Western Graduate&PostdoctoralStudies

Western University
Scholarship@Western

Electronic Thesis and Dissertation Repository

October 2015

Generalized Inclusion-Exclusion

Mike W. Ghesquiere
The University of Western Ontario

Supervisor
Stephen M. Watt
The University of Western Ontario

Graduate Program in Computer Science

A thesis submitted in partial fulfillment of the requirements for the degree in Master of Science

© Mike W. Ghesquiere 2015

Follow this and additional works at: https://ir.lib.uwo.ca/etd

Part of the Other Computer Sciences Commons

Recommended Citation
Ghesquiere, Mike W., "Generalized Inclusion-Exclusion" (2015). *Electronic Thesis and Dissertation Repository*. 3262.
https://ir.lib.uwo.ca/etd/3262

This Dissertation/Thesis is brought to you for free and open access by Scholarship@Western. It has been accepted for inclusion in Electronic Thesis and Dissertation Repository by an authorized administrator of Scholarship@Western. For more information, please contact tadam@uwo.ca.

GENERALIZED INCLUSION-EXCLUSION
(Thesis format: Monograph)

by

Mike Ghesquiere

Graduate Program in Computer Science

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science

The School of Graduate and Postdoctoral Studies
The University of Western Ontario
London, Ontario, Canada

© Mike Ghesquiere 2015

Abstract

Sets are a foundational structure within mathematics and are commonly used as a building block for more complex structures. Just above this we have functions and sequences before an explosion of increasingly specialized structures. We propose a re-hanging of the tree with *hybrid sets* (that is, signed multi-sets), as well *hybrid functions* (functions with hybrid set domains) joining the ranks of sequences and functions. More than just an aesthetic change, this allows symbolic manipulation of structures in ways that might otherwise be cumbersome or inefficient. In particular, we will consider simplifying the product and sum of two piecewise functions or block matrices, integrating over hybrid set domains and the convolution of two piecewise interval functions.

Keywords: Symbolic computation, Hybrid set, Signed multi-set, Generalized partition, Piecewise function, Block matrix algebra, Integration, Piecewise convolution

Contents

Abstract **ii**

Acknowlegements **ii**

List of Figures **v**

1 Introduction **1**
1.1 Motivation 1
1.2 Objectives 3
1.3 Related Work 4
1.4 Thesis Outline 5

2 Hybrid Set Theory **6**
2.1 Hybrid Sets 9
2.2 Hybrid Functions 12
2.3 Hybrid Functional Fold 16
2.3.1 Example: *Piecewise functions on generalized partitions* 17
2.4 Pseudo-functions 21
2.4.1 Example: *Piecewise functions revisited* 23

3 Symbolic Block Linear Algebra **26**
3.1 Oriented Intervals 28
3.2 Vector Addition 31
3.3 Higher Dimension Intervals 33
3.4 Matrix Addition 36
3.4.1 Example: *Evaluation at points* 38
3.4.2 Addition with Larger Block Matrices 39
3.5 Matrix Multiplication 40
3.5.1 Example: *Matrix Multiplication Concretely* 43
3.5.2 Multiplication with Larger Block Matrices 46

4 Integration over Hybrid Domains **48**
4.1 The Riemann Integral on k-rectangles 51
4.2 The Lebesgue Integral on Hybrid Domains 52
4.2.1 Example: *Integrating the Irrationals* 56
4.3 Differential Forms 57

4.4 Singular Cubes 61

5 Stokes' Theorem **63**
5.1 Boundary Operator 63
5.1.1 Example: *Boundary of a 1-rectangle* 65
5.1.2 Example: *Boundary of a 3-rectangle* 65
5.2 Chains 67
5.2.1 Example: *Boundary of a boundary* 68
5.3 Stokes' Theorem 70
5.3.1 Example: *Contour Integral* 73

6 Convolution of Piecewise Functions **76**
6.1 Convolution of Piecewise Functions 78
6.2 Hybrid Function Convolution 81
6.2.1 Example: *Hybrid Convolution* 82
6.3 Infinite Intervals 84
6.4 Discrete Convolution 85
6.5 Implementation 87

7 Conclusions **91**

Bibliography **93**

A Convolution with Infinite End-Points **96**

Curriculum Vitae **104**

List of Figures

2.1 Piecewise Rational Function 23

3.1 Possible block overlaps of 2×2 block matrices. 28
3.2 Cartesian product of two 1-rectangles 34

4.1 Riemann Integral 52
4.2 Approximations using simple functions 55

5.1 Unit cube with boundary 66
5.2 Orientations of 2-rectangles 66
5.3 Boundary of a boundary (of a (2-)rectagle) 68
5.4 Contour Integral 74

6.1 Convolution of box signal with itself 76
6.2 Gaussian Blurring 77
6.3 Convolution of “one-piece” functions 79

Chapter 1

Introduction

1.1 Motivation

Some data structures in mathematics are more widely adopted than others and none more than Cantor's notion of sets. The very foundations of mathematics lie in set theory: numbers are defined in terms of sets as are ordered tuples which in turn lead to relations, functions, sequences and from there branches into countless other structures. But this trunk typically omits a satisfactory treatment of several *generalized sets*. For example, it is difficult to begin speaking about *hybrid sets* without immediately punctuating, *"—that is, multi-sets with negative multiplicity"*.

It is a statement of progress that multi-sets have even entered into common mathematical parlance. On the other hand, sequences need no introduction and one can easily represent a multi-set by a sequence: if an element occurs in a multi-set with multiplicity n, ensure that it's in the corresponding sequence n times as well. In this way, sequences are often used *in place of* multi-sets with the added structure of an ordering. Indeed, *"why not?"*, even if the order of elements is not needed, *"surely it can't hurt?"*

Consider the *Fundamental Theorem of Arithmetic*:

> "Every positive integer, except 1, is a product of primes. (...) The standard form of n is unique; apart from rearrangement of factors, n can be expressed as a product of primes in one way only." (Hardy and Wright 1979, p.2-3)

By recognizing the possibility of rearranging factors, the authors implicitly define the "product

of primes" as a sequence rather than, more aptly a multi-set. For iterated, commutative operators (e.g. $\sum, \prod, \bigcap, \bigcup$), the order of terms is irrelevant so then, *"why order terms to begin with?"* Reisig [16] uses multi-sets to define relation nets where "... several individuals of some sort do not have to be distinguished", and moreover "One *should not* be forced to distinguish individuals if one doesn't wish to. This would lead to overspecification" (emphasis added). The same applies here.

Next — and this is a more subtle and subjective point — the empty sequence is not nearly as enshrined as the empty set. Whereas the empty set is often the first thing that comes to mind when searching for trivial cases or counter-examples, the empty sequence is not usually so readily recalled. Despite the empty sequence being just as well-defined as the empty set; it tends to be treated as an aberrant case. In the case of iterated operators over an empty set, it is simply the respective identity. In the case of $\prod$, this is 1, and so 1 *is* a product of primes. It is uniquely, the product of the empty set of primes. Similarly, a sequence containing just one element is still a sequence but this also can be easy to forget. Some definitions will also disregard this to say, "is prime or the product of primes". Whether this is extra specificity is a fault to the authors or simply a reminder to readers, the point remains the same. The perception of sequences can be misleading in ways in which sets are not often misconstrued. Stripped of these qualifications we are left with the more succinct:

> "Every positive integer is the product of a unique multiset of primes."

It goes deeper than a matter of aesthetics. Sets are certainly flexible objects and it is tempting to take a minimal approach to the tools required in one's toolbox. But the algebra of sets, is a fairly restrictive one. Consider the sets $[a, b) = \{x \mid a \leq x < b\}$ and $[b, c) = \{x \mid b \leq x < c\}$. Their union $[a, b) \cup [b, c)$ will depend heavily on the relative ordering of a, b and c. More often, as in integration what one really wants is "$[a, b) + [b, c) = [a, c)$". When Boolean algebra and numeric algebra interface, results can get unnecessarily messy. Hybrid sets will allow us to remove this tension, by committing fully to numeric algebra. In this spirit that we will graft hybrid sets into areas of mathematics where current approaches are unsatisfactory.

1.2 Objectives

This thesis will include and extend the work of [7] on hybrid sets and their applications. In particular we will take as inspiration from two famous identities. Consider the union of two sets A and B. To compute the cardinality, one approach would be to partition the set into three pieces: $(A \setminus B)$, $(B \setminus A)$ and $(A \cap B)$. The union, as the disjoint union of these three sets is then the sum of their cardinalities:

$$|A \cup B| = |A \setminus B| + |B \setminus A| + |A \cap B|$$

The more common approach is to use *the principle of inclusion-exclusion* and instead break $A \cup B$ into the pieces A, B and $(A \cap B)$:

$$|A \cup B| = |A| + |B| - |A \cap B| \tag{1.1}$$

Unlike the first approach, we no longer have a partition of $A \cup B$ in the traditional sense of the term but in many ways, it still behaves like one.

Our other inspiration comes from integration where it is commonly defined that

$$\int_a^b f(x)\,\mathrm{d}x = -\int_b^a f(x)\,\mathrm{d}x \tag{1.2}$$

This allows for, regardless of the ordering of a, b and c:

$$\int_a^c f(x)\,\mathrm{d}x = \int_a^b f(x)\,\mathrm{d}x + \int_b^c f(x)\,\mathrm{d}x$$

One can think of the region from a to c being partitioned into a to b and b to c. But again, this is not a partition in the usual sense. Rather than a union of disjoint subsets, we again use a generalized notion of a partition with sum of *oriented* subsets where we can exclude subsets by inverting the sign.

Reversed intervals within integration are not generally regarded any differently. Inclusion and exclusion, are simply a matter of orientation. On the other hand, when excluding sets, inverted elements are not usually regarded as objects of their own; "$-(A \cap B)$" is not a set.

We will consider several different applications. When matrices are represented in terms of block submatrices, the sum or product can vary depending on operand block sizes. Representing the regions of a block matrix with hybrid sets will allow for all cases to be consolidated into one; no matter how those blocks may overlap. Lebesgue integration relies on measurable sets and also does not naturally support orientation. Here, hybrid sets will allow us to evaluate integrals like $\int_1^0 1_{\mathbb{R}\setminus\mathbb{Q}}\,\mathrm{d}x = -1$ Finally, we will consider the convolution of piecewise interval functions. Typically different equations are used depending on the relative length of intervals but with hybrid sets we can condense this into one general equation and ignore the interval lengths altogether.

Unifying all of this is the argument that hybrid sets are a very useful structure and there are many instances where existing structures ought to be replaced. Hopefully these examples will provide inspiration to the reader to recognize similar situations elsewhere as well.

1.3 Related Work

It is difficult to date the origin of multisets. Although the term itself was coined by De Bruijn in corresponces with Knuth [13], the thought of a "collection of objects that may or may not be distinguished" is as old as tally marks. In regards to the generalization to *signed* multisets, Hailperin [12] suggests that Boole's 1854 *Laws of Thought* [6] is actually a treatise of signed multisets. Whether this was Boole's intent is debateable. Sets with negative membership explicitly began to appear in [23] and were formalized under the name Hybrid sets in Blizard's extensive work with generalized sets [4, 5] Although hybrid set and signed multiset are the most common nomenclature, other names appearing in literature include multiset (specifying positive when speaking of unsigned multisets) [16] and integral multiset [24].

Existing explicit applications of hybrid sets are currently limited. Loeb *et al.* [1, 15] use hybrid sets to generalize several combinatoric identities to negative values. Bailey *et al.* [2] and Banâtre *et al.* [3] have also had success with hybrid sets in chemical programming. Representing a solution is represented as a collection of atoms and molecules, negative multiplicities are treated as "antimatter" . For a deeper overview and systemization of generalized sets, see [18, 20]. These ideas allow symbolic computation on functions defined piecewise [7].

1.4 Thesis Outline

In Chapter 2, the foundations for hybrid sets and functions with hybrid set domains will be laid. This will provide us with formal definitions and some immediate applications to piecewise functions will be presented. Chapter 3 will see hybrid domains applied towards symbolic matrix algebra. Addition has already been considered [7], but this will be extended to multiplication as well. In Chapter 4, hybrid functions will be applied towards integration. We will start from foundations and use hybrid functions to allow for an oriented Lebesgue integral. We will then show how hybrid sets naturally come up in Stokes' theorem with the boundary operator. Finally in Chapter 5, hybrid sets will be applied towards convolution of piecewise functions.

