is appropriately negative for a statically stable aircraft, and the magnitude gets larger across the cells, indicating a stronger restoring moment as α is increased. The pitch damping derivative C_{m_q} describes the pitching moment that is caused by pitch rate. It is related to the added lift generated on the horizontal tail due to aircraft rotation about the center of mass, and is negative to indicate dynamic pitch stability. It is largest in magnitude at low α and decreases across the range of α as the horizontal tail effectiveness is reduced. Accordingly, the elevator effectiveness $C_{m_{\delta_e}}$ is maximized at low α and reduced at higher α. Advance ratio did not have an apparent impact on pitching moment at low α, but the dependency increased at higher α,

corresponding to when power was added during the stall maneuvers. Note that the uncertainty tends to be larger for the estimates in the higher-α cells where less data are obtained, where noise levels are larger, and where the aerodynamics are more difficult to model, with unsteady and coupled lateral-directional behavior possible. The physical insight offered by the stability and control derivatives estimated across the range of α was consistent with engineering expectations in this case. For future test cases of vehicles with unknown aerodynamics, these parameter estimates can offer valuable insight into localized aerodynamics across different parts of the flight envelope. The local aerodynamic models can also be useful for local linear control law design in particular flight regimes.

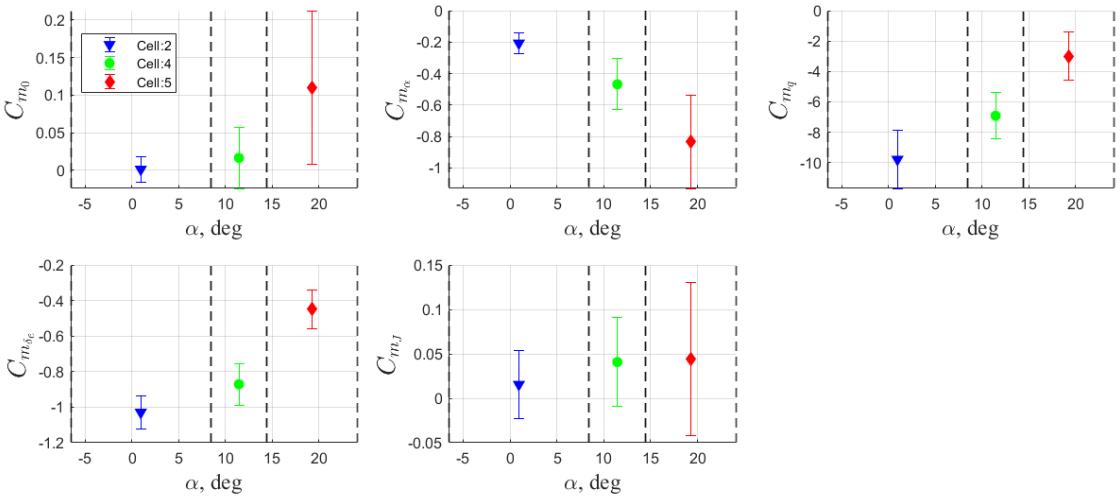


Figure 5.8: Local model parameter estimates across all cells for E1 C_m model.

Figure 5.9 shows an example of the time history of the local parameter estimate C_{m_α} through all of the splits. Note the definitive discontinuities in the estimates following each split, as this is when the unacceptable residuals are incorporated into the child cell models instantaneously. The parameter estimate for cell 2 appears to

converge due to much data, whereas those for the other final cells are only updated intermittently during the remaining high-α maneuvers following the last split. Note also the perturbations in the estimates for cells 4 and 5 at around 184 seconds and 286 seconds, as that is when the latter two stalls occur in the modeling data.

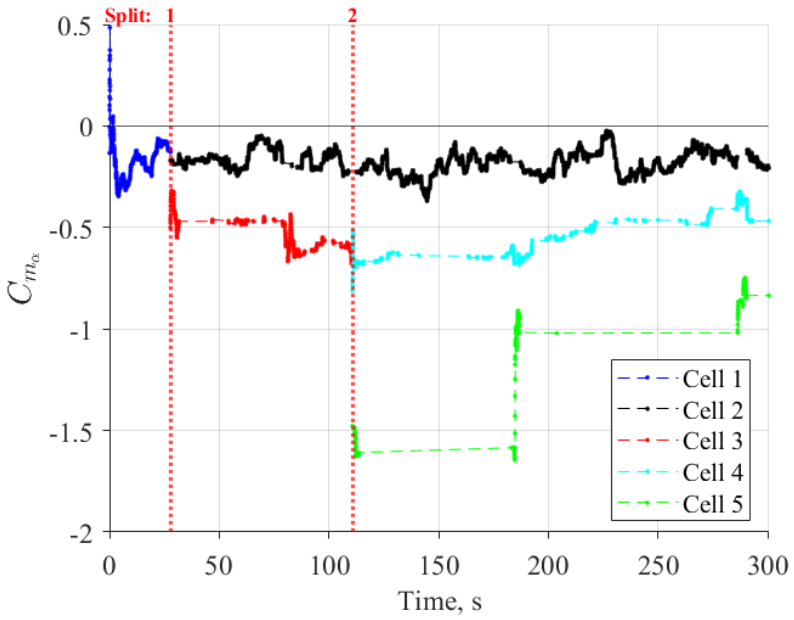
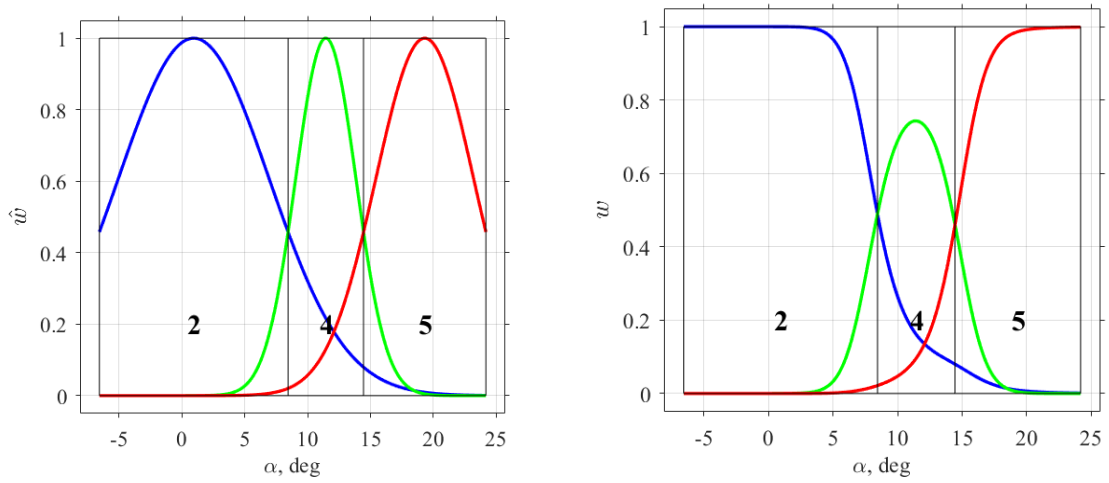


Figure 5.9: Time history of C_{m_α} estimates across all cells for E1 C_m model.

As discussed in Section 3.2.4, each cell's model is locally approximated irrespective of the adjacent cells or of the global weighting, which retains the physicality and interpretability of the resulting parameter estimates. The global nonlinear model is then calculated using the validity functions shown in Fig. 5.10 that apply weights to the local models in each cell and describe the influence of each model across the range of the PV. For example, Fig. 5.11 shows the weighted global parameter estimate for C_{m_α} across all cells, along with the local estimates relevant most strongly at the center of the respective cells. The waves in the global estimate across the cell boundaries are a result of the normalization of Gaussian validity

functions that is required to enforce a partition of unity, and which can potentially cause significant and unintentional deformation of the weighting functions for many cells of varying widths [73]. This effect could be mitigated with a wider Gaussian standard deviation that is specified with a larger λ_s.



(a) Validity functions (b) Normalized validity functions

Figure 5.10: Validity functions for E1 C_m model.

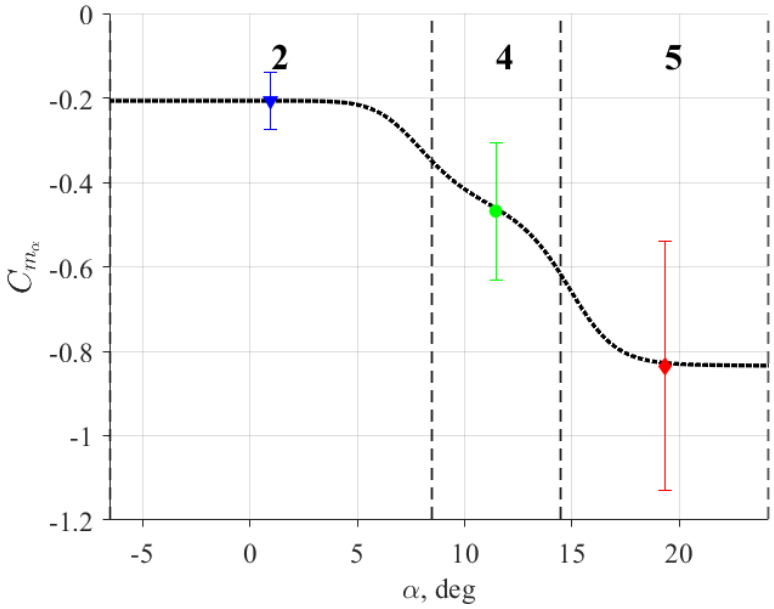


Figure 5.11: Local and weighted global parameter estimates of C_{m_α} for E1 C_m model.

Conventional aircraft exhibit linear aerodynamics about an equilibrium point, and a linear model could be sufficient at each point to describe most of the operating envelope. For the purposes of global modeling, a nonlinear model is shown to offer improvements over a single linear model defined across the entire data set. The error from the SPLITR model with 3 cells was compared to that of a batch linear model developed using all of the same data. The linear comparison model can be thought of as having the initial cell structure shown in Fig. 5.5, with only 1 cell across the entire range of α. The SPLITR model recognized the inherent nonlinearities in C_m, and this model with 3 local cells provides a better fit with $R^2 = 0.69$, compared to the linear model, which has $R^2 = 0.58$ for the modeling data.

Figure 5.12 further shows the RMS of the global residuals across 15 bins for each model. The SPLITR model shows a reduction in error at low α and high α compared to the linear model, but little to no improvement around where stall is expected to occur. Stall behavior is a complex and nonlinear phenomenon with coupled and unsteady effects that are not accounted for in these simplified local, linear models. Note in Fig. 5.6 that there is scarce information obtained in the region of the right-most bin shown here in cell 5, and the error is also largest there. Sources of modeling error can also include measurement noise, model structure errors resulting from local nonlinearity or an inadequate regressor pool, in-flight disturbances, neglected dynamics such as unsteady, structural, or aeroelastic effects, and violations of the quasi-steady assumption inherent in the model structure. It is important for model predictive capabilities that the noise content is not overfit by the model, and SPLITR has been shown to successfully partition the model and

provide a good model fit. Validation data, presented next, will confirm that the model provides adequate prediction as well.

The basis of the cell structure determination process is to reduce the RMS magnitude of the residuals as a function of the PVs that are associated with the nonlinearities in the data. For this conventional E1 vehicle, the aerodynamics are fairly linear throughout much of the flight envelope, which is why the linear model also performed well, but the logic behind this method is also applicable for less conventional vehicles. Still, improvements were seen in this case, and the SPLITR model succeeded in reducing the RMS magnitude of the global residuals as a function of α compared to the linear model, as shown in Fig. 5.13.

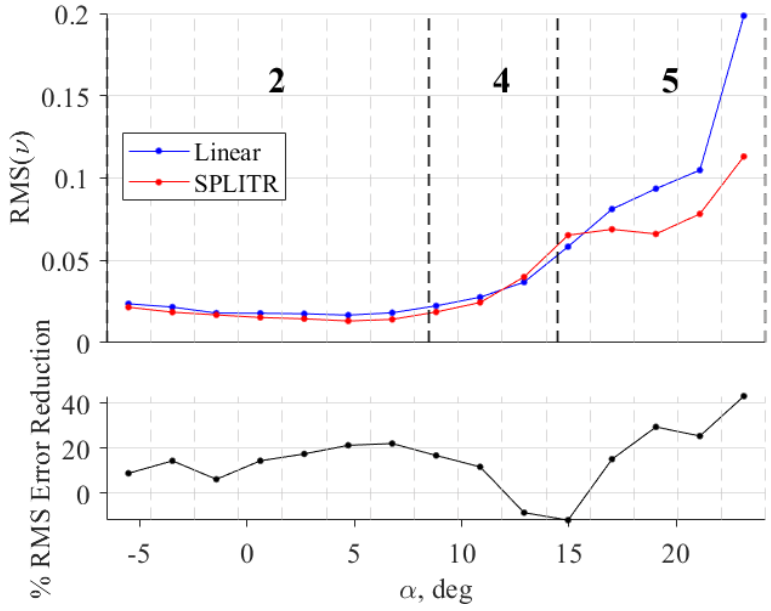
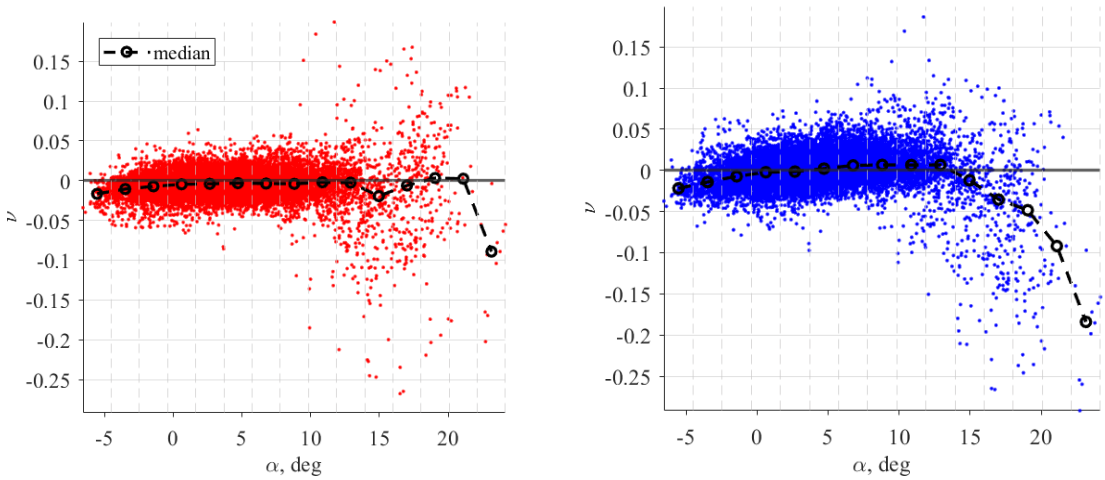


Figure 5.12: RMS of binned global residuals for modeling data for E1 C_m model.