Chapter 2

Hybrid Set Theory

The motivation behind hybrid sets and functions can be traced to wanting a better approach to piecewise functions and closely related to that is a more generalized notion of a partition. In general, we assume a piecewise function $f : P \to X$ will take the form:

$$f(x) = \begin{cases} f_1(x) & x \in P_1 \\ f_2(x) & x \in P_2 \\ \vdots & \vdots \\ f_n(x) & x \in P_n \end{cases} \tag{2.1}$$

where the set $\{P_i\}$ is a partition of P, the domain of f. and each function $f_i : P_i \to X$ is defined over a corresponding part of P. We assume that f_i is fully defined on P_i (*total*) but may be partial on P . Frequently these partitions are not expressed explicitly as sets but rather as conditions. For example, the absolute value function is far more commonly written out as

$$\text{abs}(x) = \begin{cases} -x & x < 0 \\ x & x \geq 0 \end{cases} \qquad \text{rather than} \qquad \text{abs}(x) = \begin{cases} -x & \{x \mid x \in \mathbb{R} \wedge x < 0\} \\ x & \{x \mid x \in \mathbb{R} \wedge x \geq 0\} \end{cases}$$

but using conditions rather than the sets they define can be seen as just a shorthand.

One thing that should be noted is that sometimes there are additional conjuncts bundled into a condition. As pieces of a partition, the sets generated must be disjoint so we must ensure these conditions are mutually exclusive. A common heuristic is to treat these conditions as a series of cascading *else-if* statements. For example *Maple*'s `piecewise` function:

```
piecewise(cond_1, f_1, cond_2, f_2, ..., cond_n, f_n, f_otherwise)
```

will evaluate each `cond_1` in order and when one evaluates to true, the corresponding `f_i` is then evaluated. Finally if all of `cond_i` evaluate to false then `f_otherwise` is used. Hence the partitions corresponding to each `cond_i` is *not* just $\{x \mid \texttt{cond_i}(x)\}$ but instead the set where `cond_i` is true and *all preceding* conditions are also false.

Although the notation used in (2.1) is fine for piecewise functions with only a few terms such as abs, it quickly becomes unwieldy when one wants to add more pieces. So instead we will use the *join* of disjoint function *restrictions*.

Definition Given a function $f : X \to Y$ and any subset of the domain $Z \subset X$, the **restriction of f to Z** is the function $f|_Z : Z \to Y$, such that $f|_Z(x) = f(x)$ for all $x \in Z$.

There are several ways which one could define a join operator; specifically how one deals with intersections. One could favor the first function as Maple does, but we will take the stance that the intersection is poorly defined.

Definition Define $\obar$, the **join** of two functions, f and g by:

$$f \obar g = \begin{cases} f(x) & \text{if } g(x) = \bot \\ g(x) & \text{if } f(x) = \bot \\ \bot & \textit{otherwise} \end{cases} \tag{2.2}$$

Where the bottom element $\bot$ denotes that the function is undefined. As such if $f : X \to Y$ and $g : S \to T$ then the join will be defined not on the union of X and S but on their *symmetric*

difference. Together these definitions allow us to re-write the piecewise function in (2.1) as:

$$f = f|_{P_1} \obar f|_{P_2} \obar \ldots \obar f|_{P_n}$$

Normally, the join operator $\obar$ is not associative and so omitting parentheses could lead to poorly defined expressions. However, when all support sets are pairwise disjoint, then $\obar$ is associative.

Now consider arithmetic of two piecewise functions. Let f and g be two piecewise functions $f = \left(f_1|_{P_1} \obar f_2|_{P_2} \obar f_3|_{P_3}\right)$ and $g = \left(g_1|_{Q_1} \obar g_2|_{Q_2}\right)$. To compute the sum $f + g$ we need to consider each possible intersection of partitions of P and partitions of Q. In this case, the result is generally a piecewise function with 6 terms:

$$\begin{aligned}(f+g) = \; & (f_1+g_1)|_{P_1\cap Q_1} \;\obar\; (f_1+g_2)|_{P_1\cap Q_2} \;\obar\; (f_1+g_3)|_{P_1\cap Q_3} \\ & (f_2+g_1)|_{P_2\cap Q_1} \;\obar\; (f_2+g_2)|_{P_2\cap Q_2} \;\obar\; (f_2+g_3)|_{P_2\cap Q_3}\end{aligned}$$

In specific cases, some terms may be eliminated as the corresponding intersection is empty. In general, assuming no degenerate intersections, the sum of an "n-piece" function and "m-piece" function will give a piecewise function with $n \cdot m$ pieces. This becomes compounded when dealing with more piecewise functions. The sum of b functions each with n pieces results in b^n cases making computation unrealistic in all but the smallest cases.

Another perspective to take is to first find a *common refinement* of $P = \{P_i\}_{i\in I}$ and $Q = \{Q_j\}_{j\in J}$. In the above example we selected the refinement:

$$R = \left\{ (P_1 \cap Q_1),\ (P_2 \cap Q_1),\ (P_3 \cap Q_1),\ (P_1 \cap Q_2),\ (P_2 \cap Q_2),\ (P_3 \cap Q_2) \right\}$$

That is, for each P_i in the partition P, there is a subset of R which will partition P_i (namely $P_i = \{(P_i \cap Q_1), (P_i \cap Q_2)\}$ and so R is a *refinement* of P. Similarly R also a refinement of Q and so we say it is a *common refinement* of P and Q.

Finding common refinements gives a large increase in the number of terms required, in part, due to a restrictive view of partitions. Conceptually, what one wishes out of a partition is a family of subsets which will "sum" up to the original. With Boolean sets, some additional constraints are imposed as we don't have a very good notion of subtraction. However, with more general structures, we can construct true inverse sets to allow for algebraic cancellations. Following the development of [7], over this chapter we will consider such generalized partitions and the structures to support them. We will also see how this allows for us to eliminate the multiplicative increase in terms when flattening an expression containing the sum or product of piecewise functions.

2.1 Hybrid Sets

We consider *hybrid sets* which are a generalization of multi-sets. One can view usual Boolean sets as an indicator function on the universe which maps each member element to 1 and each non-member to 0. A *multi-set* (or *bag*) extends this by allowing multiple copies of the same element. The indicator function of a multi-set would therefore range over $\mathbb{N}_0$, the set of non-negative integers. Hybrid sets take this one step further allowing for an element to occur *negatively many* times as an indicator function over the integers.

Definition Let U be a set, then any function $U \to \mathbb{Z}$ is called a **hybrid set**. We denote the collection of all hybrid sets over an underlining set U by $\mathbb{Z}^U$.

This provides a functional back-end for constructing hybrid sets. However, given the name, we would like our hybrid sets to at least resemble sets. So we introduce the following definitions to better interface with the underlying integer functions.

Definition Let H be a hybrid set. Then we say that $H(x)$ is the **multiplicity** of the element x. We write, $x \in^n H$ if $H(x) = n$. Furthermore we will use $x \in H$ to denote $H(x) \neq 0$ (or equivalently, $x \in^n H$ for $n \neq 0$). Conversely, $x \notin H$ denotes $x \in^0 H$ or $H(x) = 0$. The symbol $\emptyset$

will be used to denote the empty hybrid set for which all elements have multiplicity 0. Finally the **support of a hybrid set** is the (non-hybrid) set supp H, where $x \in \text{supp } H$ if and only if $x \in H$

We will use the notation:

$$H = \left\{\!\!\left\{ x_1^{m_1}, x_2^{m_2}, \dots \right\}\!\!\right\}$$

to describe the hybrid set H where the element x_i has multiplicity m_i. We allow for repetitions in $\{x_i\}$ but interpret the overall multiplicity of an element x_i as the sum of multiplicities among copies. For example, $H = \left\{\!\!\left\{ a^1, a^1, b^{-2}, a^3, b^1 \right\}\!\!\right\} = \left\{\!\!\left\{ a^5, b^{-1} \right\}\!\!\right\}$. The latter will generally be preferred as all a's and b's have been collected together. Such a writing in which $x_i \neq x_j$ for all $i \neq j$ is referred to as a **normalized form** of a hybrid set.

Boolean sets use the operations $\cup$ union, $\cap$ intersection, and $\setminus$ complementation. These correspond to the Boolean point-wise $\vee$ `OR`, $\wedge$ `AND`, and $\neg$ `NOT` operations. That is, for two sets A and B, then $(A \cup B)(x) = A(x) \vee B(x)$. One *could* extend these for hybrid sets using point-wise min and max on multiplicities as in [4, 5, 11, 19] but this is not very natural. Rather, our primitive hybrid set operations should derive from our primitive point-wise operators. When dealing with hybrid sets with multiplicities over the integers, we have the ring $(\mathbb{Z}, +, \times)$. Thus we will define $\oplus$, $\ominus$, and $\otimes$ by point-wise $+$, $-$, and $\times$ respectively.

Definition For any two hybrid sets A and B over a common universe U, we define the operations $\oplus, \ominus, \otimes : \mathbb{Z}^U \times \mathbb{Z}^U \to \mathbb{Z}^U$ such that for all $x \in U$:

$$(A \oplus B)(x) = A(x) + B(x) \tag{2.3}$$

$$(A \ominus B)(x) = A(x) - B(x) \tag{2.4}$$

$$(A \otimes B)(x) = A(x) \cdot B(x) \tag{2.5}$$

We also define, $\ominus A$ as $\emptyset \ominus A$ and for $c \in \mathbb{Z}$:

$$(cA)(x) = c \cdot A(x) \tag{2.6}$$

Definition We say ***A* and *B* are disjoint** if and only if $A \otimes B = \emptyset$

For Boolean sets A and B, disjointness would be defined by $A \cap B = \emptyset$. If we consider these Boolean sets as simply hybrid sets with multiplicity 0 or 1 then the operations $\cap$ and $\otimes$ identically correspond to element-wise `AND`.

From these definitions, we can use hybrid sets to model various objects that would traditionally be described otherwise. For example, any positive rational number can be represented as a hybrid set over the primes.

$$(\mathbb{Z}^{\mathbb{P}}, \oplus) \simeq (\mathbb{Q}_+, \times)$$

For a positive rational number a/b, both a and b being integers will have a prime decomposition: $a = (p_1^{m_1} \cdot p_2^{m_2} \cdot \ldots)$ and $b = (q_1^{n_1} \cdot q_2^{n_2} \cdot \ldots)$ then there is the group isomorphism f given by:

$$f(a/b) = \left\{\!\!\left\{ p_1^{m_1}, p_2^{m_2}, \ldots \right\}\!\!\right\} \ominus \left\{\!\!\left\{ q_1^{n_1}, q_2^{n_2}, \ldots \right\}\!\!\right\}$$

Furthermore, we have the equivalence $ca/cb = a/b$ for free by writing the hybrid set in normalized form (e.g. $2/4 = \left\{\!\!\left\{ 2^1, 2^{-2} \right\}\!\!\right\} = \left\{\!\!\left\{ 2^{-1} \right\}\!\!\right\} = 1/2$). One could also think of the roots and asymptotes of a rational polynomial as a hybrid set over the underlying ring:

$$\frac{(x-2)}{(x-1)^2(x+1)} = \left\{\!\!\left\{ 2^1, 1^{-2}, -1^{-1} \right\}\!\!\right\}$$

Depending on the context, a negative multiplicity could take many different meanings. Rather than attach a single rigid concept, we will keep this flexibly abstract sometimes. In these examples we used the multiplicity as an exponent in other cases it is more aptly a coefficient or as orientation.

All Boolean sets can be trivially converted to hybrid sets. For a set X, this is done simply by taking $H(x) = 1$ if $x \in X$ and $H(x) = 0$ if $x \notin X$. We will often perform this conversion silently by applying hybrid set operators to Boolean sets. The reverse conversion: the reduction of a hybrid set to a Boolean set is not always possible.

Definition Given a hybrid set H over universe U, if for all $x \in U$ $H(x) = 1$ or $H(x) = 0$ then we say that **$H(x)$ is reducible**. If H is reducible then we denote the **reduction of H** by $\mathcal{R}(H)$ as the (non-hybrid) set over U with the same membership.

In Boolean set theory, a partition of X is a collection of subsets of X such that X is a disjoint union of the subsets. When dealing with hybrid sets we no longer have (or rather choose not to use) union but use point-wise sum instead. For disjoint, reducible hybrid sets, point-wise sum and union agree but we will be even more accommodating and allow for *any* family of sets which sum to a hybrid set to be a (generalized) partition.

Definition A **generalized partition P of a hybrid set $H(x)$** is a family of hybrid sets $P = \{P_i\}_{i=1}^n$ such that:

$$H = P_1 \oplus P_2 \oplus \ldots \oplus P_n \tag{2.7}$$

We say that **P is a strict partition of H** if P_i and P_j are disjoint when $i \neq j$.

If a set H is reducible, then strict generalized partitions will correspond to the usual notion of a partition. A traditional partition will be a disjoint cover of H and for disjoint reducible sets, point-wise sum and union agree. Hence $\bigcup_i P_i = \bigoplus_i P_i$. Conversely, if H is reducible and the sum of disjoint hybrid sets, then P_i must all be reducible as well.

This does not hold for a non-strict partition of a reducible hybrid set. Consider the interval $[0, 1]$ as a hybrid set (that is, $H(x) = 1$ if $0 \leq x \leq 1$ and $H(x) = 0$ otherwise). Then $P = \{\ [0, 2],\ \ominus(1, 2]\ \}$ is a generalized partition of H. A generalized partition of a reducible set is strict if and only if each generalized partition is reducible.