(a) SPLITR nonlinear model

(b) Linear model

Figure 5.13: Global residuals vs. PV for modeling data for E1 C_m model.

A separate set of flight data that was not part of the modeling process was used for model validation to assess the model's predictive capabilities. The prediction error for the SPLITR model was also compared to that of a batch linear model. The time histories of C_m and α for this data set are shown in Fig. 5.14. The validation data included additional low-α maneuvers and 3 stalls to ensure a wide range of α was represented so that the full extent of the model could be tested. The linear model achieved an R^2 of 0.63 for the validation data, while the SPLITR model produced an R^2 of 0.74. Figure 5.15 shows the RMS of the binned residuals across the cells, and also indicates improvements compared to the linear model across most of the α range. Figure 5.16 shows how the residuals were effectively reduced in magnitude compared to the linear model, particularly at high α.

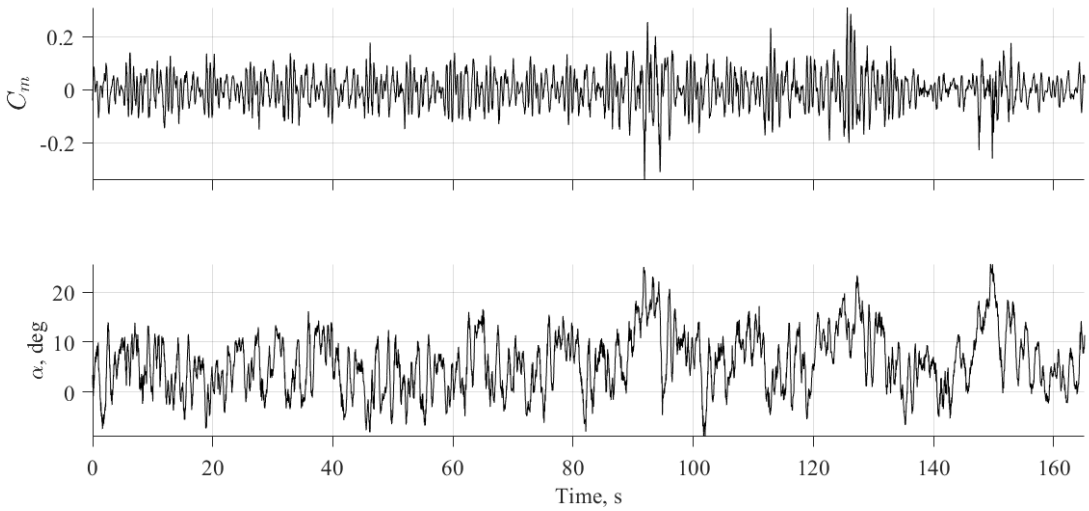


Figure 5.14: Time histories of response variable and PV validation data for E1 C_m model.

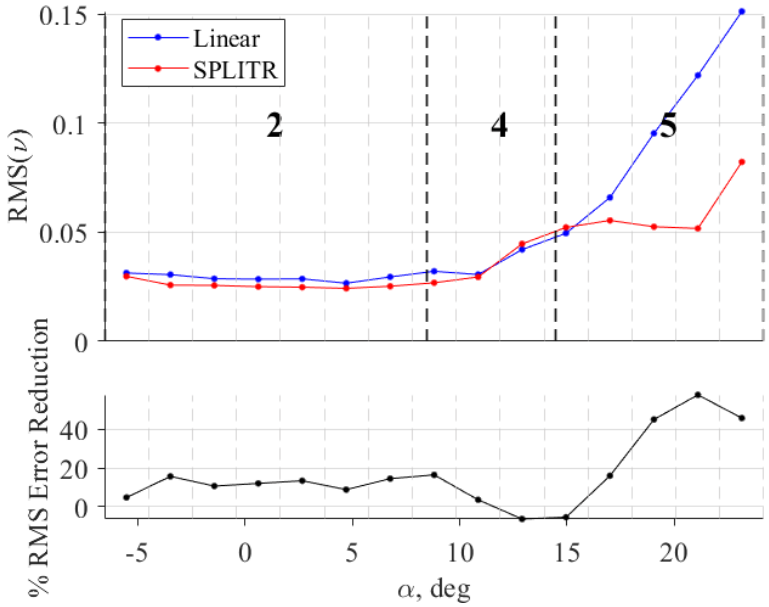
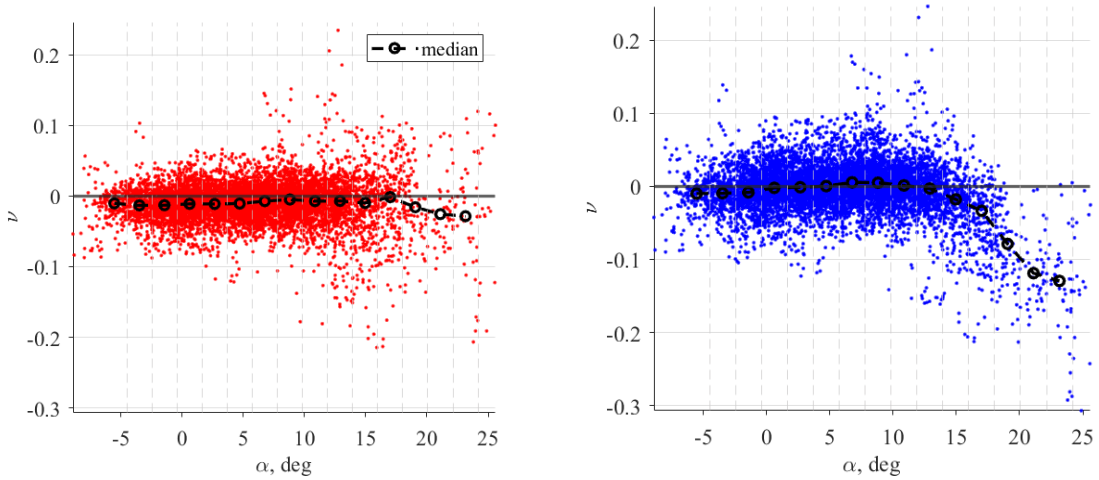


Figure 5.15: RMS of binned global residuals for validation data for E1 C_m model.



(a) SPLITR nonlinear model

(b) Linear model

Figure 5.16: Global residuals vs. PV for validation data for E1 C_m model.

The localized regions and associated models identified through SPLITR can provide further insight into where additional modeling data need to be gathered, and can potentially be used to inform an autonomous envelope expansion and model development algorithm that drives the experiment design. Each local model can be evaluated based on its fit error, the noise levels of the data it contains, its parameter estimate uncertainties, and the number of measurements, among other metrics. The regions with poor local model fits due to lack of data can then be more easily identified, and these regions can be returned to in order to obtain additional information. It is important to note that a crossplot of the data coverage of certain variables alone, or another form of data density depiction, does not provide this information since the relative number of data points in a certain area of the flight envelope is not a definitive indicator of the model quality in that region. A model fit can be sufficient with few data points if the dynamics are linear and well behaved, and

conversely, a poor fit can result even in a highly dense region if the model structure is not appropriate and/or the experiment is not performed well. Additionally, larger noise content in particular regions can lead to a poorer fit, regardless of the amount of data.

Figure 5.17 illustrates this with a histogram of the α modeling data that is colored with the binned RMS of the modeling error of the SPLITR model, which is shown quantitatively in Fig. 5.12. As expected here, the error is in fact inversely correlated with N at high α, but note that the model performs well in the lowest-α region where relatively few measurements also lie.

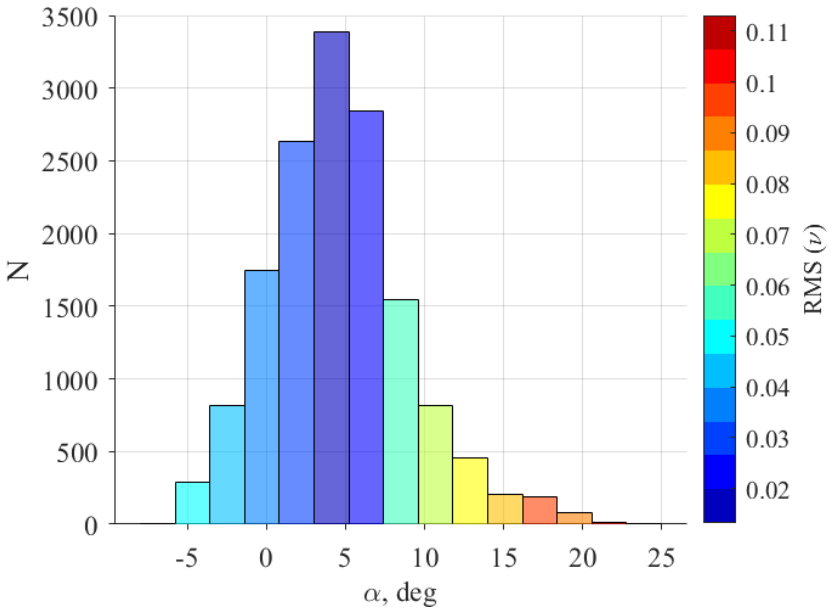


Figure 5.17: Histogram of PV modeling data with RMS of error for E1 C_m model.

5.4.3 Case study of C_X model

This section discusses the SPLITR C_X model developed using the same E1 flight data as for the C_m case study in the previous section. The model structure

for C_X is defined as

$$C_X = C_{X_0} + C_{X_\alpha}\alpha + C_{X_q}\frac{q\bar{c}}{2V_0} + C_{X_{\delta_e}}\delta_e + C_{X_J}J \tag{5.12}$$

The time history of the C_X response variable data is shown in Fig. 5.18 with an estimated SNR of 5.4. C_X is generally difficult to represent without a separate propulsion model, and since thrust was not measured directly for E1, the C_X calculation in Eq. (5.1) includes both the thrust and aerodynamic forces. As a result, there is a strong nonlinear dependency of C_X on J, as shown in Fig. 5.19, and so J was chosen as the PV for this case, with a minimum cell width of 0.05. In this case, this insight allowed for the explicit specification of J as the sole PV. In other cases where insight is not apparent, a wider pool can be specified, and the primary indicators of nonlinearity can be automatically chosen. However, for computational efficiency and to avoid the possibility of unexpected splits, the PVs should be specified as carefully as possible.

All of the same SPLITR user specifications as for the C_m example were used for this case as well, and so no further SPLITR user specification tuning was required. Note that power was increased during many of the stall maneuvers, which is apparent by comparing the plot of J in Fig. 5.18 to that of α in Fig. 5.3.

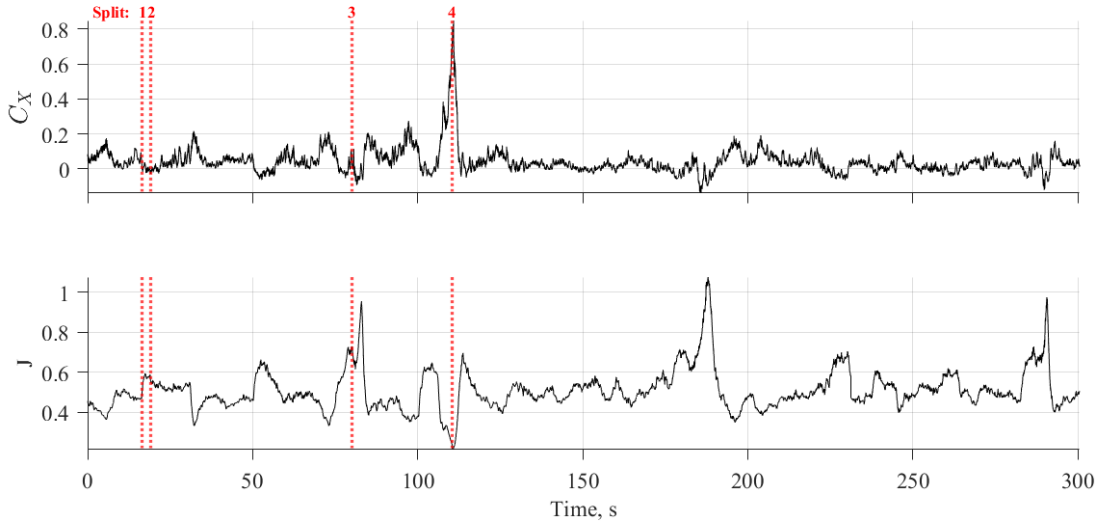


Figure 5.18: Time histories of response variable and PV modeling data for E1 C_X model.