2.2 Hybrid Functions

A function is typically defined as a mapping from elements of one set to another set. We will consider functions which have hybrid sets as their domains (but still map to Boolean sets)

which we will call *hybrid functions*. We will define hybrid functions as the collection of all pairs $(x, f(x))$ (i.e. the *graph of the function* f).

Definition For two sets S and T, a hybrid set over their Cartesian product $S \times T$ is called a **hybrid (binary) relation between S and T**. We denote the set of all such hybrid relations by $\mathbb{Z}^{S\times T}$. A **hybrid function from S to T** is a hybrid relation H between S and T such that $(x, y) \in H$ and $(x, z) \in H$ implies $y = z$. We denote the set of all such hybrid functions by $\mathbb{Z}^{S\to T}$.

Again we will need some notation for this to be more usable. We would like to think of a hybrid function not as a mapping from a hybrid set to a Boolean set but rather as a function between two Boolean sets and integer multiplicity attached to the mapping (the "arrow" itself). In this way we partially separate the traditional function and the multiplicity function (given as a hybrid set) and view a hybrid function as their combined object. Formally, given a hybrid set H over U and a function $f : B \to S$ be a function where $B \subseteq U$ and S a set. Then we denote by f^H the hybrid function from B to S defined by:

$$f^H := \bigoplus_{x\in B} H(x) \left\{\!\!\left\{ (x, f(x))^1 \right\}\!\!\right\} \tag{2.8}$$

There is little literature or established use for hybrid functions; their primary use to us will be as something that we can turn back into traditional functions. So as with hybrid sets, we would like a notion of reducibility.

Definition If H is a reducible hybrid set, then **f^H is a reducible hybrid function**. Additionally, if f^H is reducible, we extend $\mathcal{R}$ by:

$$\mathcal{R}(f^H)(x) = f|_{\mathcal{R}(H)}(x) \tag{2.9}$$

Since $\mathcal{R}(H)$ only exists if $H(x)$ is everywhere 0 or 1; $\mathcal{R}(f^H)$ only makes sense when f^H is reducible. Assuming that we end up with a reducible function, hybrid functions will make an excellent primitives to construct piecewise functions. Earlier we used the *join of functions*

to construct piecewise functions from *restricted functions*. The join operator for two hybrid functions is quite trivially defined.

Definition The **join**, $f^F \obar g^G$ of two hybrid functions f^F and g^G is the hybrid relation given by their point-wise sum.

$$f^F \obar g^G = f^F \oplus g^G \tag{2.10}$$

However, we will immediately dispense with using $\obar$ altogether and simply use $\oplus$ in order to prevent confusion between hybrid functional join (e.g. $f^F \oplus g^G$) and traditional functional join (e.g. $f|_F \obar g|_G$). It is important to note that the join operator is closed under hybrid relations but not under hybrid functions. For any two hybrid *functions* the result will be a hybrid *relation* but not necessarily another hybrid function. As with traditional functional join, we must still be wary of overlapping regions but non-disjoint hybrid domains are not nearly as "dangerous". For intersecting regions we do not have to choose between commutativity and associativity, we can have both. In general, all that we can say the result is a hybrid relation but there are many cases where we can guarantee hybrid function status is preserved.

Lemma 2.2.1 *Let A and B be hybrid sets over U and let $f : U \to S$ be a function. Then $f^A \oplus f^B$ is a hybrid function. Moreover,*

$$f^A \oplus f^B = f^{A \oplus B} \tag{2.11}$$

Proof Since f^A and f^B are hybrid functions with a common underlying function f, all elements with non-zero multiplicity in either set will be of the form $(x, f(x))$. As there can be no disagreement between among points for a common function, the pointwise sum must be a hybrid function. For $x \in U$, we have $x \in^n A$ and $x \in^m B$ for some (possibly zero) integers m and n. Hence, $(x, f(x)) \in^m f^A$ and $(x, f(x)) \in^n f^B$ and $(x, f(x)) \in^{(m+n)} (f^A \oplus f^B)$. At the same time, we have $x \in^{(m+n)} (A \oplus B)$ and $(x, f(x)) \in^{(m+n)} f^{A \oplus B}$. Thus we have $f^A \oplus f^B = f^{A \oplus B}$. ∎

Joining a function with itself is not the most interesting construction. Generally piecewise function is desired for it's ability to tie together two *different* functions. If two functions have separate, non-overlapping regions, then our definition is again trivial.

Lemma 2.2.2 *Given two function $f : U \to S$ and $g : U \to S$. The following identity holds if and only if A and B are disjoint:*

$$f^A \oplus g^B = (f|_{suppA} \uplus g|_{suppB})^{A \oplus B} \tag{2.12}$$

Notice here the use of $\uplus$ on the right hand side. Here $\uplus$ is the traditional (non-hybrid) function join as defined in (2.2). $(f \uplus g)$ was undefined for $(\text{supp}(A) \cap \text{supp}(B))$ and so the equality will not hold if A and B are not disjoint. Chaining several sums together we can represent the piecewise function f from (2.1) by:

$$f^P = f^{P_1} \oplus f^{P_2} \oplus \ldots \oplus f^{P_n}$$

Proof Again, as hybrid functions, all elements with non-zero multiplicity of f^A and g^B will be of the forms $(x, f(x))$ or $(x, g(x))$ respectively. Suppose that we have $(x, f(x)) \in^n f^A$ and $(x, g(x)) \in^m g^B$ for disjoint A and B. If $n \neq 0$, then we have $m = 0$ as disjointness implies $A \otimes B = \emptyset$ and so $A(x) \cdot B(x) = m \cdot n = 0$. Similarly if $m \neq 0$ then $n = 0$. Thus if $(x, f(x)) \in f^A$ then $(x, g(x)) \notin g^B$ and vice versa and so $f^A \oplus g^B$ is a hybrid function.

We can then safely construct $(f|_{\text{supp}A} \uplus g|_{\text{supp}B})$ without undefined points as $\text{supp}A \cap \text{supp}B = \emptyset$. For $(x, f(x)) \in^m (f^A \oplus g^B)$ with non-zero m, we have $(f|_{\text{supp}A} \uplus g|_{\text{supp}B})(x) = f(x)$ and $x \in^m A \oplus B$. Similarly for $(x, g(x)) \in^n (f^A \oplus g^B)$ with non-zero n, we have $(f|_{\text{supp}A} \uplus g|_{\text{supp}B})(x) = g(x)$ and $x \in^n A \oplus B$. Thus, $f^A \oplus g^B = (f|_{\text{supp}A} \uplus g|_{\text{supp}B})^{A \oplus B}$. ∎

But disjointness is still too strong of a condition for two hybrid functions to be compatible. The join of two non-disjoint functions may still be a hybrid function even if their respective functions do not agree at *all* points.

Theorem 2.2.3 *For hybrid functions f^A and g^B, $f^A \oplus g^B$ is a hybrid function if and only if for all $x \in supp(A \otimes B)$, we have $f(x) = g(x)$. We say that* ***f^A and g^B are compatible****.*

Proof This follows from the preceeding two lemmas. For $x \in \text{supp}(A \otimes B)$, if $f(x) = g(x)$ then we are simply combining two identical functions as in lemma 2.2.1. On the other hand for $x \notin \text{supp}(A \otimes B)$ then either one or both of $A(x)$ or $B(x)$ are zero. In this case we use lemma 2.2.2. ∎

Compatibility becomes less clear when we begin to consider multiple hybrid functions. For one, compatibility is not associative. Consider the following sequence:

$$(f^H \oplus g^H) \oplus g^{\ominus H} = f^H \oplus (g^H \oplus g^{\ominus H}) = f^H \oplus g^{\emptyset} = f^H$$

We know nothing of the compatibility of f^H and g^H but let us assume that they are not compatible. However even though $(f^H \oplus g^H)$ is a hybrid relation it is compatible with $g^{\ominus H}$. On the other hand, g^H and $g^{\ominus H}$ are clearly compatible as an instance of theorem 2.2.1. The result is a function over the empty set $g^{\emptyset}$ which is compatible with *any* hybrid function.

2.3 Hybrid Functional Fold

Compatibility and reducibility are one way of collapsing a hybrid function to a traditional function. Another approach is to fold or aggregate a hybrid function using some operator. To aid in this we will first introduce some notation for iterated operators.

Definition Let $* : S \times S \to S$ be an operator on S. Then, for $n > 0$, $n \in \mathbb{Z}$ we use $*^n : S \times S \to S$ to denote iterated $*$. So,

$$x *^n y = (\overbrace{((x * y) * y) * \ldots * y)}^{n \text{ times}} \tag{2.13}$$

If $*$ has an identity e_*, then we extend $x *^0 y = x$. If y has inverse z under $*$ then we use $x *^{-1} y$ to denote $x *^1 z$ and extend this for any natural n by $x *^{-n} y$ by $x *^n z$. Finally, we allow $*^n$ to be a unary operator, which we define by $*^n x = e_* *^n x$.

Assuming $*^m$ and $*^n$ are both defined (e.g. if $m, n \leq 0$ then $*$ must be invertible), we have $(x *^m y) *^n y = x *^{m+n} y$. For non-associative groupoids, it may be of interest to instead define $*^T$ for some tree T. For example $*^n$ above is analogous to Haskell's `foldl`. There are applications where `foldr`: $(x * (y * (y * \ldots * y)))$ or a balanced expression tree like `foldt` might be desired. The applications we will be interested in will be over associative group operators and so we will not actually explore this any further.

Definition We say that a hybrid relation $f^A = f_1^{A_1} \oplus f_2^{A_2} \oplus \ldots$ over $T \times S$ is $*$**-reducible** if $(S, *)$ is an abelian semi-group and A is everywhere non-negative or if $(S, *)$ is an abelian group, we allow for for negative A. If f^A is $*$-reducible we define its $*$**-reduction**, $\mathcal{R}_*(f^A) : T \to S$ as a (non-hybrid) function:

$$\mathcal{R}_*(f^A)(x) = \left(*^{A_1(x)} f_1(x) *^{A_2(x)} f_2(x) * \ldots \right)\Big|_{\text{supp}A} \tag{2.14}$$

If f^A is reducible then it is trivially $*$-reducible and we have:

$$\mathcal{R}(f^A) = \mathcal{R}_*(f^A) \tag{2.15}$$

If a hybrid function is already "flattened", then reducing it will do nothing. So clearly $\mathcal{R}_*$ is a projection since it is idempotent (i.e. $\mathcal{R}_*(\mathcal{R}_*(f^A)) = \mathcal{R}_*(f^A)$). Moreover, we can pull restrictions through point-wise sums:

$$\mathcal{R}_*(\mathcal{R}_*(f^F) \oplus \mathcal{R}_*(g^G)) = \mathcal{R}_*(f^F \oplus g^G) \tag{2.16}$$

2.3.1 Example: *Piecewise functions on generalized partitions*

Let $A_1 = [0, a)$, $A_2 = [0, 1] \setminus A_1$, $B_1 = [0, b)$ and $B_2 = [0, 1] \setminus B_1$ with $a, b \in [0, 1]$. $\{A_1, A_2\}$ and $\{B_1, B_2\}$ are two distinct partitions of the set $[0, 1]$. We will use these to define two piecewise

functions f and g which map to some group $f, g : [0, 1] \to G$.

$$f(x) = f_1^{A_1} \oplus f_2^{A_2} = \begin{cases} f_1(x) & x \in A_1 \\ f_2(x) & x \in A_2 \end{cases} \quad \text{and} \quad g(x) = g_1^{B_1} \oplus g_2^{B_2} = \begin{cases} g_1(x) & x \in B_1 \\ g_2(x) & x \in B_2 \end{cases}$$

If one were interested in computing their sum, $(f + g)$, the naive method would be to compute every possible intersection. Over each intersection, one then takes the restriction of the corresponding sub-functions and joins all these terms together. As in,

$$(f + g)(x) = (f_1 + g_1)|_{A_1 \cap B_1} \obar (f_1 + g_2)|_{A_1 \cap B_2} \obar (f_2 + g_1)|_{A_2 \cap B_1} \obar (f_2 + g_2)|_{A_2 \cap B_2}$$

We will take another approach. First, we can partition $[0, 1]$ into the generalized partition $A_1,\ B_1 \ominus A_1,\ B_2$. The source of this particular partition will remain mysterious for now but observe that we can still construct the partitions: $B_1 = (B_1 \ominus A_1) \oplus A_1$ and $A_2 = (B_1 \ominus A_1) \oplus B_2$. And so we can represent f and g from above with a common partition by using:

$$\begin{aligned} f &= f_1^{A_1} \oplus f_2^{(B_1 \ominus A_1) \oplus B_2} = f_1^{A_1} \oplus f_2^{B_1 \ominus A_1} \oplus f_2^{B_2} \\ g &= g_1^{A_1 \oplus (B_1 \ominus A_1)} \oplus g_2^{B_2} = g_1^{A_1} \oplus g_1^{B_1 \ominus A_1} \oplus g_2^{B_2} \end{aligned}$$

Since we have a common partition we can avoid computing pairwise intersections altogether and simply add each sub-function to the corresponding sub-function which shares a partition. Since $\{A_1, B_1, (B_1 \ominus A_1)\}$ is a generalized partition, we will need to flatten the expression back down to get a traditional function. We use $\mathcal{R}_+$ for this so that negative regions properly cancel.

$$(f + g)(x) = \mathcal{R}_+ \left((f_1 + g_1)^{A_1} \oplus (f_2 + g_1)^{B_1 \ominus A_1} \oplus (f_2 + g_2)^{B_2} \right)$$

This equation holds regardless of the relative ordering of a and b. Suppose $x \in A_1 \cap B_1$ Then we have $A_1(x) = 1$ and $(B_1 \ominus A_1)(x) = B_2(x) = 0$. And so $(f + g)(x) = (f_1 + g_1)(x)$.

Similarly, if $x \in B_1 \cap A_2$ or $x \in A_2 \cap B_2$ then only $(B_1 \ominus A_1)(x)$ or $B_2(x)$ will respectively be non-zero. However, if $x \in B_2 \cap A_1$ then we have $A_1(x) = 1$, $(B_1 \ominus A_1)(x) = -1$ and $B_2(x) = 1$. Simplifying this expression yields:

$$(f + g) \;=\; +^1(f_1 + g_1) +^{-1} (f_2 + g_1) +^1 (f_2 + g_2) \;=\; (f_1 + g_2)$$

In the above example only three regions could simultaneously exist. If $a < b$ then $A_1 \cap B_2 = \emptyset$ but if $b < a$ then $A_2 \cap B_1 = \emptyset$. Although it might seem that the three terms are a result of three regions, in the general case where A_1 and B_1 could be arbitrary subsets, we still only have three terms. We will even extend this to any generalized partition. First we will formalize some ideas we have already seen.

Definition A **refinement** of a generalized partition $P = \{P_i\}_{i \in I}$ is another generalized partition $Q = \{Q_j\}_{j \in J}$ such that, for every P_i in P there is a subset of Q: $\{Q_j\}_{j \in J_i}$, $J_i \subseteq J$ such that for some integers $\{a_{ij}\}_{j \in J_i}$

$$P_i = \bigoplus_{j \in J_i} a_{ij} Q_j \tag{2.17}$$

Given a *set* of generalized partitions a **common refinement** is a generalized partition which is a refinement of every partition in the set. A refinement Q of P is **strict** if $\mathrm{supp}(Q) = \mathrm{supp}(P)$.

In our previous example we used $\{A_1, B_1 \ominus A_1, B_2\}$ which was a common refinement of both $\{A_1, A_2\}$ and $\{B_1, B_2\}$. A_1 and B_2 have trivial representations while $A_2 = (B_1 \ominus A_1) \oplus B_2$ and $B_1 = A_1 \oplus (B_1 \ominus A_1)$. Another common refinement would be the trivial $\{A_1, B_1, A_2, B_2\}$ This is less preferable due to not only containing 4 regions instead of 3 but also the point-wise sum is $[0, 1]^2$ instead of the reducible $[0, 1]$.

We can now formally phrase the problem. Let $A = \{A_i\}_{i=1}^n$ and $B = \{B_i\}_{i=1}^m$ be two generalized partitions of a hybrid set U. We wish to find a generalized partition C of U which is a *common*, *strict* refinement of A and B and has minimal cardinality. These conditions can be

summarized into the following system of $n + m + 1$ simultaneous equations:

$$U = \bigoplus_j C_j$$

$$\forall i \in 1 \ldots n : \qquad A_i = \bigoplus_j a_{i,j} C_j$$

$$\forall i \in 1 \ldots m : \qquad B_i = \bigoplus_j b_{i,j} C_j$$

for some integers $a_{i,j}$ and $b_{i,j}$. Since we know that A and B are each separately partitions of U, we can leverage some of their dependencies to construct $\{C_j\}$. For example, A_n can be represented as $U \ominus (A_1 \oplus \ldots \oplus A_{n-1})$. Any A_i or B_i could similarly be represented by the point-wise difference with U. Although we could remove any two pieces from A and B, we will choose to remove A_n and B_n to form a set of $n + m - 1$ pieces to form a linearly independent basis for C. Expressed as a linear system we have:

$$M \cdot \begin{pmatrix} C_1 \\ \vdots \\ C_{n+m-1} \end{pmatrix} = \begin{pmatrix} U \\ A_1 \\ \vdots \\ A_{n-1} \\ B_1 \\ \vdots \\ B_{m-1} \end{pmatrix} \qquad \text{where} \quad M = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ a_{1,1} & a_{1,2} & \cdots & a_{1,n+m-1} \\ \vdots & & & \vdots \\ a_{n-1,1} & a_{n-1,2} & \cdots & a_{n-1,n+m-1} \\ b_{1,1} & b_{1,2} & \cdots & b_{1,n+m-1} \\ \vdots & & & \vdots \\ b_{m-1,1} & b_{m-1,2} & \cdots & b_{m-1,n+m-1} \end{pmatrix}$$

By definition, M is an integer matrix. But we are actually more interested in its inverse M^{-1} as this will give us values for C_i relative to $(U, A_1, \ldots, A_{m-1}, B_1, \ldots, B_{m-1})$. To stay in the realm of hybrid sets, we would also like to enforce that M^{-1} is also an integer matrix. Assuming this, then the determinant of M must be ± 1. Further restricting ourselves to upper triangular matrices we can choose M to be all 1's along the diagonal as well as the top row. Which has

the following inverse:

$$\begin{pmatrix} 1 & \cdots & \cdots & \cdots & 1 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} 1 & -1 & \cdots & \cdots & -1 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & 1 \end{pmatrix}$$

Using this, we find that one choice for C the common refinement for A and B, is:

$$C = \Big\{ \ (U \ominus A_1 \ominus \ldots \ominus A_{n-1} \ominus B_1 \ominus \ldots \ominus B_{n-1}), \quad A_1, A_2, \ldots, A_{n-1}, \quad B_1, B_2, \ldots, B_{m-1} \ \Big\}$$

Finally, to generalize the example from 2.3.1, let $f = f_1^{A_1} \oplus f_2^{A_2} \oplus f_n^{A_n}$ and $g = g_1^{B_1} \oplus \ldots \oplus g_m^{B_m}$ be two piecewise functions over a common domain U. We can express both of these functions in terms of the above common refinement by:

$$\begin{aligned} f &= f_1^{A_1} \oplus \ldots \oplus f_{n-1}^{A_{n-1}} \oplus f_n^{U \oplus B_1 \oplus \ldots \oplus B_{m-1}} \\ g &= g_1^{B_1} \oplus \ldots \oplus g_{m-1}^{B_{m-1}} \oplus g_m^{U \oplus A_1 \oplus \ldots \oplus A_{n-1}} \end{aligned}$$

To compute $(f * g)$ one just needs to collect like terms and encapsulate in a $*$-reduction

$$\begin{aligned} f * g = \ \mathcal{R}_*\Big(& (f_1 * g_m)^{A_1} \oplus \ldots \oplus (f_{n-1} * g_m)^{A_{n-1}} \\ & \oplus (f_n * g_1)^{B_1} \oplus \ldots \oplus (f_n * g_{m-1})^{B_{m-1}} \\ & \oplus (f_n * g_m)^{U \ominus (A_1 \oplus \ldots \oplus A_{n-1} \oplus B_1 \oplus \ldots \oplus B_{m-1})} \Big) \end{aligned} \tag{2.18}$$

2.4 Pseudo-functions

One last detail remains to be settled. So far we have assumed that the sub-functions of f and g are defined over *all* of U. This may not be the case, we would like to only require f_i to at least

be defined over its corresponding part A_i and similarly g_j over B_j. This poses a problem for the previous equation (2.18) when evaluated at say $x \in A_1 \cap B_1$. Then we have the following terms with non-zero multiplicity:

$$(f * g)(x) = \mathcal{R}_* \left((f_1(x) * g_m(x))^1 \oplus (f_n(x) * g_1(x))^1 \oplus (f_n(x) * g_m(x))^{-1} \right)$$

We would like for the $g_m(x)$ in the first term to cancel with the $g_m(x)$ in the third term. Similarly $f_n(x)$ in the second term should cancel with the $f_n(x)$ in the third term leaving only $f_1(x) * g_1(x)$. This requires that $g_m(x)$ and $f_n(x)$ to actually be defined which is more than we'd care to assume. The approach we take instead is to delay the evaluation of functions until cancellations occur. To do this we use a lambda-lifting trick. Instead of having the elements of a hybrid function be pairs containing the input and output of the underlying function we consider them as pairs containing the input and a "function pointer" to the underlying function.

Definition We define a pseudo-function $\widetilde{f}^A$ as:

$$\widetilde{f}^A = \bigoplus_{x \in B} A(x) \left\{\!\!\left\{ (x, f)^1 \right\}\!\!\right\} \tag{2.19}$$

One should notice the similarity between (2.19) and (2.8). The difference is that we have replaced $(x, f(x))$ with the unevaluated (x, f). This formally makes $\widetilde{f}^A$ a hybrid relation over $U \times (U \to S)$ as opposed to a hybrid function over $U \times S$. To evaluate $\widetilde{f}^A$ we map back to f^A and evaluate that. This mapping between $(x, f(x))$ and (x, f) is very natural and we will perform it unceremoniously, often using f^A and $\widetilde{f}^A$ interchangeably. Properties of hybrid functions such as compatibility and reducibility will be lifted to hybrid pseudo-functions by this as well.

2.4.1 Example: *Piecewise functions revisited*

We repeat the example from 2.3.1 but concretely use the following function:

$$f(x) = \begin{cases} (2 - x^2) & -1 \leq x \leq 1 \\ 1/x^2 & \text{otherwise} \end{cases}$$

Graphically, this resembles a bell-shaped curve with no discontinuities or any apparent irregular behavior. This can be seen in the black plot below in Figure 2.1. Although f is defined for all of $\mathbb{R}$, this is not the case for its sub-functions. The plots in red and blue show the behavior of $(2 - x^2)$ and $1/x^2$ respectively outside of their defined ranges in f. Of note, $1/x^2$ (shown in blue) is obviously undefined at 0.

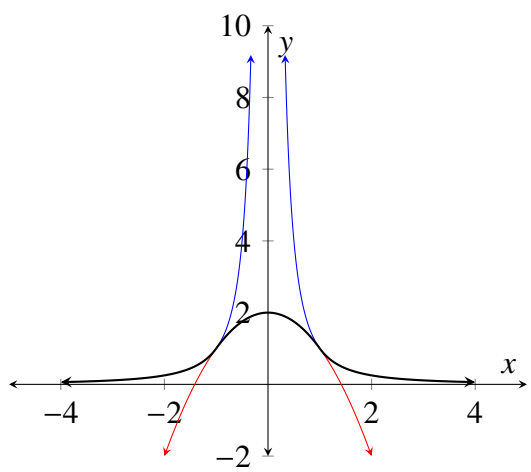


Figure 2.1: A piecewise rational function is shown in black. The plots in red and blue are continuations of $(2 - x^2)$ and $1/x^2$ respectively.