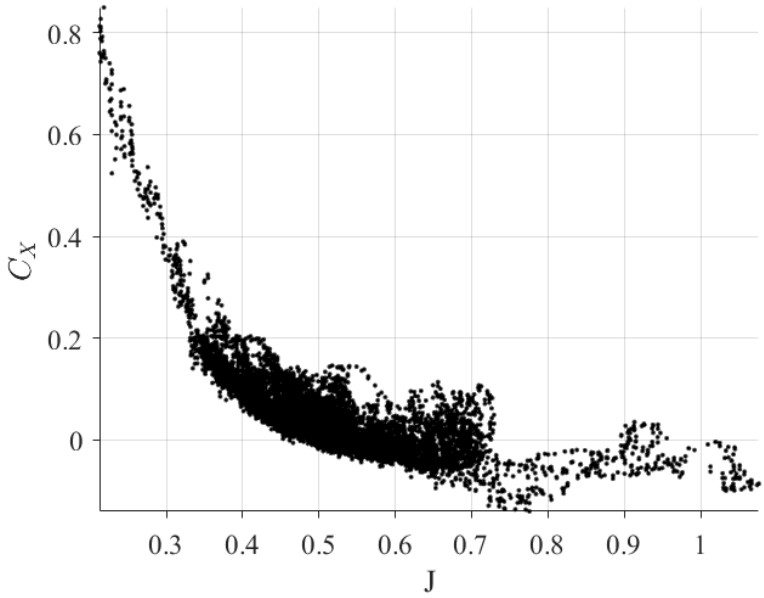


Figure 5.19: C_X vs. J modeling data for E1 C_X model.

The SPLITR model for C_X resulted in 5 cells across the range of J, as shown in Fig. 5.20. Note that the clustering of partitions lies in the regions of large slope change across the modeling data in Fig. 5.19. It is possible that fewer cells may have been required if the cell initialization proportion was set lower since there is

stronger nonlinearity apparent in this case compared to C_m.

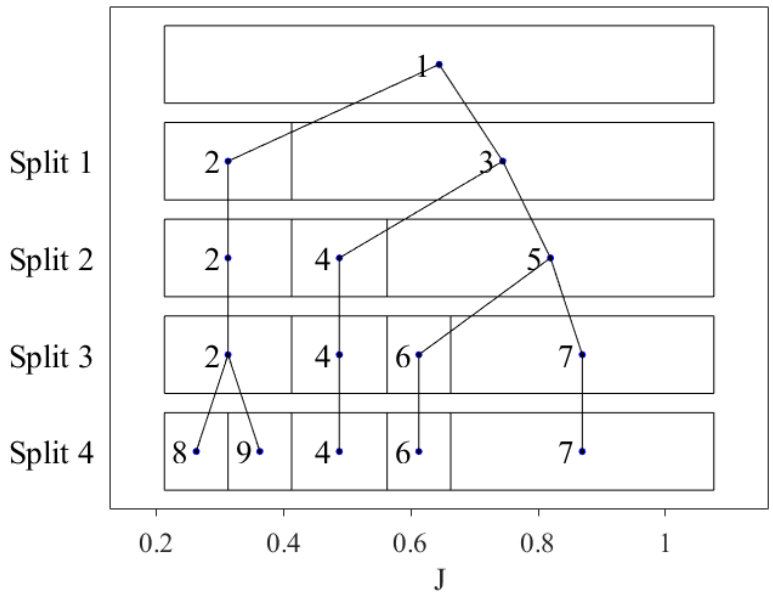


Figure 5.20: Cell structure evolution for E1 C_X model.

The final cell parameters for the SPLITR model are presented in Fig. 5.21. Consistent with the dependency of C_X on J shown in Fig. 5.19, C_{X_0} and C_{X_J} are maximized in magnitude in the lowest-J cell, and shift toward 0 at higher J. The C_{X_q} parameter increases across higher J, while C_{X_α} and $C_{X_{\delta_e}}$ do not appear to show a clear trend associated with J. Recall that much of the higher-J data occur during the stall maneuvers, so the parameter estimates in the higher-J cells may also be influenced by the high-α stall behavior there.

Figure 5.22 shows how the validity functions weight the global C_{X_J} parameter across all of the cells, with the constant and near zero slope value in cell 7 reflecting the behavior in the modeling data in Fig. 5.19. The corresponding validity functions across all cells are presented in Fig. 5.23.

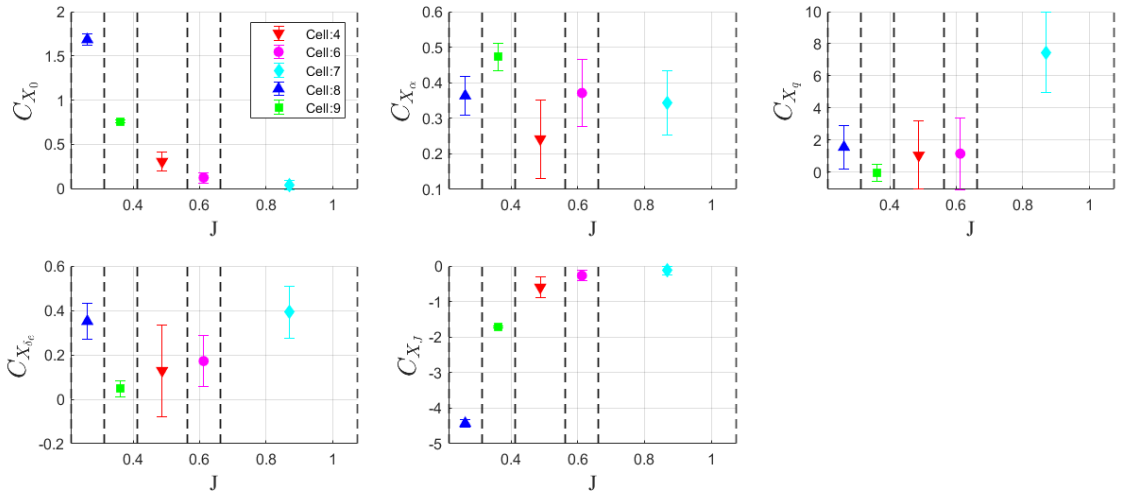


Figure 5.21: Local model parameter estimates across all cells for E1 C_X model.

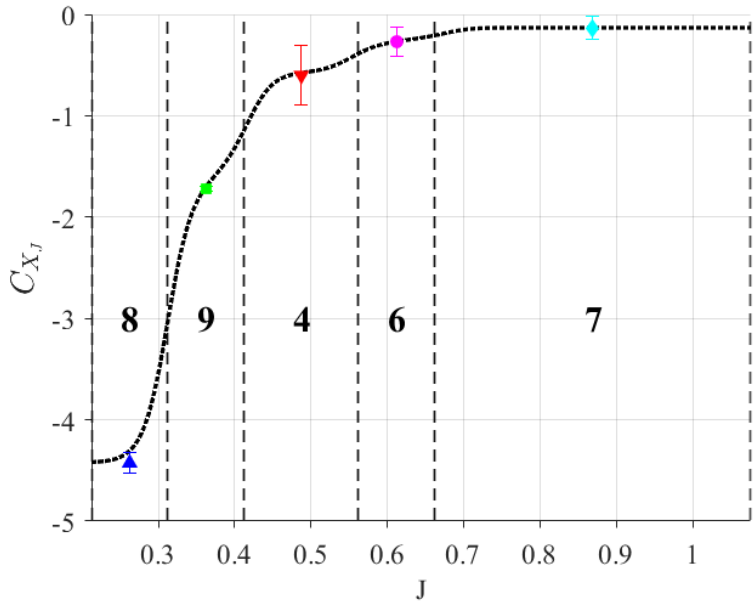
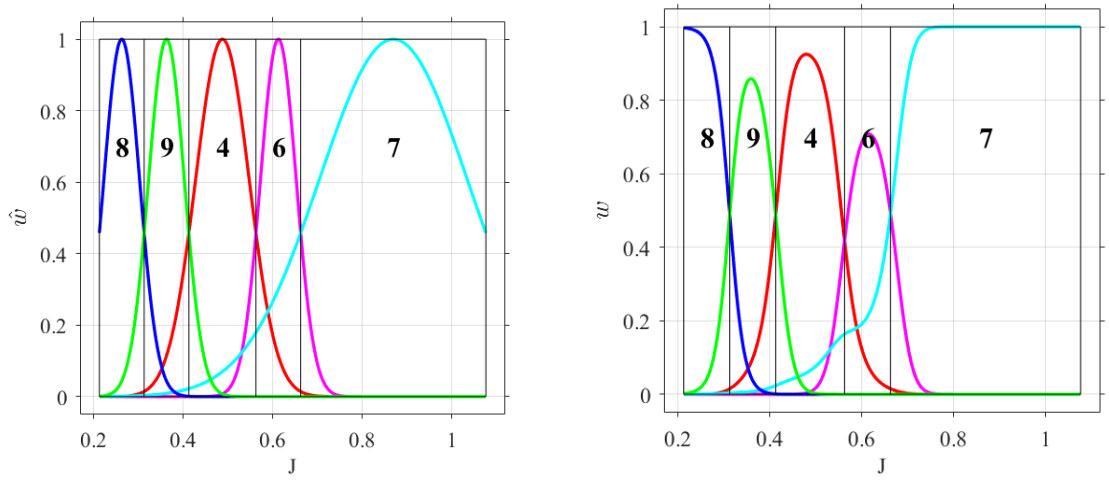


Figure 5.22: Local and weighted global parameter estimates of C_{X_J} for E1 C_X model.



(a) Validity functions (b) Normalized validity functions

Figure 5.23: Validity functions for E1 C_X model.

Similar to the case for C_m, the SPLITR model for C_X was compared to a single linear model developed using the entire data set. In this case, the strongly nonlinear dependency of C_X on J is much better captured using the SPLITR model which automatically accounts for the nonlinear effects. The RMS of the error using the modeling data is displayed in Fig. 5.24, which shows significant improvement for the SPLITR model with 5 cells compared to the linear model. Additionally, the error is shown to be relatively constant across the range of J for the SPLITR model, indicating a consistent model fit in each cell. The SPLITR model has an R^2 of 0.92, while the linear model had an R^2 of 0.66. Figure 5.25 also shows how the residuals are effectively more constant in RMS magnitude for the SPLITR model results compared to the linear model results.

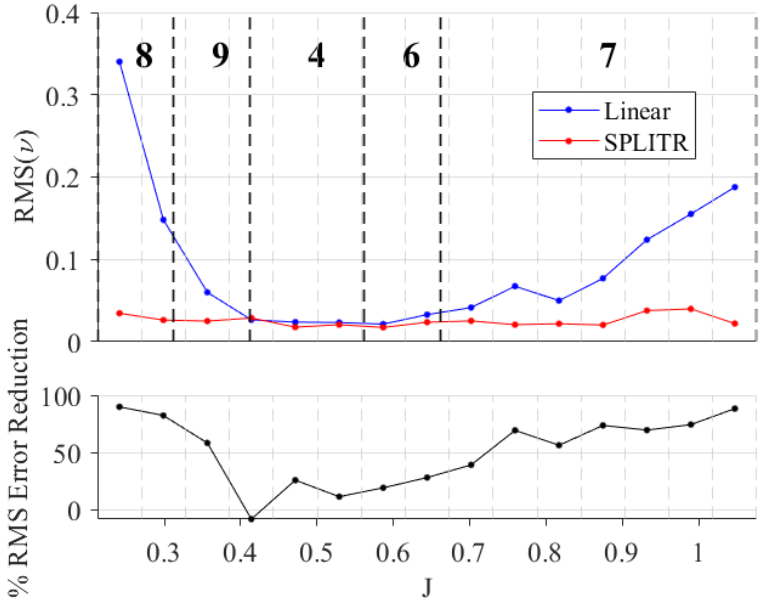
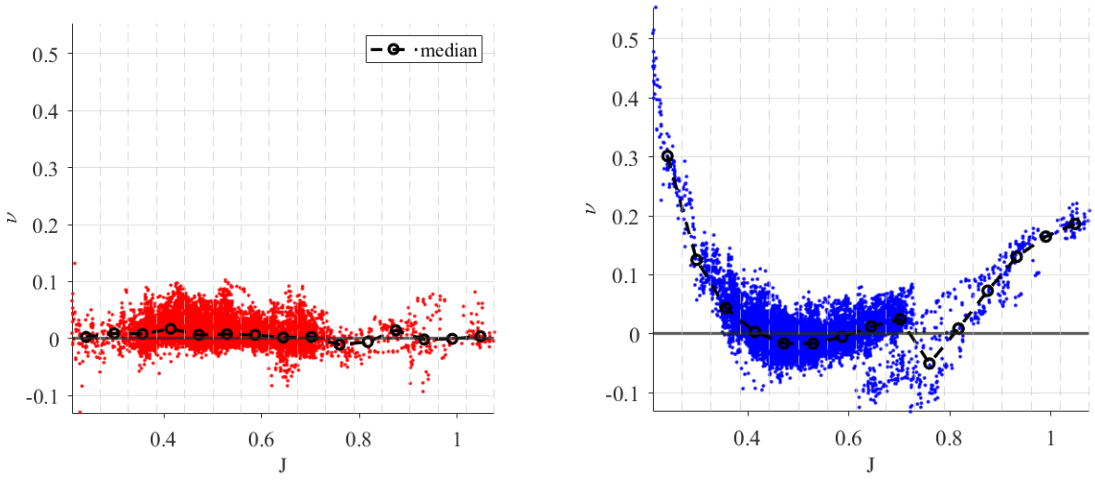


Figure 5.24: RMS of binned global residuals for modeling data for E1 C_X model.



(a) SPLITR nonlinear model

(b) Linear model

Figure 5.25: Global residuals vs. PV for modeling data for E1 C_X model.

Validation data were obtained from the same data set as in the C_m example, and the validation data time histories of C_X and J are given in Fig. 5.26. The SPLITR validation results were also compared to that of the linear model, with

an R^2 of 0.89 and 0.60, respectively. The RMS of the residuals in Fig. 5.27 shows improvement in error for the SPLITR model compared to the linear model, as well as displays similar RMS(ν) characteristics compared to the modeling data. Figure 5.28 also shows the more uniform magnitude of the residuals of the SPLITR model compared to the linear model.