To represent f as a hybrid function, we partition the real line into the sets $A_1 = [-1, 1]$ and $A_2 = \mathbb{R} \setminus [-1, 1] = (-\infty, -1) \oplus (1, \infty)$. This gives the closely related hybrid function f and pseudo-function $\widetilde{f}$:

$$\begin{aligned} f &= \left(2 - x^2\right)^{A_1} \oplus \left(1/x^2\right)^{A_2} \\ \widetilde{f} &= \left(x \mapsto 2 - x^2\right)^{A_1} \oplus \left(x \mapsto 1/x^2\right)^{A_2} \end{aligned}$$

Additionally we will multiply f by the Heaviside function H: a piecewise function over the intervals $B_1 = [0, \infty)$ and $B_2 = (-\infty, 0)$ defined as follows:

$$\begin{aligned} H &= (1)^{B_1} \oplus (0)^{B_2} \\ \widetilde{H} &= (x \mapsto 1)^{B_1} \oplus (x \mapsto 0)^{B_2} \end{aligned}$$

In both the pseudo and non-pseudo function cases, we first need to find a minimal common refinement. As before, we can construct the common refinement $P = \{P_1, P_2, P_3\}$:

$$\begin{aligned} P_1 &= A_1 = [-1, 1] \\ P_2 &= B_1 = [0, \infty) \\ P_3 &= U \ominus (A_1 \oplus B_1) = \mathbb{R} \ominus [-1, 1] \ominus [0, \infty) \end{aligned}$$

To illustrate the problem with using (non-pseudo) hybrid functions, we shall consider $f \cdot H$ evaluated at 0. Since $f(0) = 2$ and $H(0) = 1$ we expect $(f \cdot H)(0) = 2$. Following from (2.18), we construct a hybrid function and *attempt*:

$$\begin{aligned} (f \times H)(0) &= \mathcal{R}_\times \left(\left((2 - x^2)(0)\right)^{P_1} \oplus \left((1/x^2)(1)\right)^{P_2} \oplus \left((1/x^2)(0)\right)^{P_3} \right)(0) \\ &= \mathcal{R}_\times \left(\left((2 - 0^2)(0)\right)^{1} \oplus \left((1/0^2)(1)\right)^{1} \oplus \left((1/0^2)(0)\right)^{-1} \right) \\ &= \frac{(2 - 0^2)(0) \cdot (1)(1) \cdot (0^2)(1)}{(1)(1) \cdot (0^2)(1) \cdot (1)(0)} \end{aligned}$$

But it is not so easy to argue that this evaluates to 2. Alternatively, we could use the very similar pseudo-function:

$$\begin{aligned} \widetilde{f} \cdot \widetilde{H} = \mathcal{R}_\times \Big(& \left((x \mapsto 2 - x^2) \cdot (x \mapsto 0)\right)^{P_1} \\ & \oplus \left((x \mapsto 1/x^2) \cdot (x \mapsto 1)\right)^{P_2} \\ & \oplus \left((x \mapsto 1/x^2) \cdot (x \mapsto 0)\right)^{P_3} \Big) \end{aligned}$$

Leaving these functions unevaluated is the key to making non-total functions work. Once again, we evaluate each of P_1, P_2 and P_3 at 0 and find:

$$\begin{aligned}(\widetilde{f} \cdot \widetilde{H})(0) = \mathcal{R}_\times\Big(&\big((x \mapsto 2 - x^2) \cdot (x \mapsto 0)\big)^1 \\ &\oplus \big((x \mapsto 1/x^2) \cdot (x \mapsto 1)\big)^1 \\ &\oplus \big((x \mapsto 1/x^2) \cdot (x \mapsto 0)\big)^{-1} \Big)(0)\end{aligned}$$

Once we have this, we can then evaluate the multiplication-reduction $\mathcal{R}_\times$, on the still unevaluated functions. Clearly $x \mapsto 0$ occurs with canceling signs as does $x \mapsto 1/x^2$. This leaves us with the product of two unevaluated functions:

$$(\widetilde{f} \cdot \widetilde{H})(0) = \big((x \mapsto 2 - x^2) \cdot (x \mapsto 1)\big)$$

After these cancellations occur, then we can evaluate $(f \cdot H)(x)$ by way of $((\widetilde{f} \cdot \widetilde{H})(x))(x)$:

$$(f \cdot H)(0) \;=\; \Big(\big(\widetilde{f} \cdot \widetilde{H}\big)(0)\Big)(0) \;=\; \big((x \mapsto 2 - x^2) \cdot (x \mapsto 1)\big)(0) \;=\; 2$$

Aside from the introduction of $\mathcal{R}_*$ as a correction to the $\circledast^*$ operation, the material of this chapter can be found entirely in [7]. Given this foundation, the following four chapters will explore new applications for hybrid sets and functions. Matrix addition with block matrices was also explored in [7] as well as [17]. In the following chapter these will be revisited and extended. A novel technique for matrix multiplication will also be presented. The next obvious application for hybrid sets is as oriented domain of integration. We will perform an otherwise standard treatment of integration but for the new usage of hybrid sets and oriented intervals. There are other ways to deal with orientation in Lebesgue integrals but hybrid sets will allow us do so directly without the need to "sanitize" domains. Finally all of our work with integrals and piecewise functions will culminate in chapter 6 with a novel approach to convolution of piecewise functions over symbolic intervals.

Chapter 3

Symbolic Block Linear Algebra

In mathematics literature, it is common practice to represent matrices as being broken up into blocks or sub-matrices. For example if A, B, C and D are matrices given by:

$$A = \begin{bmatrix} A_{1,1} & \dots & A_{1,m} \\ \vdots & & \vdots \\ A_{n,1} & \dots & A_{n,m} \end{bmatrix} \quad B = \begin{bmatrix} B_{1,1} & \dots & B_{1,q} \\ \vdots & & \vdots \\ B_{n,1} & \dots & B_{n,q} \end{bmatrix} \quad C = \begin{bmatrix} C_{1,1} & \dots & C_{1,m} \\ \vdots & & \vdots \\ C_{r,1} & \dots & C_{r,m} \end{bmatrix} \quad D = \begin{bmatrix} D_{1,1} & \dots & D_{1,q} \\ \vdots & & \vdots \\ D_{r,1} & \dots & D_{r,q} \end{bmatrix}$$

Where, critically, A and C have the same width (namely m) as do B and D (width q). Additionally, A and B have the same height (in this case n), as do C and D (height r). Then they can be "glued" together into a single **(2 × 2) block matrix** M. Notationally, we write these matrices as elements of M but we *interpret* M as a sort of concatenation of the sub-matrices.

$$M = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] = \left[\begin{array}{ccc|ccc} A_{1,1} & \dots & A_{1,m} & B_{1,1} & \dots & B_{1,q} \\ \vdots & & \vdots & \vdots & & \vdots \\ A_{n,1} & \dots & A_{n,m} & B_{n,1} & \dots & B_{n,q} \\ \hline C_{1,1} & \dots & C_{1,m} & D_{1,1} & \dots & D_{1,q} \\ \vdots & & \vdots & \vdots & & \vdots \\ C_{r,1} & \dots & C_{r,m} & D_{r,1} & \dots & D_{r,q} \end{array}\right] \tag{3.1}$$

So M is actually a $(n + r) \times (m + q)$ matrix but we write it as a 2×2 *block* matrix. Within an individual block, there is a one-to-one correspondence from the entries of M to the entries of a sub-matrix (shifted by some offset). In the above example, elements of B would have an offset of $(0, m)$ since $B_{i,j}$ corresponds with $M_{i+0,j+m}$.

There is no reason to stop at combining 4 matrices into a 2×2 block matrix. We can take a set of matrices $A_{i,j}$ and combine them into an **$(n \times m)$ block matrix**:

$$M = \left[\begin{array}{c|c|c|c} A_{1,1} & A_{1,2} & \dots & A_{1,m} \\ \hline A_{2,1} & A_{2,2} & \dots & A_{2,m} \\ \hline \vdots & \vdots & & \vdots \\ \hline A_{n,1} & A_{n,2} & \dots & A_{n,m} \end{array}\right]$$

In the 2×2 case, we enforced that for example the height of A was the same as the height of B. Similarly, here it is an important condition that these partitions are divided by *unbroken* horizontal and vertical lines. Formally, for each sub-matrix $A_{i,j}$ in M then $A_{i,j}$ is a $s_i \times t_j$ matrix for strictly positive integer sequences $\{s_i\}_{i=1}^{n}$ and $\{t_j\}_{j=1}^{m}$ common to all sub-matrices. As before we interpret the block matrix as a concatenation of its sub-matrices. Thus M is a $n \times m$ block matrix but a $(\sum_{i=1}^{n} s_i) \times \left(\sum_{j=1}^{m} t_j\right)$ block matrix.

Clearly block matrices are at the very least a convenient notation but they also have considerable practical applications as well. For example when multiplying large matrices, block matrices can be used to improve cache complexity [14]. Additionally, in some cases, when a sub-matrices are known to have a nice properties, many optimizations can arise. For example to invert what is known as a *block diagonal matrix*, one can invert each block individually:

$$\begin{bmatrix} A_1 & 0 & \dots & 0 \\ 0 & A_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & A_n \end{bmatrix}^{-1} = \begin{bmatrix} {A_1}^{-1} & 0 & \dots & 0 \\ 0 & {A_2}^{-1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & {A_n}^{-1} \end{bmatrix} \tag{3.2}$$

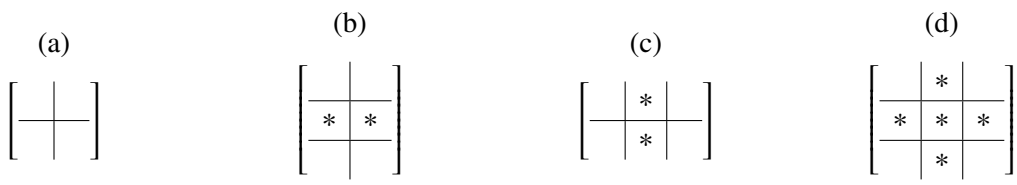


Figure 3.1: The sum of two block matrices each with four blocks leads to 9 possible cases. When blocks are exactly the same size, the sum will also be a 2×2 block matrix (a). Otherwise, a 2×3 (b) *(two possible cases)*, 3×2 (c) *(two cases)* or 3×3 (d) *(four cases)* block matrix could arise. The starred blocks may sample from different blocks depending on the relative size of operand blocks.

Existing techniques allow for fixed block size but are unsatisfactory when the bounds between blocks are symbolic. For example, the sum of 2×2 block matrices, naively leads to 9 possible cases of overlapping regions depending on the relationship between horizontal and vertical boundaries between blocks. In this chapter we will show a method using hybrid functions to avoid this case based approach for both addition and multiplication of matrices. First we introduce some notation that will be used frequently over the next few chapters.

3.1 Oriented Intervals

Definition Given a totally ordered set $(X, \leq)$ *(and with an implied strict ordering $<$)*, for any $a, b \in X$, an **interval between *a* and *b*** is the set of elements in X between a and b, up to inclusion of a and b themselves. Formally:

$$\begin{aligned}
[a, b]_X &= \{x \in X \mid a \leq x \leq b\} \\
[a, b)_X &= \{x \in X \mid a \leq x < b\} \\
(a, b]_X &= \{x \in X \mid a < x \leq b\} \\
(a, b)_X &= \{x \in X \mid a < x < b\}
\end{aligned} \tag{3.3}$$

When context makes X obvious or the choice of X is irrelevant, we shall omit the subscript.

It should be noted that when b is less than a, $[b, a]$ is the empty set. In terms of idempotency, the bounds determine whether or not an interval will be empty. $[a, a]$ which contains a and all

points equivalent to a while (a, a), $(a, a]$, and $[a, a)$ are all empty sets. As intervals are simply sets, they can naturally be interpreted as hybrid sets. If $a \leq b \leq c$, for intervals then we have $[a, b) \oplus [b, c) = [a, c)$. In this case, $\oplus$ seems to behave like concatenation but this is not always true. If instead we had $a \leq c \leq b$ then $[a, b) \oplus [b, c) = [a, b)$.

$$[a, b) \oplus [b, c) = \begin{cases} [a, c) & a \leq b \leq c \\ [a, b) & a \leq c \leq b \\ [b, c) & b \leq a \leq c \\ \emptyset & \text{otherwise} \end{cases}$$

One could alternatively write $[a, b) \oplus [b, c) = [\ \min(a, b), \max(b, c)\ )$ but this simply sweeps the problem under the rug. When working with intervals, a case-based approach to consider relative ordering of endpoints easily becomes quite cumbersome. Previously, the ξ function was introduced in [17] to solve this problem. Although it solves the problem of cases, it quickly leads to unnecessarily heavy notation. Instead we introduce oriented intervals which are considerably more readable. It should be noted that the definitions are equivalent; $\xi(i, y, z)$ and $[\![y, z)\!)$ can be used interchangeably.

Definition We define **oriented intervals** with $a, b \in X$, where X is a totally ordered set, using hybrid set point-wise subtraction as follows:

$$\begin{aligned} [\![a, b)\!) &= [a, b) \ominus [b, a) \\ (\!(a, b]\!] &= (a, b] \ominus (b, a] \\ [\![a, b]\!] &= [a, b] \ominus (b, a) \\ (\!(a, b)\!) &= (a, b) \ominus [b, a] \end{aligned} \tag{3.4}$$

For any choice of *distinct* a and b, exactly one term will be empty; there can be no "mixed" multiplicities from a single oriented interval. Unlike traditional interviews where $[a, b)$ would be empty if $b < a$, the oriented interval $[\![a, b)\!)$ will have elements with negative multiplicity.

Several results follow immediately from this definition.

Theorem 3.1.1 *For all a, b, c,*

$$\begin{aligned} [\![a, b)\!) &= \ominus [\![b, a)\!) \\ (\!(a, b]\!] &= \ominus (\!(b, a]\!] \\ [\![a, b]\!] &= \ominus (\!(a, b)\!) \\ (\!(a, b)\!) &= \ominus [\![a, b]\!] \end{aligned} \tag{3.5}$$

Proof This identities can all be shown in nearly identical fashion

$$\begin{aligned} [\![a, b)\!) &= [a, b) \ominus [b, a) = \ominus\big([b, a) \ominus [a, b)\big) = \ominus [\![b, a)\!) \\ (\!(a, b]\!] &= (a, b] \ominus (b, a] = \ominus\big((b, a] \ominus (a, b]\big) = \ominus (\!(b, a]\!] \\ [\![a, b]\!] &= [a, b] \ominus (b, a) = \ominus\big((b, a) \ominus [a, b]\big) = \ominus (\!(b, a)\!) \end{aligned}$$

And since $[\![a, b]\!] = \ominus (\!(b, a)\!)$ we get $(\!(a, b)\!) = \ominus [\![b, a]\!]$ for free. ∎

We should make a note here how oriented intervals behave when $a = b$. Like their unoriented analogues, the oriented intervals $[\![a, a)\!)$ and $(\!(a, a]\!]$ are still both empty sets. The interval $[\![a, a]\!]$ still contains points equivalent to a (with multiplicity 1). However, unlike traditional intervals $(\!(a, a)\!)$ is *not* empty but rather, $(\!(a, a)\!) = \ominus [\![a, a]\!]$ and so contains all points equivalent to a but with a multiplicity of -1. The advantage of using oriented intervals is that now $\oplus$ does behave like concatenation.