This C_X example demonstrates both how significant nonlinearities can arise in practical flight test data, and how they can still be well-captured with local linear models using the SPLITR technique to automatically partition the aerodynamics in real time. Additionally, maintaining the flexibility of real-time updates allows the model to adapt quickly and automatically throughout the course of the envelope expansion.

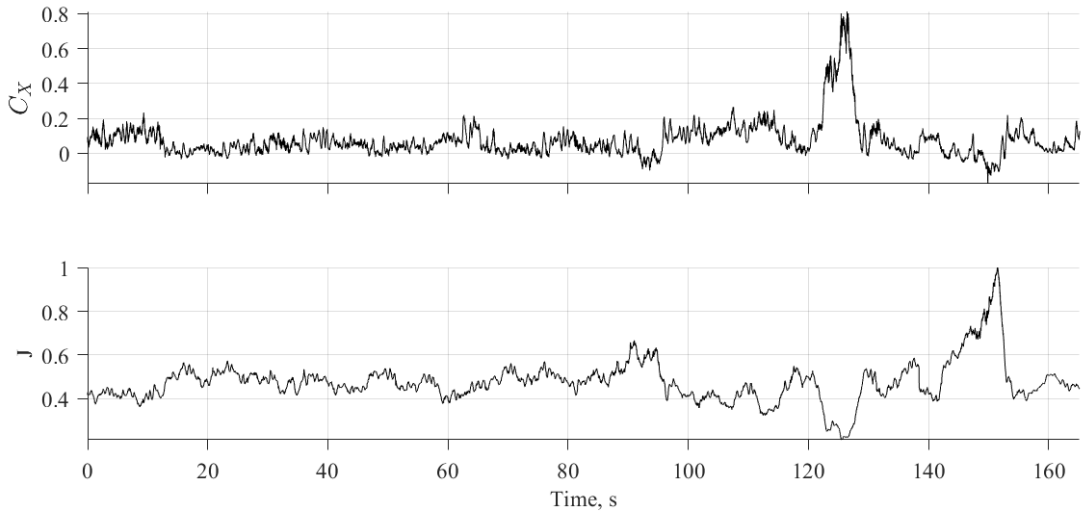


Figure 5.26: Time histories of response variable and PV validation data for E1 C_X model.

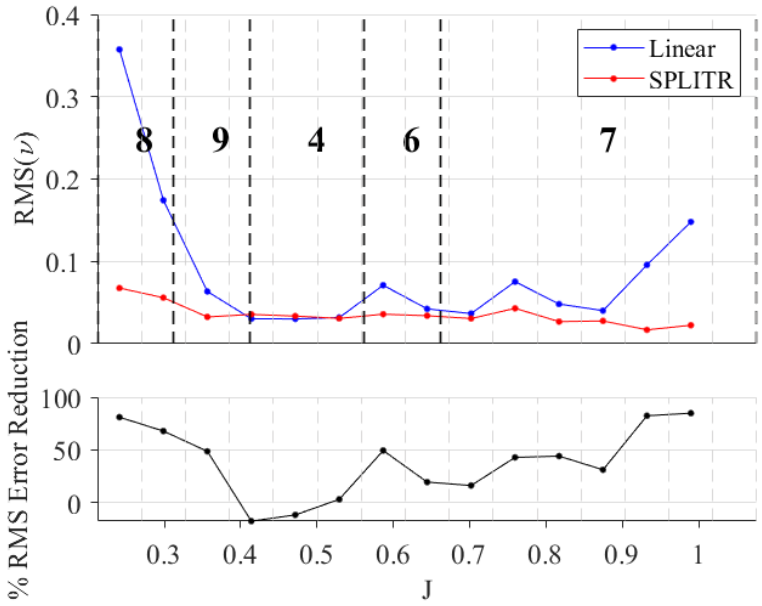
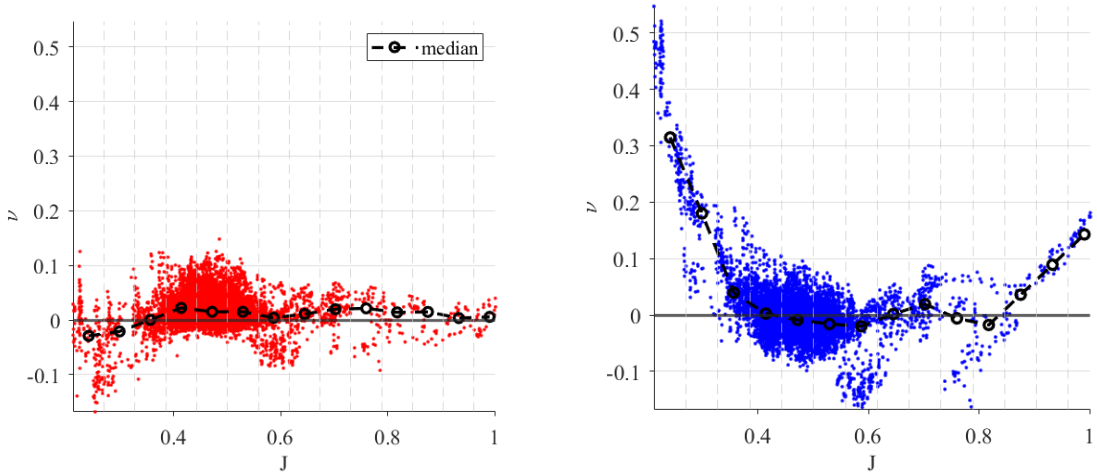


Figure 5.27: RMS of binned global residuals for validation data for E1 C_X model.



(a) SPLITR nonlinear model

(b) Linear model

Figure 5.28: Global residuals vs. PV for validation data for E1 C_X model.

5.5 T-2 Generic Transport Model flight test data

This section will show a final test case to develop a SPLITR model for C_L using flight data from the NASA T-2 Generic Transport Model (GTM) test aircraft.

This example uses all of the same SPLITR user specifications as the E1 cases, and is intended to show the consistency and versatility of SPLITR across multiple experimental platforms without the need to tune algorithm parameters.

5.5.1 T-2 test aircraft

The T-2 GTM is a 5.5% dynamically scaled twin-turbine powered test aircraft that represents a generic transport aircraft, and which was used as a research testbed at NASA Langley to explore modeling and control capabilities for aviation safety. It was flown by a remote pilot with real-time telemetry using the NASA Airborne Subscale Transport Aircraft Research (AirSTAR) capability. The T-2 is shown in Fig. 5.29, with the mass properties summarized in Table 5.3. Further information and details on the T-2 vehicle development and functionality can be found in Refs. [94, 95].

Figure 5.29: T-2 GTM test aircraft (credit: NASA Langley Research Center).

Symbol	Value	Unit
$\bar{c}$	0.915	ft
b	6.849	ft
S	5.902	ft^2
m	1.585	slug
I_x	1.179	slug-ft^2
I_y	4.520	slug-ft^2
I_z	5.527	slug-ft^2
I_{xz}	0.211	slug-ft^2

Table 5.3: T-2 geometry and mass properties.

5.5.2 Case study of C_L model

For the case of modeling C_L, the model structure was specified as in Eq.(5.13), with a PV of α and a 1-deg minimum cell width.

$$C_L = C_{L_0} + C_{L_\alpha}\alpha + C_{L_q}\frac{q\bar{c}}{2V_0} + C_{L_{\delta_e}}\delta_e \tag{5.13}$$

The T-2 flight data used for modeling in this test case were obtained at a nominal altitude of 1300 ft, at a nominal speed of 140 ft/s, and with the PTIs active on the control surfaces to provide dynamic excitation.

Figure 5.30 shows the time history data of C_L across the duration of the flight data used for modeling, with an SNR of 8. The data contain several periods of low-α maneuvers, as well as 3 very high amplitude stalls across 5.5 minutes of flight time. Two splits were performed during the first and last stalls. Ideally, additional flight data containing stall maneuvers would be obtained following the final split to update the cells, but this example shows reasonable results even without a large amount of further data required.

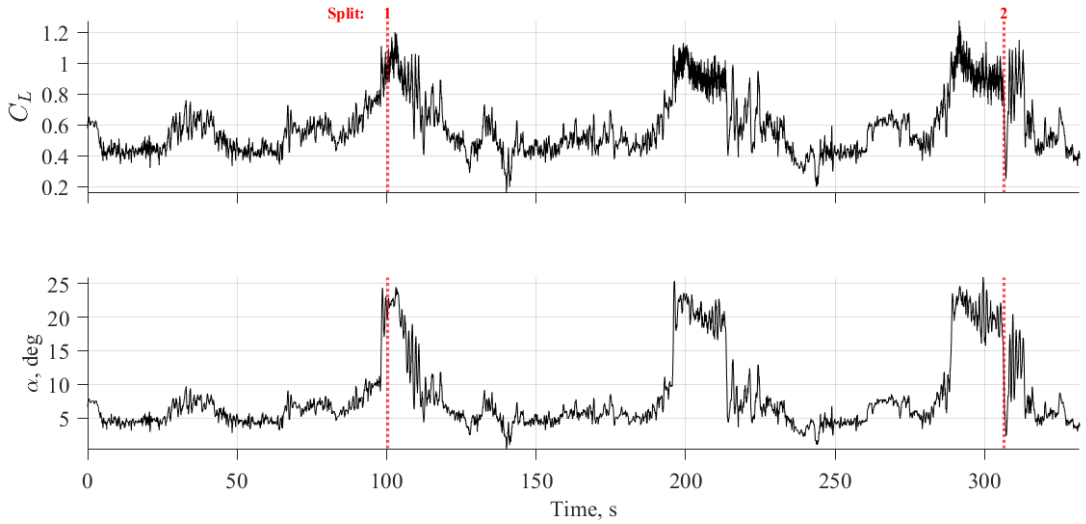


Figure 5.30: Time histories of response variable and PV modeling data for T-2 C_L model.

Figure 5.31 shows the response variable of C_L as a function of the PV α, which indicates linear aerodynamics in the low-α range, a stall region, and a post-stall region. The unsteady effects in the data cannot be captured with non-contiguous data that are sorted into each cell, but this behavior can be approximated with a varying C_{L_q} term in each cell. The cell structure evolution for the C_L SPLITR model is shown in Fig. 5.32. The first split partitions what appears to be the post-stall region from the lower-α data, and the second split offers a finer resolution around the onset of stall, ending with low-, mid-, and high-α cells that are consistent with the physical interpretation of the crossplot, and with the partitioning pattern for the E1 C_m case and F-16 C_L case.

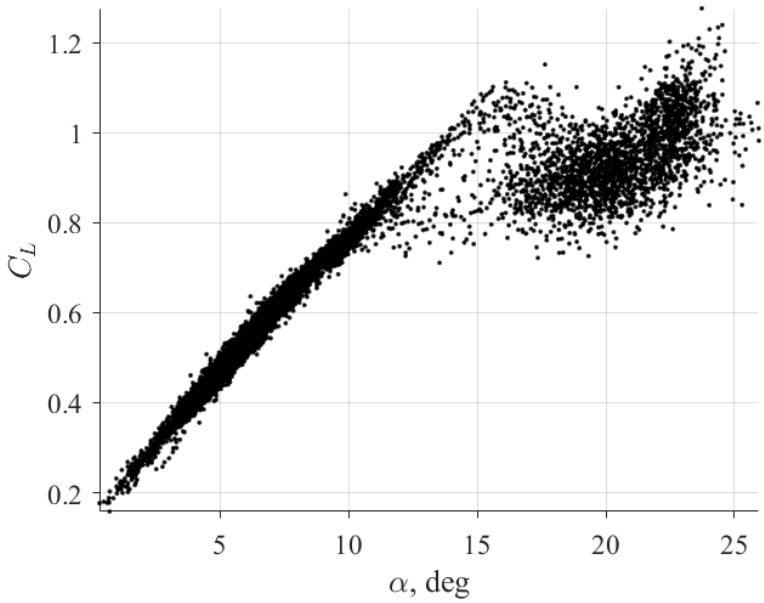


Figure 5.31: C_L vs. α modeling data for T-2 C_L model.

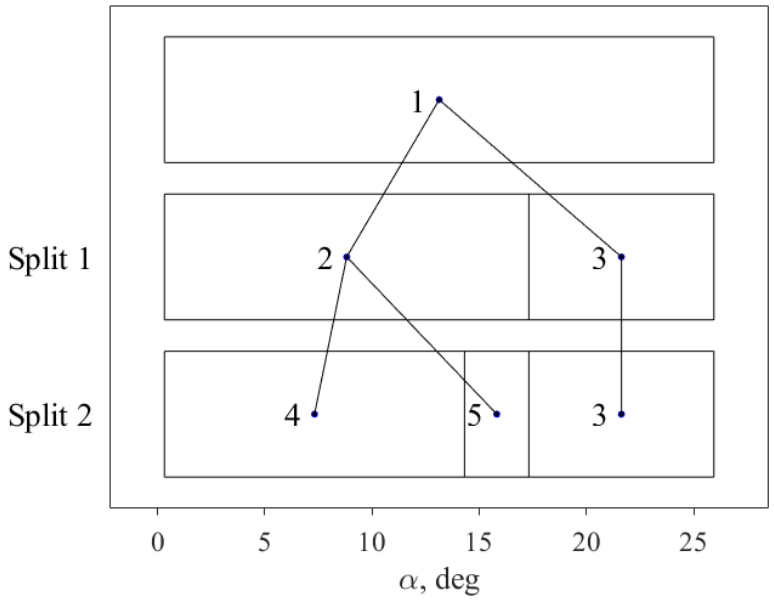


Figure 5.32: Cell structure evolution for T-2 C_L model.