Theorem 3.1.2 *For all a, b, c (regardless of relative ordering),*

$$[\![a, b)\!) \oplus [\![b, c)\!) = [\![a, c)\!) \tag{3.6}$$

Proof Following from definitions we have:

$$\begin{aligned} [\![a, b)\!) \oplus [\![b, c)\!) &= ([a, b) \ominus [b, a)) \oplus ([b, c) \ominus [c, b)) \\ &= ([a, b) \oplus [b, c)) \ominus ([c, b) \oplus [b, a)) \end{aligned}$$

Case 1: $a \leq c$ then $[c, a) = \emptyset$ and so $[\![a, c)\!) = [a, c)$.

Case 1.a: $a \leq b \leq c$ then $[c, b) = [b, a) = \emptyset$ and $[a, b) \oplus [b, c) = [a, c)$

Case 1.b: $b \leq a \leq c$ then $[b, c) \ominus [b, a) = [b, a) \oplus [a, c) \ominus [b, a) = [a, c)$

Case 1.c: $a \leq c \leq b$ then $[a, b) \ominus [c, b) = ([a, c) \oplus [c, b)) \ominus [c, b) = [a, c)$

Case 2: $c < a$ then $[a, c) = \emptyset$ and so $[\![a, c)\!) = \ominus[c, a)$.

Case 2.a: $c \leq b \leq a$ then $[a, b) = [b, c) = \emptyset$ and $\ominus[c, b) \ominus [b, a) = \ominus[c, a)$

Case 2.b: $b \leq c \leq a$ then $\ominus[b, a) \oplus [b, c) = \ominus([b, c) \oplus [c, a)) \oplus [b, c) = \ominus[c, a)$

Case 2.c: $c \leq a \leq b$ then $\ominus[c, b) \oplus [a, b) = \ominus([c, a) \oplus [a, b)) \oplus [a, b) = \ominus[c, a)$ ∎

This sort of reasoning is routine but a constant annoyance when dealing with intervals and is exactly the reason we want to be working with oriented intervals. But now that the above work is done, we can use oriented intervals and not concern ourselves with the relative ordering of points. Many similar formulations such as $[\![a, b]\!] \oplus (\!(b, c)\!) = [\![a, c)\!)$ or $(\!(a, b)\!) \oplus [\![b, c)\!) = (\!(a, c)\!)$ are also valid for any ordering of a, b, c by an identical argument.

3.2 Vector Addition

Addition for partitioned vectors and 2×2 matrices using hybrid functions has already been considered in [17, 7]. The method is nearly identical to that of adding piecewise functions. In fact, one could think of both as simply addition of piecewise functions over a subset of $\mathbb{N}$ and $\mathbb{N} \times \mathbb{N}$ respectively. However it will provide a good example of oriented intervals in use and as an introduction to multiplication of symbolic block matrices.

First we will consider the addition of two n-dimensional vectors. Addition of two vectors: $U = (u_1, u_2, \ldots u_n$ and $V = (v_1, v_2, \ldots, v_n)$ is itself an n dimensional vector defined as:

$$U + V = (u_1 + v_1,\ u_2 + v_2, \ldots, u_n + v_n) \tag{3.7}$$

In particular, we would like to consider the addition of vectors U and V which are each partitioned into two intervals, $[1, k]$ and $(k, n]$ as well as $[1, \ell]$ and $(\ell, n]$. Over each interval, taking the value of different functions, as in:

$$U = [u_1, u_2, \ldots, u_k, u'_1, u'_2, \ldots, u_{n-k}] \tag{3.8}$$

$$V = [v_1, v_2, \ldots, v_\ell, v'_1, v'_2, \ldots, v_{n-\ell}] \tag{3.9}$$

These can be written more concisely as hybrid functions over intervals. Using intervals, these vectors can be represented by hybrid functions over their indices. For example

$$U = (i \mapsto u_i)^{[\![1,k]\!]} \oplus (i \mapsto u'_{i-k})^{(\!(k,n]\!]} \tag{3.10}$$

$$V = (i \mapsto v_i)^{[\![1,\ell]\!]} \oplus (i \mapsto v'_{i-\ell})^{(\!(\ell,n]\!]} \tag{3.11}$$

Although for clarity and succinctness we will use (u_i) instead of $(i \mapsto u_i)$.

$$U = (u_i)^{[\![1,k]\!]} \oplus (u'_{i-k})^{(\!(k,n]\!]} \tag{3.12}$$

$$V = (v_i)^{[\![1,\ell]\!]} \oplus v'_{i-\ell})^{(\!(\ell,n]\!]} \tag{3.13}$$

To add U and V

$$U + V = \left((u_i)^{[\![1,k]\!]} \oplus (u'_{i-k})^{(\!(k,n]\!]}\right) + \left((v_i)^{[\![1,\ell]\!]} \oplus (v'_{i-\ell})^{(\!(\ell,n]\!]}\right) \tag{3.14}$$

$$= \left((u_i)^{[\![1,k]\!]} \oplus (u'_{i-k})^{(\!(k,\ell]\!]} \oplus (u'_{i-k})^{(\!(\ell,n]\!]}\right) + \left((v_i)^{[\![1,k]\!]} \oplus (v_i)^{(\!(k,\ell]\!]} \oplus (v'_{i-\ell})^{(\!(\ell,n]\!]}\right) \tag{3.15}$$

$$= \mathcal{R}_+\left((u_i + v_i)^{[\![1,k]\!]} \oplus (u'_{i-k} + v_i)^{(\!(k,\ell]\!]} \oplus (u'_{i-k} + v'_{i-\ell})^{(\!(\ell,n]\!]}\right) \tag{3.16}$$

The choice to partition $[\![1, n]\!]$ into $[\![1, k]\!] \oplus (\!(k, \ell]\!] \oplus (\!(\ell, n]\!]$ is only one common refinement. We can just as easily use $[\![1, \ell]\!] \oplus (\!(\ell, k]\!] \oplus (\!(k, n]\!]$ to get the equivalent expression:

$$U + V = \mathcal{R}_+\left((u_i + v_i)^{[\![1,\ell]\!]} \oplus (u_i + v'_{i-\ell})^{(\!(\ell,k]\!]} \oplus (u'_{i-k} + v'_{i-\ell})^{(\!(k,n]\!]}\right) \tag{3.17}$$

We must be careful while evaluating these expressions to not forget that $(u'_{i-k}+v_i)$ is actually shorthand for the function:

$$(u'_{i-k} + v_i) = (i \mapsto u'_{i-k}) + (i \mapsto v_i) = (i \mapsto u'_{i-k} + v_i)$$

As a function, it may not be evaluable over the entire range implied in a given term. The same lambda-lifting trick of using pseudo-functions as in the previous section easily solves this.

For example, consider the concrete example where $n = 5$, $k = 4$ and $\ell = 1$ so that $U = [u_1, u_2, u_3, u_4, u'_1]$ and $V = [v_1, v'_1, v'_2, v'_3, v'_4]$. We will also only assume that the functions u_i, u'_i, v_i and v'_i are defined only on the intervals in which they appear (e.g. u_5 is undefined, as is v'_1). Then we have:

$$U + V = (u_i + v_i)^{[\![1,4]\!]} \oplus (u'_{i-4} + v_i)^{(\!(4,1]\!]} \oplus (u'_{i-4} + v'_{i-1})^{(\!(1,5]\!]}$$

None of the individual sub-terms cannot be evaluated directly. In the first term, v_i is not totally defined over the interval $[\![1, 4]\!]$. In the third term, on the interval $(\!(1, 5]\!]$, u'_{i-4} would even evaluated on negative indices. However, these un-evaluable terms also appear in the middle term however the interval $(\!(4, 1]\!]$ is a negatively oriented interval and the offending points cancel exactly as in the previous chapter.

$$\begin{aligned}
U + V &= (u_i + v_i)^{[\![1,1]\!] \oplus (\!(1,4]\!]} \oplus (u'_{i-4} + v_i)^{\ominus (\!(1,4]\!]} \oplus (u'_{i-4} + v'_{i-1})^{(\!(1,4]\!] \oplus (\!(4,5]\!]} \\
&= (u_i + v_i)^{[\![1,1]\!]} \oplus \big((u_i + v_i) - (u'_{i-4} + v_i) + (u'_{i-4} + v'_{i-1})\big)^{[\![1,4]\!]} \oplus (u'_{i-4} + v'_{i-1})^{(\!(4,5]\!]} \\
&= (u_i + v_i)^{[\![1,1]\!]} \oplus (u_i + v'_{i-1})^{(\!(1,4]\!]} \oplus (u'_{i-4} + v'_{i-1})^{(\!(4,5]\!]}
\end{aligned}$$

3.3 Higher Dimension Intervals

Oriented intervals work perfectly well when we are only dealing with the indices of a vector. However, we are more interested in the rectangular blocks of a matrix. We can move from

1-dimensional intervals to 2-dimensional blocks using the Cartesian product

Definition Let $X = \{\!\{ x_1^{m_1}, ..., x_k^{m_k} \}\!\}$ and $Y = \{\!\{ y_1^{n_1}, ..., y_\ell^{n_\ell} \}\!\}$ be hybrid sets over sets S and T We define the **Cartesian product of hybrid sets X and Y**, to be a hybrid set over $S \times T$ and denoted with $\times$ operator as

$$X \times Y = \{\!\{ (x, y)^{m \cdot n} \ : \ x \in^m X, y \in^n Y \}\!\} \tag{3.18}$$

If $[\![a, b]\!]$ and $[\![c, d]\!]$ are both positively oriented intervals in $\mathbb{R}$ then their Cartesian product $[\![a, b]\!] \times [\![c, d]\!]$ is shown in Figure 4.1 is clearly a two dimensional rectangle in $\mathbb{R}^2$. If one of $[\![a, b]\!]$ or $[\![c, d]\!]$ were negatively oriented then we would have a negatively oriented rectangle. If both were negative, then the signs will cancel and the Cartesian product will be *positively* oriented.

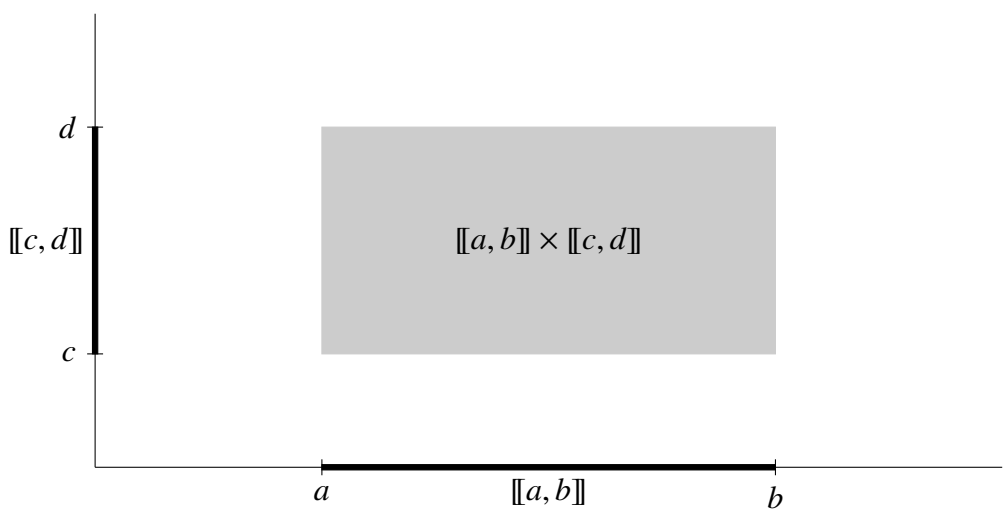


Figure 3.2: The Cartesian product of two positively oriented 1-rectangles $[\![a, b]\!]$ and $[\![c, d]\!]$ is a positively oriented 2-rectangle.