The final parameter estimates across each of these cells are presented in Fig. 5.33, and they can provide physical insight into the aerodynamic behavior in each cell. As indicated in Fig. 5.31, there is a strong nonlinear dependency of C_L on α throughout

the data. The lift curve slope C_{L_α} is shown as positive, largest in magnitude in the low-α cell 4, and then much smaller in the stall and post-stall regions. The effect of positive pitch rate q is to create a larger effective angle of attack on the horizontal tail, which offers additional lift. This is reflected in the positive C_{L_q} parameters, which also show an increased effect at higher α, but then a drop off in the post-stall region. The elevator term $C_{L_{\delta_e}}$ is positive to show increased lift with a positive elevator deflection. The effectiveness also appears to decrease with α as the elevator becomes blanketed in the wing's flow wake. Because the last split was performed close to the end of the duration of the flight data used for modeling in this example, with no further stalls performed afterwards, only a limited amount of additional data were obtained after the last split to update the final child cell parameters. If those child cell models were initialized with no parent cell information inherited, then the C_{L_α} parameter in cell 3 was actually larger than that in cell 5 to reflect the increase in the lift curve slope in Fig. 5.31. Nevertheless, this example shows that acceptable local models were obtained even without a large amount of data following the final split. Recall also that global EV values were used throughout the modeling work for local model parameter estimation, so the bias term is not defined as the reference value at the center of each cell.

This physical insight into the aerodynamics can be used for explanatory purposes, local linear control law design, envelope expansion, or other purposes. Since it is being provided and updated in real time, it also offers significant time-saving advantages and model-based vehicle autonomy facilitation. The modularity also provides robustness to poor modeling data in certain parts of the flight envelope, as

well as localized insight into model performance and uncertainty.

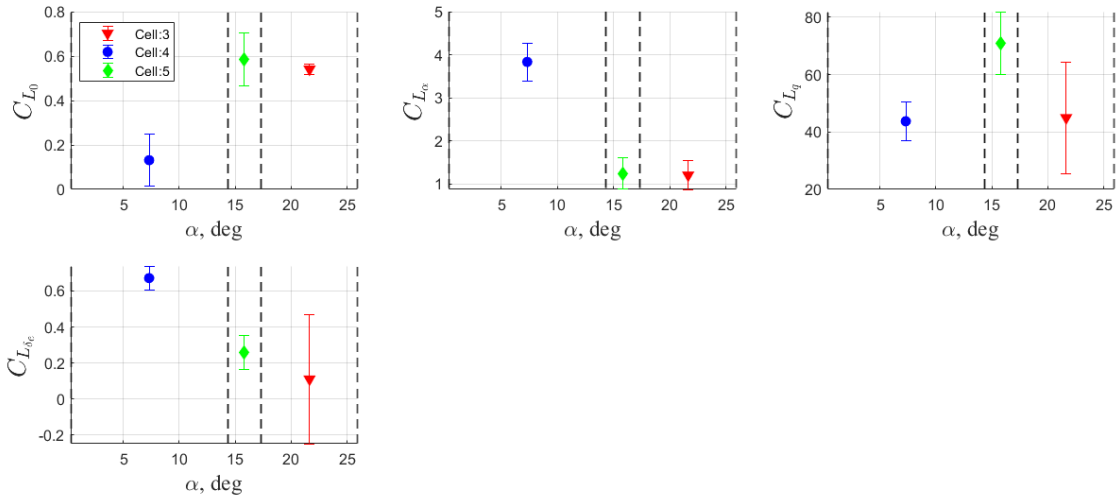


Figure 5.33: Local model parameter estimates across all cells for T-2 C_L model.

The SPLITR model was compared to a single linear model that was developed using all of the flight data, and the RMS of the error is shown in Fig. 5.34. The SPLITR model offers significantly reduced error across the range of α, although with the largest error still in the highest-α region, as it is difficult to model well. It is possible though that further flight data in the highest-α regime or another way of accounting for nonlinear unsteady effects could further improve the model fit there. The SPLITR model provided an R^2 of 0.972, while the linear model had an R^2 of 0.945. Figure 5.35 shows significant reduction in magnitude of the residuals from the SPLITR model compared to the linear model.

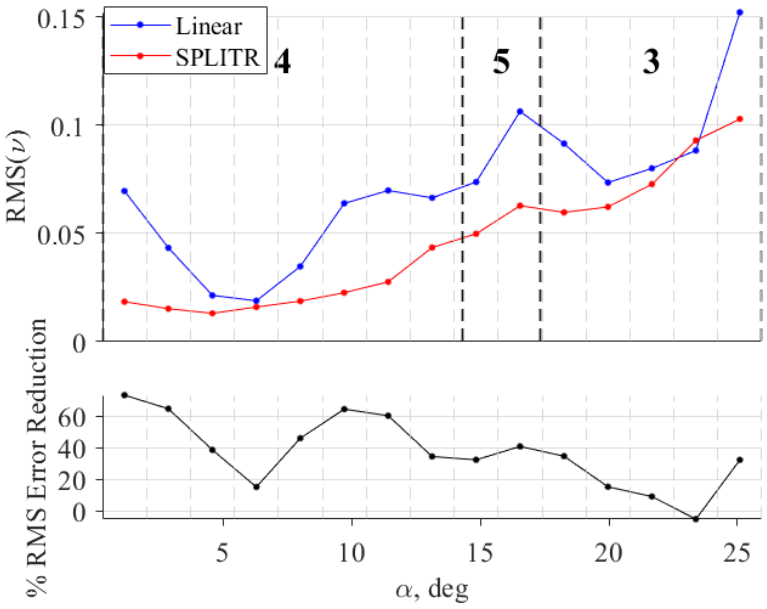
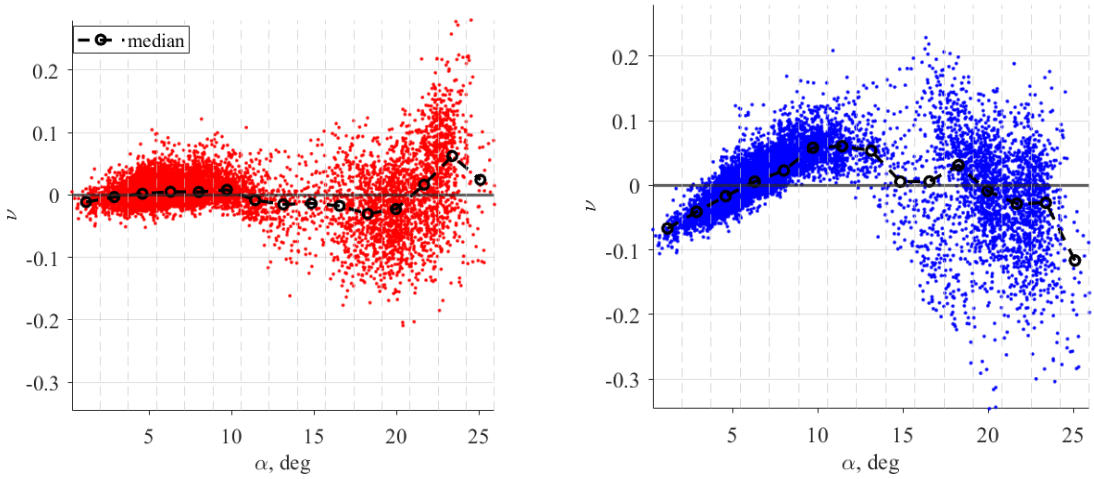


Figure 5.34: RMS of binned global residuals for modeling data for T-2 C_L model.



(a) SPLITR nonlinear model

(b) Linear model

Figure 5.35: Global residuals vs. PV for modeling data for T-2 C_L model.

A separate set of flight data not used in the modeling process, and from a separate flight, was used for model validation. Figure 5.36 shows the time history of the validation data, which includes many high-α and rapid stall maneuvers, as well

as data across the full range of α that was modeled. The RMS of the global residuals is shown in Fig. 5.37, which shows significant reduction in error across the full range of α compared to the linear model. The R^2 for the validation data was 0.935 for the SPLITR model, and 0.810 for the linear model. The plots of the residuals as a function of the PV α in Fig. 5.38 show reduction in magnitude of the residuals in the SPLITR model, similar to the plot in Fig. 5.35 for the modeling data.

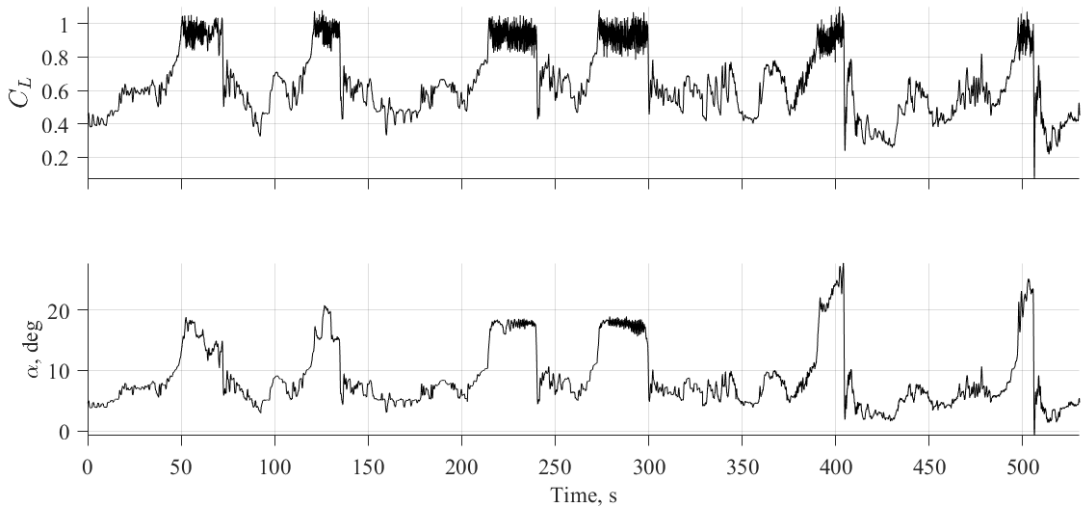


Figure 5.36: Time histories of response variable and PV validation data for T-2 C_L model.

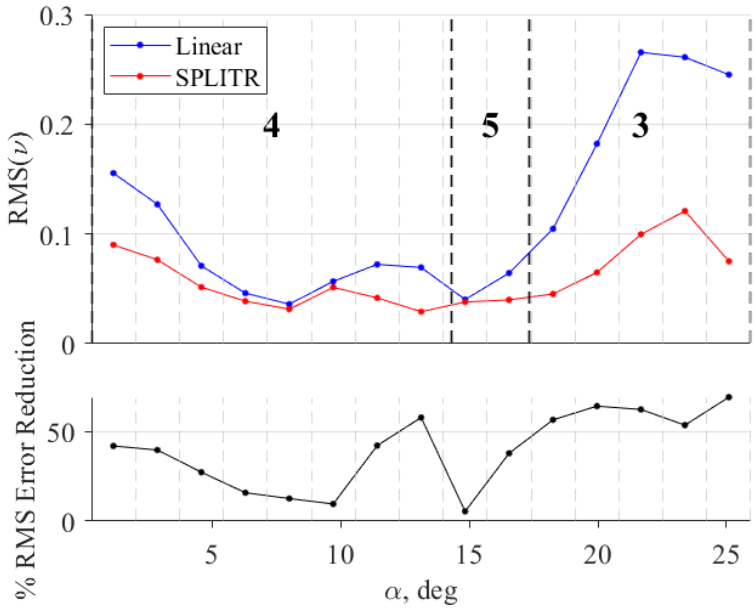
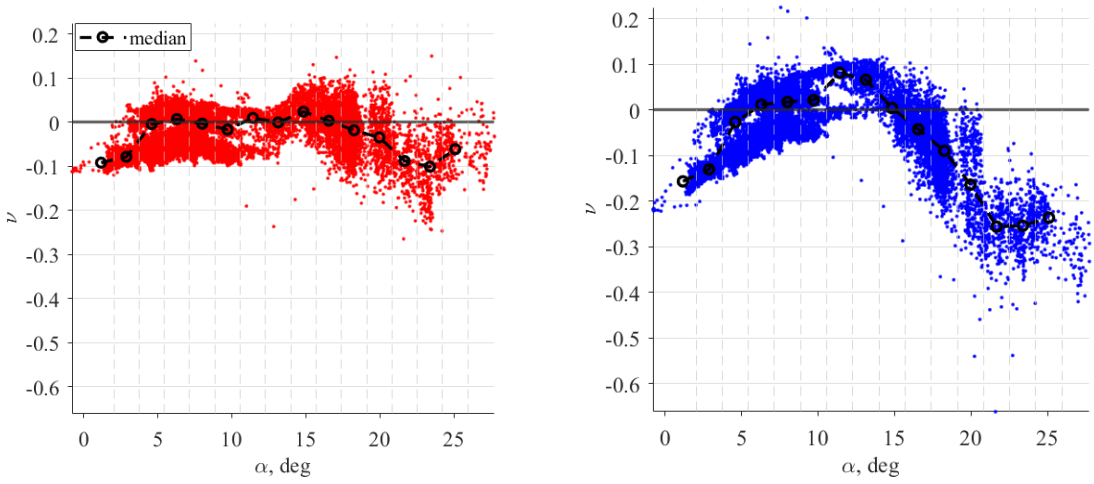


Figure 5.37: RMS of binned global residuals for validation data for T-2 C_L model.



(a) SPLITR nonlinear model

(b) Linear model

Figure 5.38: Global residuals vs. PV for validation data for T-2 C_L model.

The test cases presented in this dissertation were conducted through a real-time simulation on a laptop computer by streaming the data one point at a time. The computation time for each simulation was significantly shorter than the total

duration of data for the 50 Hz data rates assumed and/or used throughout this work. For the T-2 case, which had the largest data set consisting of 5.5 minutes of data, the computation time after the algorithm initialization was around 20 seconds using MATLAB on a Dell Precision 7510 laptop computer. This timing was achieved by removing much of the data logging and other extraneous computation that was used to provide the figures presented for each example, while retaining only the relevant information to evaluate the model. The code was not optimized for improved timing, but future work on improving the efficiency of the code could reduce the computational load even more. Additionally, the code can be compiled and converted to C to improve the algorithm speed. Further testing and validation would be required to ensure it can operate onboard the flight computers used to achieve the flight data for the test cases in this chapter.