There is no reason to stop here. $[\![a, b]\!] \times [\![c, d]\!]$ is still a hybrid set, we can take its Cartesian product with another interval, say $[\![e, f]\!]$ to get a rectangular cuboid in $\mathbb{R}^3$. We should note here that we do not distinguish between $((x, y), z)$ and $(x, (y, z))$ but rather we treat both as different names for the ordered triple (x, y, z). That is, the Cartesian product is associative, and any

difference in the brackets that arise:

$$\left\{\!\!\left\{ ((x,y),z)^{(m\cdot n)\cdot p} | x \in^m X, y \in^n Y, z \in^p Z \right\}\!\!\right\} = X \times Y \times Z = \left\{\!\!\left\{ (x,(y,z)^{m\cdot(n\cdot p)} | x \in^m X, y \in^n Y, z \in^p Z \right\}\!\!\right\}$$

Although we will not be using them in this chapter, the objects resulting from iterated Cartesian product of intervals turn out to be quite useful. We will call them k-rectangles. A 1-dimensional (non-degenerate) oriented interval will be called a 1-rectangle. A 2-dimensional rectangle will be called a 2-rectangle and a cuboid a 3-rectangle, and so on.

Theorem 3.3.1 *The Cartesian product of a k-rectangle in $\mathbb{R}^m$ (where, $k \leq m$) and ℓ-rectangle in $\mathbb{R}^n$ (again, $\ell \leq n$) is a $(k+\ell)$-rectangle in $\mathbb{R}^{m+n}$.*

For completeness we will also define a 0-rectangle as a hybrid set containing a single point with multiplicity 1 or -1. This allows us to embed k-rectangles in $\mathbb{R}^n$. For example $[\![a,b]\!]_{\mathbb{R}} \times [\![c,d]\!]_{\mathbb{R}} \times \left\{\!\!\left\{ e^1 \right\}\!\!\right\}$ is the product of two 1-rectangles and a 0-rectangle and so it is a 2-rectangle. But it was still a Cartesian product of 3 hybrid sets (each over $\mathbb{R}$) and so is a 2-rectangle in $\mathbb{R}^3$. Specifically, it is the 2-rectangle $[\![a,b]\!] \times [\![c,d]\!]$ on the plane $z = e$. This also illustrates the principle that given a k-rectangle in $\mathbb{R}^n$ where $n > k$ we can always find a k dimensional subspace which also contains the rectangle.

Finally, one last note regarding k-rectangles before we return to the realm of symbolic linear algebra. We will re-use the interval notation and allow for intervals between two vectors: $[\![\boldsymbol{a},\boldsymbol{b}]\!]$. But one should be careful to "type-check" when interpreting. When a and b are real numbers then we continue to use the definition $[\![a,b]\!] = [a,b) \ominus [b,a)$. However, when $\boldsymbol{a}$ and $\boldsymbol{b}$ are n-tuples (for example, coordinates in $\mathbb{R}^n$ then this is *not* the oriented line interval, $[\boldsymbol{a},\boldsymbol{b}) \ominus [\boldsymbol{b},\boldsymbol{a})$ rather we define it as follows:

Definition Let $\boldsymbol{a} = (a_1, a_2, \ldots, a_n)$ and $\boldsymbol{b} = (b_1, b_2, \ldots, b_n)$ be ordered n-tuples then we use the notation:

$$[\![\boldsymbol{a},\boldsymbol{b}]\!] = [\![a_1,b_1]\!] \times [\![a_2,b_2]\!] \times \ldots \times [\![a_n,b_n]\!] \tag{3.19}$$

The dimension of $[\![\boldsymbol{a}, \boldsymbol{b}]\!]$ is equal to the number of indices where a_i and b_i are distinct. For any i where $a_i = b_i$, the corresponding term: $[\![a_i, b_i]\!]$ will be a hybrid set containing a single point, that is, a 0-rectangle. The orientation of $[\![\boldsymbol{a}, \boldsymbol{b}]\!]$ is based on the number of negatively oriented intervals $[\![a_i, b_i]\!]$. Should there be an odd number of indices i such that $a_i > b_i$ then $[\![\boldsymbol{a}, \boldsymbol{b}]\!]$ will also be negatively oriented. Otherwise, it will be positively oriented.

For the remainder of this chapter, we will only be interested in matrices thought of as the space $\mathbb{N}_0 \times \mathbb{N}_0$. Here there is only room for a single Cartesian product and so this notation will not be immediately useful. We will return to this discussion of higher dimension rectangles in Chapter 4 when investigating integration.

3.4 Matrix Addition

Now we will consider the addition of 2×2 block matrices A and B with overall dimensions $n \times m$ of the form:

$$A = \left[\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right] \quad \text{and} \quad B = \left[\begin{array}{c|c} B_{11} & B_{12} \\ \hline B_{21} & B_{22} \end{array}\right]$$

Since these are block matrices then A_{ij} and B_{ij} are not entries but sub matrices themselves. We shall assume that A_{11} is a $(q \times r)$ matrix and B_{11} is a $(s \times t)$ matrix. The sum of A and B will also be a $n \times m$ matrix. Our universe, $\mathcal{U}$ is therefore the the set of all indices in an $n \times m$ matrix:

$$\mathcal{U} \;=\; [\![0, n)\!)_{\mathbb{N}_0} \times [\![0, m)\!)_{\mathbb{N}_0} \;=\; \{(i, j) \mid 0 \leq i < n \text{ and } 0 \leq j < m \text{ and } i, j \in \mathbb{N}_0\}$$

First we must convert A and B to hybrid function notation. We will use $\mathcal{A}_{ij}$ an $\mathcal{B}_{ij}$ to respectively denote the regions for which A_{11} and B_{ij} are defined. Explicitly,

$$\mathcal{A}_{11} = [\![0, q)\!) \times [\![0, r)\!) \quad \mathcal{A}_{12} = [\![0, q)\!) \times [\![r, m)\!) \quad \mathcal{A}_{21} = [\![q, n)\!) \times [\![0, r)\!) \quad \mathcal{A}_{22} = [\![q, n)\!) \times [\![r, m)\!)$$

$$\mathcal{B}_{11} = [\![0, s)\!) \times [\![0, t)\!) \quad \mathcal{B}_{12} = [\![0, s)\!) \times [\![t, m)\!) \quad \mathcal{B}_{21} = [\![s, n)\!) \times [\![0, t)\!) \quad \mathcal{B}_{22} = [\![s, n)\!) \times [\![t, m)\!)$$

Which will allow us to rewrite A and B as:

$$A = A_{11}^{\mathcal{A}_{11}} \oplus A_{12}^{\mathcal{A}_{12}} \oplus A_{21}^{\mathcal{A}_{21}} \oplus A_{22}^{\mathcal{A}_{22}}$$
$$B = B_{11}^{\mathcal{B}_{11}} \oplus B_{12}^{\mathcal{B}_{12}} \oplus B_{21}^{\mathcal{B}_{21}} \oplus B_{22}^{\mathcal{B}_{22}}$$

Depending on the relation of q with s and r with t the regions in the sum of A and B may vary. In Figure 3.1, the shapes of block matrices that can arise are shown. Intuitively, the approach we will take is to not concern ourselves with all possible cases that *could* arise but to just choose one ordering. If this ordering is wrong, then the hybrid function multiplicities will handle cancellations to yield the correct expression regardless.

Since there are 4 partitions in A and 4 partitions in B, we only require 7 pieces to form a common refinement. To this refinement for, we follow the same method as used previously:

$$\left\{ \mathcal{A}_{11},\ \mathcal{A}_{12},\ \mathcal{A}_{21},\ \mathcal{B}_{11},\ \mathcal{B}_{12},\ \mathcal{B}_{21},\ \mathcal{P} \right\} \tag{3.20}$$

with $\mathcal{P}$ is defined as,

$$\mathcal{P} = \mathcal{U} \ominus (\mathcal{A}_{11} \oplus \mathcal{A}_{12} \oplus \mathcal{A}_{21} \oplus \mathcal{B}_{11} \oplus \mathcal{B}_{12} \oplus \mathcal{B}_{21})$$

Clearly we can still express $\mathcal{A}_{22}$ using only the terms from the common refinement by:

$$\begin{aligned}
\mathcal{A}_{22} &= \mathcal{U} \ominus (\mathcal{A}_{11} \oplus \mathcal{A}_{12} \oplus \mathcal{A}_{21}) \\
&= \mathcal{U} \ominus (\mathcal{A}_{11} \oplus \mathcal{A}_{12} \oplus \mathcal{A}_{21} \oplus \mathcal{B}_{11} \oplus \mathcal{B}_{12} \oplus \mathcal{B}_{21}) \oplus \mathcal{B}_{11} \oplus \mathcal{B}_{12} \oplus \mathcal{B}_{21} \\
&= \mathcal{P} \oplus \mathcal{B}_{11} \oplus \mathcal{B}_{12} \oplus \mathcal{B}_{21}
\end{aligned}$$

Similarly $\mathcal{B}_{22}$ can be represented as $\mathcal{B}_{22} = \mathcal{P} \oplus \mathcal{A}_{11} \oplus \mathcal{A}_{12} \oplus \mathcal{A}_{21}$ and $\mathcal{U}$ as the sum of all 7 regions, $\mathcal{U} = \mathcal{A}_{11} \oplus \mathcal{A}_{12} \oplus \mathcal{A}_{21} \oplus \mathcal{B}_{11} \oplus \mathcal{B}_{12} \oplus \mathcal{B}_{21} \oplus \mathcal{P}$. Thus A and B can be rewritten using

this new generalized partition as:

$$A = A_{11}^{\mathcal{A}_{11}} \oplus A_{12}^{\mathcal{A}_{12}} \oplus A_{21}^{\mathcal{A}_{21}} \oplus A_{22}^{\mathcal{P}\oplus\mathcal{B}_{11}\oplus\mathcal{B}_{12}\oplus\mathcal{B}_{21}}$$
$$B = B_{11}^{\mathcal{B}_{11}} \oplus B_{12}^{\mathcal{B}_{12}} \oplus B_{21}^{\mathcal{B}_{21}} \oplus B_{22}^{\mathcal{P}\oplus\mathcal{A}_{11}\oplus\mathcal{A}_{12}\oplus\mathcal{A}_{21}}$$

And addition becomes straightforward. We add functions for terms over corresponding regions. Since we are using *generalized partitions*, not traditional partitions we cannot guarantee disjointness. As such we must also apply a +-reduction after summing each matching pair:

$$\begin{aligned}(A + B) = \mathcal{R}_+ \Big(&(A_{11} + B_{22})^{\mathcal{A}_{11}} \oplus (A_{12} + B_{22})^{\mathcal{A}_{12}} \oplus (A_{21} + B_{22})^{\mathcal{A}_{21}} \\ &\oplus (A_{22} + B_{11})^{\mathcal{B}_{11}} \oplus (A_{22} + B_{12})^{\mathcal{B}_{12}} \oplus (A_{22} + B_{21})^{\mathcal{B}_{21}} \\ &\oplus (A_{22} + B_{22})^{\mathcal{P}} \Big)\end{aligned}$$

3.4.1 Example: *Evaluation at points*

We will now demonstrate evaluating this expression. Let us assume a point (i, j) exists in the region $\mathcal{A}_{11} \cap \mathcal{B}_{12}$. That is, $0 \leq i < \min(q, s)$ and $t \leq j < r$. Evaluating each of the hybrid sets from (3.20) we find that only three have non-zero multiplicities: $\mathcal{A}_{11}(i, j) = 1$, $\mathcal{B}_{12} = 1$ and $\mathcal{P}(i, j) = 1 - (1 + 0 + 0 + 0 + 1 + 0) = -1$. After removing all zero terms, this yields:

$$\begin{aligned}(A + B)(i, j) &= \mathcal{R}_+ \left((A_{11} + B_{22})^1 \oplus (A_{22} + B_{12})^1 \oplus (A_{22} + B_{22})^{-1} \right) \\ &= (A_{11} + B_{22}) + (A_{22} + B_{12}) - (A_{22} + B_{22})\,(i, j) \\ &= (A_{11} + B_{12})(i, j)\end{aligned}$$

As a second example assume $(i, j) \in \mathcal{A}_{22} \cap \mathcal{B}_{12}$. Then we find there is only one partition with non-zero multiplicity. Clearly $\mathcal{B}_{12} = 1$ but $\mathcal{A}_{22} \notin$(3.20). Calculating the multiplicity of $\mathcal{P}$

also yields $1 - (0 + 0 + 0 + 0 + 1 + 0) = 0$. Very simply:

$$\begin{aligned}(A + B)(i, j) &= \mathcal{R}_+\left((A_{22} + B_{12})^1\right)(i, j) \\ &= (A_{22} + B_{12})(i, j)\end{aligned}$$

3.4.2 Addition with Larger Block Matrices

This method extends easily from addition of two 2×2 block matrices to arbitrary addition of block matrices. If we consider (*conformable*) $k \times \ell$ and $n \times m$ block matrices A and B respectively of the form:

$$A = \begin{bmatrix} A_{11} & \dots & A_{1\ell} \\ \vdots & & \vdots \\ A_{k1} & \dots & A_{k\ell} \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} B_{11} & \dots & B_{1m} \\ \vdots & & \vdots \\ B_{n1} & \dots & B_{nm} \end{bmatrix}$$

For matrices to be conformable for additon they must have the same dimensions. So we can partition the rows of A by the strictly increasing sequence $\{q_i\}_{i=0}^k$ and the columns by $\{r_j\}_{j=0}^\ell$. Similarly for B we partition the rows by $\{s_i\}_{i=0}^n$ and the columns by $\{t_j\}_{j=0}^m$. With the additional constraints that $q_0 = r_0 = s_0 = t_0 = 0$ and $q_k = s_n$ and $r_\ell = t_m$. Each A_{ij} and B_{ij} is defined over a rectangular region $\mathcal{A}_{ij}$ and $\mathcal{B}_{ij}$:

$$\mathcal{A}_{ij} = [\![q_{i-1}, q_i)\!) \times [\![r_{j-1}, r_j)\!) \qquad \mathcal{B}_{ij} = [\![s_{i-1}, s_i)\!) \times [\![t_{j-1}, t_j)\!)$$

which gives the expression:

$$(A + B) = \mathcal{R}_+\left(\left(\bigoplus_{(i,j)\neq(n,m)} (A_{ij} + B_{nm})^{\mathcal{A}_{ij}}\right) \oplus \left(\bigoplus_{(i,j)\neq(n,m)} (A_{nm} + B_{ij})^{\mathcal{B}_{ij}}\right) \oplus (A_{nm} + B_{nm})^{\mathcal{P}}\right) \tag{3.21}$$

3.5 Matrix Multiplication

Next we will consider the product of symbolic block matrices. Again, we will assume 2×2 block matrices A and B. However for these matrices to be conformable for multiplication they must be $n \times m$ and $m \times p$ rather than the same size as was required for addition.