5.6 Summary

Following the introduction of the SPLITR modeling method in Chapter 3, and exploring its utility and versatility using simplified test data in Chapter 4, this chapter culminated in applying the new SPLITR approach to aerodynamic modeling using simulated and experimental flight data. First the modeling process was discussed, particularly in the context of conventional fixed-wing aerodynamic modeling, to describe how the modeling data are obtained. Then the F-16 simulation was used as a straightforward initial aircraft example to demonstrate how the SPLITR method partitions the familiar lift curve, and to tie the results to previous related

plots shown in Chapter 3.

The E1 flight test vehicle was then introduced, and two case studies were presented to show the SPLITR modeling results for the C_m and C_X flight data. The results from the SPLITR models were compared to those from a single linear model to demonstrate the improvements in the model fit achieved by the localization. Even with only linear regressors and one-dimensional splits, SPLITR produced models with improved accuracy and valuable insight. Finally, the T-2 test aircraft was described, and SPLITR modeling results were discussed for the C_L flight data. These conventional aircraft examples enabled the results to be validated and interpreted in conjunction with aerodynamic expectations. They also showed the robustness of the SPLITR method to experimental data, and the ease of application in congruence with the simplified examples in Chapter 4.

Each of these flight test examples demonstrated that SPLITR can successfully and effectively model nonlinear aerodynamics with real-time partitioning, provide valuable physical insight into the aerodynamics through the partitions and the local parameter estimates, and offer good predictive capabilities, consistent with the goals that were specified at the beginning of this dissertation in Section 1.5. Furthermore, these goals were realized without the need to tune the SPLITR algorithm user specifications across different test cases throughout both Chapter 4 and Chapter 5, and with the use of real experimental flight data that contain significant and unknown noise content.

These examples, and the visualization of the results, are used to emphasize the need for a nonlinear global aerodynamic model, and the ability to capture the

nonlinear effects successfully with several strategically and automatically placed linear models.

Chapter 6

Conclusions

This chapter will summarize the dissertation, discuss the capabilities, advantages, and limitations of the SPLITR modeling method, specifically highlight the novel contributions of this work, and address directions for future research.

6.1 Dissertation summary

The rapidly changing air vehicle environment prompted by novel UAV capabilities introduces a demand for new, innovative, and efficient modeling and control development tools. The NASA L2F concept offers a paradigm for real-time onboard aerodynamic modeling tools that addresses these challenges. Model identification was recognized as the central component of aircraft system development, and a detailed background on system identification tools and processes was then presented. A literature review of aerodynamic modeling methods, outlined in Chapter 1, summarized many current methods, as well as their advantages and limitations, including global nonlinear polynomial modeling, data partitioning, and other approaches

such as neural networks. The specific modeling goals that this work set forth to accomplish included developing a model that offers physical insight into the system, prioritizing both good local and global prediction capabilities, and abiding by the constraints of real-time operation and recursive updates. The field of LMNs was identified as offering an amenable framework for the particular specified modeling goals, but needed to be further developed from the current practices of manual post-processed data partitioning in aerodynamic modeling.

Chapter 2 then explored the underlying theory of LMNs, provided background on various properties, and described many existing LMN construction algorithms. Across the leaf model properties, the global architecture, and the particular construction logic, a wide breadth and flexibility within LMNs was portrayed. The advantages and limitations associated with certain customizations and design choices related to the LMN formulation were detailed and evaluated for the application of real-time and global nonlinear modeling. These discussions motivated and informed the final structure applied in the SPLITR framework.

The novel SPLITR modeling method that was developed and explored in this dissertation was then described in Chapter 3. The particular chosen LMN properties were discussed, from the leaf model attributes through the global weighting method. The unique SPLITR real-time cell structure determination process was developed and described in detail, including the recursive data processing and cell splitting procedures. A detailed explanation of the SPLITR user inputs was then provided, along with insights to inform the specifications.

In Chapter 4, the SPLITR method was explored through a series of simple

simulated test cases that were chosen to convey important SPLITR properties and sensitivities. These results were enhanced by informative visualizations and explanations that clarified how the SPLITR model is developed, and the logic that drives the split decision making. The test data complexity was gradually increased across piecewise linear, quadratic, and cubic test data across one and two dimensions to demonstrate the applicability of SPLITR across these varying model complexities. The sensitivity of the SPLITR results to several user inputs was also demonstrated to provide insight into designating those specifications.

Finally, in Chapter 5, SPLITR was applied to simulated and experimental flight test data for conventional aircraft test vehicles. This chapter expanded the complexity from Chapter 4 by testing SPLITR on data that represent true physical systems with real experimental flight data. Two additional aspects of the results provided by those test cases are the confirmation that SPLITR can effectively process experimental data with unknown and large noise content, and the verification and validation of the results to be consistent with physical engineering expectations. In addition to providing a model with improved predictive capabilities compared to a single linear model, SPLITR also supplied a cell structure that offered valuable physical insight into the local linear aerodynamic regions. The results from Chapter 5 confirmed that the modeling goals specified in Chapter 1 were fulfilled through the SPLITR modeling method.

6.2 SPLITR summary, advantages, and limitations

This dissertation presented a novel approach to automated global nonlinear modeling using local model networks in real time. SPLITR successively partitions complex nonlinear behavior into local regions to develop a global model composed of weighted local models that also accurately captures the local behavior, and provides valuable physical insight and interpretability. This method uses a unique residual characterization scheme based on the noise content in the data. The statistical information of the residuals is binned across the ranges of specified PVs to recursively preserve the information in real time. The result of the partitioning is that the residuals are reduced in magnitude as a function of the PVs. SPLITR provides three sets of informative results: the cell structure, the local models, and the global nonlinear model. The decision-making process inherent in the cell structure evolution can be visualized and understood, and the resulting model can also be analyzed and validated, both globally and locally. The cell structure and model composition offer a modularity that provides robustness to poor modeling data in certain parts of the flight envelope, as well as localized insight into model performance and uncertainty. The model complexity, as characterized by the number of cells, is also inherently restricted due to the reliance on noise estimates to prevent overfitting.

During the model development stage a global model is often desirable, and the examples shown throughout Chapter 5 demonstrated significant nonlinear effects throughout the global envelope, where a single linear model is inadequate. For these cases, strategically and automatically placed partitions can be successfully used to

split the model into meaningful local, linear regions. For unconventional vehicles and other strongly nonlinear systems, the benefits offered by the SPLITR model may be even more appreciable than for these conventional cases.

Model quality is often assessed with global model fit statistics, such as R^2 and RSS. While these metrics offer a general assessment of the model quality, they often fail to provide localized insight into particular regions of the input space where the model may be performing at different quality levels. For example, a model with an R^2 of 0.98 may be accepted as satisfactory, but it is possible there is a large amount of data in one region that is fit well, and little data in another region that are fit poorly. The localization offered by the SPLITR models readily provides localized fit statistics across the input space as a way to gauge the model fit across particular regions.

There were several technical challenges that were faced in this work. Most importantly, the real-time compatibility constraint implied that all of the past data could not be saved to be processed in batch during the model development. As a result, all of the computation and method logic relied on instantaneous measurements and recursive updates to a small number of stored variables. This introduced a challenge that was repeatedly discussed throughout this work, namely balancing the tradeoffs between remembering vs. forgetting information. This became relevant with regards to initializing the child cell model with the parent cell model according to a factor of certainty based on how similar they were expected to be, as well as regarding the forgetting factor and the filter window. SPLITR was initially tested on conventional fixed-wing aircraft, but the L2F concept demanded the aerodynamic

modeling methods to be compatible with other air vehicles. Additionally, this work sought to develop a method that could be agnostic to the particular testbed, and with potential to extend beyond the scope of aerospace.

The current limitations of the SPLITR approach are summarized throughout the remainder of this section. The residual characterization procedure identifies deterministic properties in the residuals, and responds by splitting cells. If the EV and PV pools are inadequate and lack important modeling information, multiple splits can occur without model improvement. Note, however, that all modeling methods are subject to the quality of the input variables, so this is not unique to SPLITR. Ultimately, this modeling approach is designed to be automated, but any *a priori* knowledge of the system behavior can be useful to specifying the algorithm parameters and EV and PV pools to lead to more efficient computation. The data density and data information requirements are also more strict for a partitioning strategy where sufficient information is required, not only in a global sense, but particularly in each local region, to ensure a reasonable local model fit.

Additionally, while the residual information in each bin is recursively updated and used for split decision making, the residuals are technically a function of the *current* model parameters, which are updated with every new measurement. As a result, the residuals from each prior point in time are "outdated" when the parameters are updated next. This is an unavoidable artifact of recursive real-time parameter estimation in general. Since real-time applications are restricted to a limited amount of information at a given time, and must be able to quickly adapt to new information, these methods must balance the tradeoffs of increasing the amount

of data saved, and ensuring quick adaptation to new information. This aspect could also be mitigated by updating the model parameters at a lower rate than the data are received.

The child cell parameter initialization proved to be a significant challenge in real time that led to limitations in the SPLITR logic. Without access to past data that can be used to update parameter estimates, the input space needs to be repeatedly covered to obtain new information and ensure each new local model can be updated following a split. From an experiment design standpoint, this would require additional test data and test time to obtain new data throughout the same regions of the input space.

Although the model structure is transparent and offers improved physical insight into the system with axes-orthogonal splits, strongly coupled nonlinearities can be better resolved by allowing multidimensional axes-oblique splits. However, partitioning through multiple dimensions often requires a nonlinear optimization, which would be impractical to compute in a real time method.

The local parameter estimation is treated independently from the global model weighting process, and as a result, the validity functions are simply overlaid on the local models, offering a global model with parameters that are not optimized. Furthermore, Gaussian normalization for many cells of varying widths can cause undesirable side effects to the shapes of the normalized validity functions.

Finally, the real-time constraint enforced in this work strongly affects the modeling results, and while foregoing the advantages of batch processes and optimization procedures, it is possible that the SPLITR modeling results could be inferior to batch

nonlinear modeling results from the same data. However, since SPLITR is uniquely designed for recursive updates, it would be difficult to compare the results to those from another modeling method that has the advantage of iterative processing.

Despite the significant constraints of real-time operation, which limited the choice of tools and denied access to most past data, SPLITR was developed and demonstrated as a successful nonlinear modeling technique using methods that are compatible with real-time applications.

6.3 Contributions of this work

The central contribution of this work is the development of a novel real-time modeling approach known as SPLITR. This method extended current data partitioning approaches from offline to online applications, and enabled automated data-based partitioning and model development without requiring extensive data analysis. SPLITR relies on a unique new cell split decision making logic that was developed based on automated residual analysis. It also expands upon existing methods by accounting for unknown noise content in the data to set modeling expectations and prevent overfitting.

The benefits of the SPLITR method were described and demonstrated throughout this work, and summarized in the previous section. Test data were simulated and used to explore the SPLITR method, as well as the sensitivity of the results to the user specifications. Finally, using the same user specifications as for the simple simulated test data, the SPLITR method was successfully tested using experimental

flight data that contain an unknown underlying model and noise content.

This work strongly emphasized the benefits of transparency applied to both the final model, as well as the model development. Effective visualization of the model development, decision making, and results were therefore important tools to convey the complete modeling process. Numerous unique figures and diagrams were designed to provide visualizations and understanding of the entire process and results. Some of the depictions of the model development included a time history plot of the response variables with specified split timing to show when a split was triggered; a cell structure evolution plot to show how the input space was successively partitioned; a residual characterization plot to display the residual-based decision making; and a plot of the residual threshold over time in each cell to show how it varied, and ultimately affected the split decision making. The final model was expressed through plots of each parameter estimate across all of the cells to show, for example, how the stability and control derivatives varied across the PV input space; and a single parameter estimate over time to show how the estimates change as a function of time, among others. For the flight test examples, the error from the SPLITR model was also compared to that of a batch linear model through both a plot of the RMS of the error, as well as a depiction of the residuals across the PV. This showed the reduction of the SPLITR model error compared to the linear model. Each of these figures conveyed useful information about the modeling process.

The effectiveness of the SPLITR approach was demonstrated using conventional aircraft, for which traditional understanding of aircraft aerodynamics could be used for validation. This real-time modeling capability has the potential to

improve the efficiency of the aircraft modeling process and enable novel aircraft configurations to be developed and tested more rapidly and flown more reliably. Although SPLITR was motivated by and developed for the purpose of aerodynamic modeling, its potential applications and impact can also extend far beyond the scope of aircraft.

The original contributions of this research are summarized below.

- Developed Smoothed Partitioning with LocalIzed Trees in Real time (SPLITR) — a novel approach to global nonlinear modeling using automated LMNs in real time.

- Extended current data partitioning approaches applied to aerodynamic data to allow automated data-based partitioning and model development without requiring extensive analysis.

- Developed a new approach to data partitioning with variable split locations based on residual analysis that can be used for online applications.

- Described and demonstrated the benefits of SPLITR, which are summarized as follows: This method can successively partition complex nonlinear behavior into local regions to develop a weighted global model that also accurately captures the local behavior, and provides valuable physical insight and interpretability. It provides three sets of informative results: the cell structure, the local models, and the global nonlinear model. The decision making inherent in the cell structure evolution process can be visualized and understood, and the

resulting model can also be analyzed and validated, both globally and locally. Furthermore, the model complexity, as characterized by the number of cells, is restricted based on noise estimates to prevent overfitting and ensure a model that is not overly complex. It also estimates the noise content in the data to set those modeling expectations.

- Showed results of applying SPLITR to simulated test data to demonstrate and visualize the modeling process and to explore the sensitivities of the modeling results to user specifications.

- Developed unique and informative visualizations of the model development and results to convey useful information and offer transparency.

- Identified aerodynamic models using SPLITR for experimental flight test data from the NASA E1 and T-2 test aircraft, validated the results using separate flight data, and showed that the split locations and estimated stability and control derivatives offered physical insight.

6.4 Future directions and applications for SPLITR

The directions for future work are divided into three parts in this section. First, several possible extensions to the SPLITR method are discussed, including automation for user specifications, modifications to the SPLITR logic, expansion of LMN properties, and improvements in validity functions. Second, several additional and alternative uses of the SPLITR model are mentioned, in addition to offering

insight into the physical system. Last, additional applications and testbeds are discussed.

Although SPLITR was developed to operate automatically with minimal *a priori* knowledge needed, there are still several user specifications that are required. These parameters could also potentially be better specified with some prior insight. Many of these user inputs, particularly those that have a strong impact on the results and that may be difficult to predict with no prior data, may be automated in real time based on the data. In particular, the residual threshold factor could be modulated based on a real-time assessment of the residuals and the impact of a split. The split initialization could also be improved by automating the cell initialization proportion through comparing the parent data with the child data to determine how much information to inherit. Each of these improvements, however, would need to be studied to determine what information can be extracted and used in real time.

The SPLITR logic was shown to have the capability of placing the splits in meaningful and effective locations to partition the input space and improve the model fit. There are other possibilities for ways to improve the logic with additional sources of information. For example, instead of relying on bin failure solely for the split decision and location, the decision could also be made based on parameter convergence in each cell. However, this could also curtail splits since a large amount of data in each cell may be required, which could be difficult to obtain in hard-to-reach regions of the input space. A split could also be placed "on hold" until new data are obtained to update the child cell models, although this could require additional memory and complexity.

The simplified LMN properties used in this work could also be modified to allow for greater flexibility. Local nonlinear models could be allowed to reduce the number of cells required for highly nonlinear systems. An automated model structure determination process in each cell can also improve the local modeling capabilities, even if the regressor pool is still restricted to include only the linear terms, but particularly for the inclusion of nonlinear terms. Real-time or post-process pruning methods can also be explored to remove redundant cells that are deemed unnecessary based on comparing the prediction with that of adjacent cells, or based on recognizing similar parameter estimates within the parameter uncertainties. If this is performed in real time, there is an additional challenge of pruning cells with limited available information. Alternatively, this technique may be used after the model has already been developed, if all of the data are stored and available for batch post-processing.

The global weighting technique used in SPLITR overlaid the validity functions on the identified local linear models, thereby providing a global nonlinear model that is not guaranteed to be optimal. The validity functions determine how much of each local model contributes towards the global model at each point in the PV input space. Only Gaussian validity functions were considered in this work, and the standard deviations were determined based solely on the cell width. These Gaussian functions can also be customized based on the predictive capability of each cell and the adjacent neighbors. Alternative weighting functions can also be explored and customized based on the real-time quality assessment of each local model. This approach could weight the local models that are considered more reliable with higher

validity, based on uncertainty metrics computed from the data.

The partitioning offered by SPLITR can also be used for other applications such as input space (or envelope) expansion through model-based experiment design. As the input space is partitioned in real time, an assessment of the model quality in each cell, as well as data density, could be used to either confirm that enough data are obtained in that region, or to inform the experiment design or test guidance to return to obtain additional data. This could have significant impact during wind tunnel tests as well to improve the test efficiency. The linear regions of the input space that do not require high resolution test points could be quickly traversed, while the insight provided by the partitioning would allow the rig to be automatically directed to the regions that require more data to improve the prediction.

The LMN structure and the real-time development are also particularly amenable to control applications. The SPLITR model provides local linear models, which are applicable to gain scheduling-based control development. Parallel distributed compensation (PDC) can be used to automatically design a local linear control law for the local model in each cell, and then weight the control inputs according to the cell validity functions. If the SPLITR model is combined with a PDC method applied in real time, it could take advantage of the cell structure to offer both a modeling and control strategy. More generally, the adaptive nature of the modeling can provide updates to an adaptive control law, and the control gains can be modified in real time accordingly.

The scope of this dissertation included applying SPLITR to simple simulated test data and conventional aircraft in post-flight real-time simulations, but alterna-

tive applications and testbeds can also be considered.

Although SPLITR is a real-time modeling method, the unique logical decision-making processes may also be useful for batch modeling. If the real-time constraints are removed, then SPLITR could be adapted to operate in a more flexible, iterative manner. However, there exist a large number of batch LMN modeling algorithms that in their original formulation and development, were not restricted to real-time operation and limited information. These techniques may therefore be more suited for batch mode operation.

SPLITR was developed under the constraints of real-time operation that would be compatible with onboard model development, but future work could validate this assertion by compiling the software and testing onboard the aircraft.

This dissertation focused on applications to fixed-wing vehicles, but SPLITR has a much wider applicability, and future work could apply the SPLITR method to additional air vehicle testbeds, such as rotorcraft, lift+cruise, tiltwing, and other novel air vehicle configurations. This real-time modeling approach could potentially be used in the fast-paced context of Urban Air Mobility (UAM) to obtain an aerodynamic model efficiently, to mitigate the need for many ground-based tests, to provide much-needed physical insight into these new and unconventional vehicles, and to provide additional safety through adaptable autonomous onboard model-based systems.

Finally, although this work was motivated by the NASA L2F concept for the purpose of aerodynamic modeling, future work could expand the test cases and applications to other physical systems with similar modeling goals and priorities.

Bibliography

[1] *Advancing Aerial Mobility: A National Blueprint.* National Academies of Sciences, Engineering, and Medicine, National Academies Press, 2020.

[2] "Mobility on demand." `https://webcms.pima.gov/UserFiles/Servers/Server_6/File/Government/Transportation/ATDM-ActiveTransportationDemandMgmt/200127%20USDOT%20MobilityonDemand%20Fact%20sheet.pdf`, US Department of Transportation. Accessed: 2021-04-06.

[3] M. D. Moore, "Personal Air Vehicles: A Rural/Regional and Intra-Urban On-Demand Transportation System," in *AIAA International Air and Space Symposium and Exposition,* AIAA 2003-2646, 2003.

[4] A. Bacchini and E. Cestino, "Electric VTOL Configurations Comparison," in *Aerospace,* Multidisciplinary Digital Publishing Institute, vol. 6, no. 3, 2019. doi: 10.3390/aerospace6030026.

[5] E. H. Heim, E. M. Viken, J. M. Brandon, and M. A. Croom, "NASA's Learn-to-Fly Project Overview," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2018-3307, Atlanta, GA, June 2018. doi: 10.2514/6.2018-3307.

[6] S. E. Riddick, "An Overview of NASA's Learn-to-Fly Technology Development," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2020-0760, Orlando, FL, January 2020. doi: 10.2514/6.2020-0760.

[7] E. A. Morelli, "Real-Time Global Nonlinear Aerodynamic Modeling for Learn-to-Fly," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2016-2010, San Diego, California, January 2016. doi: 10.2514/6.2016-2010.

[8] "Vortex Lattice Utilization," NASA Langley Research Center, NASA SP-405, May 1976.

[9] P. Parikh, W. Engelund, S. Armand, and R. Bittner, "Evaluation of a CFD Method for Aerodynamic Database Development using the Hyper-X Stack Configuration," in *22nd Applied Aerodynamics Conference and Exhibit,* AIAA Paper 2004-5385, Providence, Rhode Island, August 2004. doi: 10.2514/6.2004-5385.

[10] J. Chambers, *Modeling Flight: The Role of Dynamically Scaled Free Flight Models in Support of NASA's Aerospace Programs.* NASA Langley Research Center, NASA SP 2009-575, 2009.

[11] J. M. Brandon and E. A. Morelli, "Nonlinear Aerodynamic Modeling From Flight Data Using Advanced Piloted Maneuvers and Fuzzy Logic," NASA Langley Research Center, NASA TM 2012-217778, October 2012.

[12] J. M. Brandon and E. A. Morelli, "Real-Time Onboard Global Nonlinear Aerodynamic Modeling from Flight Data," *Journal of Aircraft*, vol. 53, no. 5, pp. 1261–1297, September 2016. doi: 10.2514/1.C033133.

[13] S. E. Riddick, R. C. Busan, D. E. Cox, and S. A. Laughter, "Learn-to-Fly Test Setup and Concept of Operations," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2018-3308, Atlanta, GA, June 2018. doi: 10.2514/6.2018-3308.

[14] E. A. Morelli, "Practical Aspects of Real-Time Modeling for the Learn-to-Fly Concept," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2018-3309, Atlanta, GA, June 2018. doi: 10.2514/6.2018-3309.

[15] J. V. Foster, "Autonomous Guidance Algorithms for NASA Learn-to-Fly Technology Development," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2018-3310, Atlanta, GA, June 2018. doi: 10.2514/6.2018-3310.

[16] S. M. Snyder, B. J. Bacon, and E. A. Morelli, "Online Control Design for Learn-to-Fly," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2018-3311, Atlanta, GA, June 2018. doi: 10.2514/6.2018-3311.

[17] E. A. Morelli, "Determining Aircraft Moments of Inertia from Flight Test Data," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2021-1642, January 2021. doi: 10.2514/6.2021-1642.

[18] E. A. Morelli, "Autonomous Real-Time Global Aerodynamic Modeling in the Frequency Domain," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2020-0761, Orlando, FL, January 2020. doi: 10.2514/6.2020-0761.

[19] R. Weinstein and J. E. Hubbard, "Global Aerodynamic Modeling Using Automated Local Model Networks in Real Time," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2020-0762, Orlando, FL, January 2020. doi: 10.2514/6.2020-0762.

[20] S. M. Snyder, "Autopilot Design with Learn-to-Fly," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2020-0763, Orlando, FL, January 2020. doi: 10.2514/6.2020-07632.

[21] E. A. Morelli and V. Klein, *Aircraft System Identification: Theory and Practice.* Williamsburg, VA: Sunflyte Enterprises, 2nd ed., 2016.

[22] O. Nelles, *Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models*. Springer-Verlag, Berlin, Germany, 2002. doi: 10.1007/978-3-662-04323-3.

[23] R. Murray-Smith, *A Local Model Network Approach to Nonlinear Modeling*. PhD thesis, University of Strathclyde, Glasgow, U.K, 1994.

[24] E. A. Morelli and V. Klein, "Application of system identification to aircraft at NASA Langley Research Center," *Journal of Aircraft*, vol. 42, no. 1, pp. 12–25, 2005. doi: 10.2514/1.3648.

[25] R. Jategaonkar, D. Fischenberg, and W. von Gruenhagen, "Aerodynamic Modeling and System Identification from Flight Data–Recent Applications at DLR," *Journal of Aircraft*, vol. 41, no. 4, pp. 681–691, 2004. doi: 10.2514/1.3165.

[26] E. A. Morelli, "Multiple Input Design for Real-Time Parameter Estimation in the Frequency Domain," *IFAC Proceedings Volumes*, vol. 36, no. 16, pp. 639–644, 2003. doi: 10.1016/S1474-6670(17)34833-4.

[27] E. A. Morelli, "Flight Test Experiment Design for Characterizing Stability and Control of Hypersonic Vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 3, pp. 949–959, May 2009. doi: 10.2514/1.37092.

[28] E. A. Morelli, "Flight Test Maneuvers For Efficient Aerodynamic Modeling," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2011-6672, Portland, Oregon, August 2011. doi: 10.2514/6.2011-6672.

[29] E. A. Morelli, "Efficient Global Aerodynamic Modeling from Flight Data," in *50th AIAA Aerospace Sciences Meeting,* AIAA Paper 2012-1050, Nashville, Tennessee, January 2012. doi: 10.2514/6.2012-1050.

[30] M. B. Tischler and R. K. Remple, *Aircraft and Rotorcraft System Identification*. Reston, VA: AIAA, 2nd ed., 2012.

[31] L. Ljung, "Frequency Domain Versus Time Domain Methods in System Identification–Revisited," in *Control of Uncertain Systems: Modelling, Approximation, and Design*, pp. 277–291, Springer, 2006.

[32] L. Ljung and K. Glover, "Frequency Domain Versus Time Domain Methods in System Identification," *Automatica*, vol. 17, no. 1, pp. 71–86, 1981. doi: 10.1016/0005-1098(81)90085-6.

[33] R. E. Maine and K. W. Iliff, "Application of Parameter Estimation to Aircraft Stability and Control: The Output-Error Approach," NASA, NASA RP-1168, 1986.

[34] J. A. Grauer and M. J. Boucher, "Aircraft System Identification from Multisine Inputs and Frequency Responses," *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 12, pp. 2391–2398, December 2020. doi: 10.2514/1.G005131.

[35] S. Seher-Weiss, User's Guide FITLAB - Parameter Estimation Using MATLAB - Version 2.3. DLR Internal Report. DLR-IB 111-2014/07, 73 S, 2014.

[36] R. V. Jategaonkar, *Flight Vehicle System Identification: A Time-Domain Methodology*. Reston, VA: AIAA Progress in Aeronautics and Astronautics, vol. 247, 2nd ed., 2015. doi: 10.2514/4.102790.

[37] K. Hui, J. Ricciardi, K. Ellis, and D. Tuomey, "Beechjet Flight Test Data Gathering and Level-D Simulator Aerodynamic Mathematical Model Development," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2001-4012, Montreal, Canada, August 2001. doi: 10.2514/6.2001-4012.

[38] R. V. Jategaonkar, W. Mönnich, D. Fishenberg, and B. Krag, "Identification of C-160 Simulator Data Base from Flight Data," in *Proceedings of the 10th IFAC Symposium on System Identification*, pp. 1031–1038, Elsevier Sciences, Oxford, 1994. doi: 10.1016/S1474-6670(17)47844-X.

[39] M. Millidere, U. Cakin, and T. Yigit, "Generating Aerodynamic Database from Closed Loop Simulated Flight Data," in *8th European Conference for Aeronautics and Space Sciences (EUCASS)*, July 2019. doi: 10.13009/EUCASS2019-742.

[40] M. B. Tischler and E. L. Tobias, "A Model Stitching Architecture for Continuous Full Flight-Envelope Simulation of Fixed-Wing Aircraft and Rotorcraft from Discrete Point Linear Models," tech. rep., US Army Aviation and Missile Research, Development and Engineering Center, Redstone, Alabama, 2016.

[41] E. Tobias, M. Tischler, T. Berger, and S. G. Hagerott, "Full Flight-Envelope Simulation and Piloted Fidelity Assessment of a Business Jet Using a Model Stitching Architecture," in *AIAA Modeling and Simulation Technologies Conference,* AIAA Paper 2015-1594, Kissimmee, Florida, January 2015. doi: 10.2514/6.2015-1594.

[42] L. T. Nguyen, M. E. Ogburn, W. P. Gilbert, K. S. Kibler, P. W. Brown, and P. L. Deal, "Simulator Study of Stall/Post-Stall Characteristics of a Fighter Airplane With Relaxed Longitudinal Static Stability," NASA Langley Research Center, NASA TP 1538, 1979.

[43] M. S. Selig, "Real-Time Flight Simulation of Highly Maneuverable Unmanned Aerial Vehicles," *Journal of Aircraft*, vol. 51, no. 6, pp. 1705–1725, 2014. doi: 10.2514/1.C032370.

[44] C. C. de Visser, J. A. Mulder, and Q. P. Chu, "Global Aerodynamic Modeling with Multivariate Splines," in *AIAA Modeling and Simulation Technologies Conference,* AIAA Paper 2008-7500, Honolulu, Hawaii, August 2008. doi: 10.2514/6.2008-7500.

[45] J. G. Batterson and V. Klein, "Partitioning of Flight Data for Aerodynamic Modeling of Aircraft at High Angles of Attack," *Journal of Aircraft*, vol. 26, no. 4, pp. 334–339, April 1989. doi: 10.2514/3.45765.

[46] J. G. Batterson, Estimation of Airplane Stability and Control Derivatives From Large Amplitude Longitudinal Maneuvers. NASA Langley Research Center, NASA TM 83185, 1981.

[47] V. Klein, J. G. Batterson, and P. C. Murphy, "Determination of Airplane Model Structure From Flight Data by Using Modified Stepwise Regression," NASA Langley Research Center, NASA TP 1916, 1981.

[48] E. A. Morelli, "Global Nonlinear Aerodynamic Modeling Using Multivariate Orthogonal Functions," *Journal of Aircraft*, vol. 32, no. 2, pp. 270–277, 1995. doi: 10.2514/3.46712.

[49] J. A. Grauer and E. A. Morelli, "Generic Global Aerodynamic Model for Aircraft," *Journal of Aircraft*, vol. 52, no. 1, pp. 13–20, 2015. doi: 10.2514/1.C032888.

[50] V. Klein and J. G. Batterson, "Determination of Airplane Model Structure From Flight Data Using Splines and Stepwise Regression," NASA TP-2126, March, 1983.

[51] C. C. de Visser, J. A. Mulder, and Q. P. Chu, "A Multidimensional Spline-Based Global Nonlinear Aerodynamic Model for the Cessna Citation II," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2010-7950, Toronto, Ontario, Canada, August 2010. doi: 10.2514/6.2010-7950.

[52] S. Seher-Weiss, "Identification of Nonlinear Aerodynamic Derivatives using Classical and Extended Local Model Networks," *Aerospace Science and Technology*, vol. 15, pp. 33–44, 2011. doi: 10.1016/j.ast.2010.06.002.

[53] D. Rohlf, "Global Model Approach for X-31 VECTOR System Identification," *Journal of Aircraft*, vol. 42, no. 1, pp. 54–62, 2005. doi: 10.2514/1.6147.

[54] S. Weiss, H. Friehmelt, E. Plaetschke, and D. Rohlf, "X-31A System Identification Using Single-Surface Excitation at High Angles of Attack," *Journal of Aircraft*, vol. 33, no. 3, pp. 485–490, 1996. doi: 10.2514/3.46970.

[55] S. Weiss and F. Thielecke, "Aerodynamic Model Identification Using Local Model Networks," in *AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2000-4098, Denver, Colorado, August 2000. doi: 10.2514/6.2000-4098.

[56] S. Weiss, D. Rohlf, and E. Plaetschke, "Parameter Identification for X-31A at High Angles of Attack," in *Fourth High Alpha Conference*, vol. 1, NASA Dryden Research Center, 1994.

[57] R. Hess, "On the Use of Back Propagation with Feed-Forward Neural Networks for the Aerodynamic Estimation Problem," in *Flight Simulation and Technologies,* AIAA Paper 1993-3638, Monterey, California, August 1993. doi: 10.2514/6.1993-3638.

[58] T. Rajkumar and J. E. Bardina, "Prediction of Aerodynamic Coefficients Using Neural Networks for Sparse Data," in *FLAIRS Conference*, pp. 242–246, May 2002.

[59] D. J. Linse and R. F. Stengel, "Identification of Aerodynamic Coefficients Using Computational Neural Networks," *Journal of Guidance, Control, and Dynamics*, vol. 16, no. 6, pp. 1018–1025, November 1993. doi: 10.2514/3.21122.

[60] M. Larsson, P. De Raedt, and M. Hedlund, *Aerodynamic Identification Using Neural Networks*. Linköping University Electronic Press, 1997.

[61] A. Ghosh, S. Raisinghani, and S. Khubchandani, "Estimation of Aircraft Lateral-Directional Parameters Using Neural Networks," *Journal of Aircraft*, vol. 35, no. 6, pp. 876–881, 1998. doi: 10.2514/2.2407.

[62] R. Kumar and A. Ghosh, "Estimation of Aerodynamic Derivatives Using Neural Network Based Method," *IFAC Proceedings Volumes*, vol. 47, no. 1, pp. 897–904, 2014. doi: 10.3182/20140313-3-IN-3024.00057.

[63] R. F. Stengel and D. J. Linse, "System Identification for Nonlinear Control Using Neural Networks," in *Conference on Information Sciences and Systems*, Princeton, New Jersey, March 1990.

[64] D. J. Linse and R. F. Stengel, "A System Identification Model for Adaptive Nonlinear Control," in *1991 American Control Conference*, pp. 1752–1757, IEEE, Boston, Massachusetts, June 1991. doi: 10.23919/ACC.1991.4791685.

[65] Y. Chen, "Modeling of Longitudinal Unsteady Aerodynamics at High Angle-of-Attack Based on Support Vector Machines," in *2012 8th International Conference on Natural Computation*, pp. 431–435, IEEE, 2012. doi: 10.1109/ICNC.2012.6234640.

[66] J. Sanwale and D. J. Singh, "Aerodynamic Parameters Estimation Using Radial Basis Function Neural Partial Differentiation Method.," *Defence Science Journal*, vol. 68, no. 3, pp. 241–250, 2018. doi: 10.14429/dsj.68.11843.

[67] A. Graps, "An Introduction to Wavelets," *IEEE Computational Science and Engineering*, vol. 2, no. 2, pp. 50–61, 1995.

[68] M. Sharma, G. Singh, and R. Gupta, "Application of Wavelet – An Advanced Approach of Transformation," *Advanced Research in Electrical and Electronic Engineering*, vol. 1, no. 1, pp. 28–34, 2014.

[69] S. Mohammadi, M. Sabzeparvar, and M. KArrari, "Aircraft Stability and Control Model Using Wavelet Transforms," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 224, no. 10, pp. 1107–1118, 2010. doi: 10.1243/09544100JAERO717.

[70] X.-s. Gan, J.-s. Duanmu, Y.-b. Meng, and W. Cong, "Wavelet neural network aerodynamic modeling from flight data based on pso algorithm with information sharing and velocity disturbance," *Journal of Central South University*, vol. 20, no. 6, pp. 1592–1601, 2013. doi: 10.1007/s11771-013-1651-3.

[71] O. Nelles, A. Fink, and R. Isermann, "Local Linear Model Trees (LOLIMOT) Toolbox for Nonlinear System Identification," *12th IFAC Symposium on System Identification*, vol. 33, no. 15, pp. 845–850, June 2000. doi: 10.1016/S1474-6670(17)39858-0.

[72] R. Murray-Smith and T. A. Johansen, "Local Learning in Local Model Networks," in *4th International Conference on Artificial Neural Networks*, pp. 40-46, 1995.

[73] R. Shorten and R. Murray-Smith, "Side effects of normalising radial basis function networks," *International Journal of Neural Systems*, vol. 7, no. 2, pp. 167–179, 1996. doi: 10.1142/S0129065796000130.

[74] G. Gregorčič and G. Lightbody, "Nonlinear model-based control of highly nonlinear processes," *Computers and Chemical Engineering*, vol. 34, no. 8, pp. 1268–1281, 2010. doi: 10.1016/j.compchemeng.2010.03.011.

[75] L. Breiman, "Hinging Hyperplanes for Regression, Classification, and Function Approximation," *IEEE Transactions on Information Theory*, vol. 39, no. 3, pp. 999–1013, 1993. doi: 10.1109/18.256506.

[76] S. Ernst, "Hinging Hyperplane Trees for Approximation and Identification," in *Proceedings of the 37th IEEE Conference on Decision and Control*, vol. 2, pp. 1266–1271, IEEE, 1998. doi: 10.1109/CDC.1998.758452.

[77] O. Nelles, "Axes-Oblique Partitioning Strategies for Local Model Networks," in *2006 IEEE International Symposium on Intelligent Control*, pp. 2378–2383, IEEE, 2006. doi: 10.1109/CACSD-CCA-ISIC.2006.4777012.

[78] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*. Wadsworth, 1984.

[79] "Matlab regression tree toolbox." `https://www.mathworks.com/help/stats/regression-trees.html`, 2021. Accessed: 2021-04-08.

[80] A. Karalic, "Linear Regression in Regression Tree Leaves," in *Proceedings of the International School for Synthesis of Expert Knowledge (ISSEK)*, 1992.

[81] P. Chaudhuri, M. Huang, W. Loh, and R. Yao, "Piecewise-Polynomial Regression Trees," *Statistica Sinica*, pp. 143–167, 1994.

[82] W. Loh, "Regression Trees with Unbiased Variable Selection and Interaction Detection," *Statistica Sinica*, pp. 361–386, 2002.

[83] A. Dobra and J. Gehrke, "SECRET: A Scalable Linear Regression Tree Algorithm," in *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 481–487, 2002. doi: 10.1145/775047.775117.

[84] J. H. Friedman, "Multivariate Adaptive Regression Splines," *The Annals of Statistics*, vol. 19, no. 1, pp. 1–67, 1991. doi: 10.1214/aos/1176347963.

[85] O. Nelles and R. Isermann, "Basis Function Networks for Interpolation of Local Linear Models," *Proceedings of 35th IEEE Conference on Decision and Control*, vol. 1, pp. 470–475, 1996. doi: 10.1109/CDC.1996.574356.

[86] O. Bänfer, M. Franke, and O. Nelles, "Adaptive Local Model Networks with Higher Degree Polynomials," in *International Conference on Control, Automation and Systems*, pp. 168–171, IEEE, October 2010. doi: 10.1109/ICCAS.2010.5669893.

[87] O. Bänfer and O. Nelles, "Polynomial Model Tree (POLYMOT)—A New Training Algorithm for Local Model Networks with Higher Degree Polynomials," in *2009 IEEE International Conference on Control and Automation*, pp. 1571–1576, IEEE, 2009. doi: 10.1109/ICCA.2009.5410547.

[88] O. Bänfer, B. Hartmann, and O. Nelles, "POLYMOT versus HILOMOT – A Comparison of Two Different Training Algorithms for Local Model Networks," *IFAC Proceedings Volumes*, vol. 45, no. 16, pp. 1569–1574, 2012. doi: 10.3182/20120711-3-BE-2027.00354.

[89] D. Potts and C. Sammut, "Online Nonlinear System Identification Using Linear Model Trees," *Proceedings of the 16th IFAC World Congress*, pp. 202–207, 2005. doi: 10.3182/20050703-6-CZ-1902.00034.

[90] D. Potts and C. Sammut, "Incremental Learning of Linear Model Trees," *Machine Learning*, vol. 61, pp. 5–48, 2005. doi: 10.1007/s10994-005-1121-8.

[91] C. Hametner and S. Jakubek, "Neuro-Fuzzy Modelling Using a Logistic Discriminant Tree," in *Proceedings of the 2007 American Control Conference*, pp. 864–869, IEEE, 2007. doi: 10.1109/ACC.2007.4283048.

[92] C. Hametner and S. Jakubek, "Combustion Engine Modelling using an Evolving Local Model Network," in *2011 IEEE International Conference on Fuzzy Systems*, pp. 2802–2807, IEEE, 2011. doi: 10.1109/FUZZY.2011.6007357.

[93] C. Hametner and S. Jakubek, "Local model network identification for online engine modelling," *Information Sciences*, vol. 220, pp. 210–225, 2013. doi: 10.1016/j.ins.2011.12.034.

[94] T. Jordan, J. Foster, R. Bailey, and C. Belcastro, "AirSTAR: A UAV Platform for Flight Dynamics and Control System Testing," in *25th AIAA Aerodynamic Measurement Technology and Ground Testing Conference,* AIAA Paper 2006-3307, San Francisco, California, June 2006. doi: 10.2514/6.2006-3307.

[95] E. Morelli and K. Cunningham, "Aircraft Dynamic Modeling in Turbulence," in *25th AIAA Atmospheric Flight Mechanics Conference,* AIAA Paper 2012-4650, Minneapolis, Minnesota, August 2012. doi: 10.2514/6.2012-4650.