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \tag{3.22}$$

Where A_{11} is a $q \times r$ matrix and B_{11} is a $s \times t$ matrix. Note that $0 \leq r, s \leq m$ but the ordering of r and s is unknown.

In the simplest case, $r = s$, four regions will arise each with simple closed expressions.

$$AB = \begin{bmatrix} (A_{11}B_{11} + A_{12}B_{21}) & (A_{11}B_{12} + A_{12}B_{22}) \\ (A_{21}B_{11} + A_{22}B_{21}) & (A_{21}B_{12} + A_{22}B_{22}) \end{bmatrix} \tag{3.23}$$

One should notice the similarity between this and multiplication of simple 2×2 matrices. If we consider only the top-left block, since $r = s$ then the $(q \times r)$ matrix A_{11} and the $(s \times t)$ matrix B_{11} are conformable. As are the $(q \times m - r)$ matrix A_{12} and the $(m - s \times t)$ matrix B_{21}. Both products will result in a $q \times t$ matrix which are conformable for addition. Thus the term $A_{11}B_{11} + A_{12}B_{21}$ is a $q \times t$ block.

If $r \neq s$ then one approach would be to partition A into a 2×3 block matrix split along the vertical lines r and s and the horizontal line q. And split B into a 3×2 block matrix split along the vertical line t and the horizontal lines r and s: Depending on the relative ordering of r and s this may cause different blocks to be split. If $s < r$ then A_{11} and A_{21} will be split into blocks with columns from 0 to s and then from s to r while B_{21} and B_{22} would be split into blocks

with rows from s to r and from r to m.

$$A = \left[\begin{array}{cc|c} A_{11}^{(1)} & A_{11}^{(2)} & A_{12} \\ \hline A_{21}^{(1)} & A_{21}^{(2)} & A_{22} \end{array}\right] \quad \text{and} \quad B = \left[\begin{array}{c|c} B_{11} & B_{12} \\ \hline B_{21}^{(1)} & B_{22}^{(1)} \\ B_{21}^{(2)} & B_{22}^{(2)} \end{array}\right]$$

The resulting product is still a 2×2 matrix. Additionally, each block is still the same size; the first block in the top-left is still $q \times t$. However each block is now the sum of three block products:

$$AB = \begin{bmatrix} \left(A_{11}^{(1)}B_{11} + A_{11}^{(2)}B_{21}^{(1)} + A_{12}B_{21}^{(2)}\right) & \left(A_{11}^{(1)}B_{12} + A_{11}^{(2)}B_{22}^{(1)} + A_{12}B_{22}^{(2)}\right) \\ \left(A_{21}^{(1)}B_{11} + A_{21}^{(2)}B_{21}^{(1)} + A_{22}B_{21}^{(2)}\right) & \left(A_{21}^{(1)}B_{12} + A_{21}^{(2)}B_{22}^{(1)} + A_{22}B_{22}^{(2)}\right) \end{bmatrix}$$

On the other hand, if $r < s$ then A_{12} and A_{22} will be the blocks split vertically while B_{11} and B_{12} will be split horizontally. In turn, this leads to a different expression for the product of A and B. In a now familiar, pattern we can use hybrid functions to give a single expression to deal with all permutations simultaneously.

First we shall refer to the product AB by the block matrix C:

$$AB = C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \tag{3.24}$$

C is an $n \times p$ matrix as determined by the sizes of A and B and C_{11} is a $q \times t$ sub-matrix. This leaves C_{12}, C_{21} and C_{22} to be $q \times (p - t)$, $(n - q) \times t$ and $(n - q) \times (p - t)$ respectively. We will partition all three matrices along the axes $0..n$, $0..p$ and $0..m$ into the oriented intervals:

$$N_1 = [\![0, q)\!) \qquad N_2 = [\![q, n)\!)$$

$$P_1 = [\![0, t)\!) \qquad P_2 = [\![t, p)\!)$$

$$M_1 = [\![0, r)\!) \qquad M_2 = [\![r, s)\!) \qquad M_3 = [\![s, m)\!)$$

Assumption is too strong a word, but these partitions follow the *guess* that $r < s$. So we will be constructing expressions with this in mind. If we chose incorrectly, then we plan to use the negative orientation of M_2 to correct our expression. Using these intervals, we can now rewrite our matrices inline as:

$$A = A_{11}^{N_1 \times M_1} \oplus A_{12}^{N_1 \times (M_2 \oplus M_3)} \oplus A_{21}^{N_2 \times M_1} \oplus A_{22}^{N_2 \times (M_2 \oplus M_3)} \tag{3.25}$$

$$B = B_{11}^{(M_1 \oplus M_2) \times P_1} \oplus B_{12}^{(M_1 \oplus M_2) \times P_2} \oplus B_{21}^{M_3 \times P_1} \oplus B_{12}^{M_3 \times P_2} \tag{3.26}$$

$$C = C_{11}^{N_1 \times P_1} \oplus C_{12}^{N_1 \times P_2} \oplus C_{21}^{N_2 \times P_1} \oplus C_{22}^{N_2 \times P_2} \tag{3.27}$$

It should be noted here that $\oplus$ is still the point-wise sum of hybrid functions. It should not be confused with the direct sum nor the Kronecker sum of matrices which both use the same $\oplus$ operator. The $\times$ operator refers to the Cartesian product of intervals.

For $i, j \in \{1, 2\}$ the terms of C are given by.

$$\begin{aligned} C_{i,j}^{N_i \times P_j}(x, y) = \sum_M \mathcal{R}_\times \Big(& A_{i,1}^{N_1 \times M_1}\Big|_{X=x} \oplus B_{1,j}^{M_1 \times P_1}\Big|_{Y=y} \\ & \oplus A_{i,2}^{N_1 \times M_2}\Big|_{X=x} \oplus B_{1,j}^{M_2 \times P_1}\Big|_{Y=y} \\ & \oplus A_{i,2}^{N_1 \times M_3}\Big|_{X=x} \oplus B_{2,j}^{M_3 \times P_1}\Big|_{Y=y} \Big) \end{aligned} \tag{3.28}$$

There is some new notation here so let us unpack it. Recall that we are taking the approach that matrices are simply functions defined on $\mathbb{N} \times \mathbb{N}$. As a function we can take a restriction of a matrix to a set of indices. In the above, we use X and Y to denote the row and column indexing respectively. For example with the matrix M, given below $M|_{X=0}$ and $M_{Y=0}$ would be as follows:

$$M = \begin{bmatrix} M[0,0] & \dots & M[0,n] \\ \vdots & & \vdots \\ M[m,0] & \dots & M[m,n] \end{bmatrix} \quad M|_{X=0} = \begin{bmatrix} M[0,0] & \dots & M[0,n] \end{bmatrix} \quad M|_{Y=0} = \begin{bmatrix} M[0,0] \\ \vdots \\ M[m,0] \end{bmatrix}$$

But this is more powerful than just simple evaluation. We are selecting not a fixed axis as (x, y) is the input to our function. And so for a matrix $M|_{X=x}$ or $M|_{Y=y}$ we transform $M : X \times Y \to Z$ to the curried $M|_{X=i} : Y \to (X \to Z)$ or $M|_{Y=j} : X \to (Y \to Z)$. Within the context of 3.28, this transforms the blocks of A into horizontal vector slices and B into vertical slices.

Ignoring the differences in transposition, when thought of as functions these functions both map from M (the common axis of A and B) to functions with a common range. And so we have the pointwise sum of terms of the forms $m \mapsto (x \mapsto A[x][m])$ and $m \mapsto (y \mapsto B[m][y])$. The work of multiplying matching $A[x][m]$ with $B[m][y]$ is handled by the $\mathcal{R}_\times$. This leaves us with the product of two functions with different domains, but common range:

$$(x \mapsto A[x][m]) \times (y \mapsto B[m][y]) = (x, y) \mapsto A[x][m] \times B[m][y]$$

Finally, we have the sum over M. If A and B are matrices over a field F then the $\times$-reduction yields a function $M \to (N \times P \to F)$. Summing over the set M leaves us with a function $(N \times P \to F)$ which agrees (at least by object type) with our expectations for C. The familiar structure of summing over a product suggest correctness when $\{M_1, M_2, M_3\}$ is a strict partition of M (that is, when $r \leq s$). Despite the mental hurdles of say a $2 \times (-3)$ matrix, it continues to hold for general partitions as well.

3.5.1 Example: *Matrix Multiplication Concretely*

We will consider the product of two block matrices Q and R. For this example, to better differentiate between blocks, we will change our notation slightly and give each block a distinct

letter names: A, B, C, D for the blocks of Q and E, F, G, H for the blocks of R.

$$Q = \left[\begin{array}{cc|c} a_1 & a_2 & b_1 \\ a_3 & a_4 & b_2 \\ \hline c_1 & c_2 & d_1 \\ c_3 & c_4 & d_2 \end{array}\right] \quad \text{and} \quad R = \left[\begin{array}{c|cccc} e_1 & f_1 & f_2 & f_3 & f_4 \\ \hline g_1 & h_1 & h_2 & h_3 & h_4 \\ g_2 & h_5 & h_6 & h_7 & h_8 \end{array}\right]$$

We will again use M, N and P for the sets of indices. As 4×3 and 3×5 matrices, we have $M = [\![0, 3]\!]$, $N = [\![0, 2]\!]$ and $P = [\![0, 4]\!]$. To align with the blocks of Q and R, each of these sets is partitioned as follow:

$$\begin{array}{lll} N_1 = [\![0, 1]\!] & N_2 = [\![2, 3]\!] & \\ P_1 = [\![0]\!] = \left\{\!\!\left\{ 0^{+1} \right\}\!\!\right\} & P_2 = [\![1, 4]\!] & \\ M_1 = [\![0, 1]\!] & M_2 = (\!(1)\!) = \left\{\!\!\left\{ 1^{-1} \right\}\!\!\right\} & M_3 = [\![1, 2]\!] \end{array}$$

We should note here that our guess was wrong; M_2 is negatively oriented! Although we could have constructed two expressions to handle this case as well, this will not be necessary. We can continue as if nothing is wrong, and the hybrid function structure will take care of cancellations.

We can still write Q and R as:

$$Q = A^{N_1 \times M_1} \oplus B^{N_1 \times (M_2 \oplus M_3)} \oplus C^{N_2 \times M_1} \oplus D^{N_2 \times (M_2 \oplus M_3)}$$
$$R = E^{(M_1 \oplus M_2) \times P_1} \oplus F^{(M_1 \oplus M_2) \times P_2} \oplus G^{M_3 \times P_1} \oplus H^{M_3 \times P_2}$$

The only difference is that originally the sum $(M_2 \oplus M_3) = \{2\}$ was intended to *extend* M_3. When M_2 is negative, it is a set of indices which is *smaller* than the $M_3 = \{1, 2\}$ we started with. Similarly, in the expression for R, $(M_1 \oplus M_2)$ is smaller than M_1. We will use S to denote

the product QR which is still another 2×2 block matrix by the same construction as (3.27):

$$S = Q \cdot R = \begin{bmatrix} S_1 & S_2 \\ S_3 & S_4 \end{bmatrix} = {S_1}^{N_1 \times P_1} \oplus {S_2}^{N_1 \times P_2} \oplus {S_3}^{N_2 \times P_1} \oplus {S_4}^{N_2 \times P_2}$$

Let us compute one of these blocks: S_1.

$$\begin{aligned} {S_1}^{N_1 \times P_1}(i, j) = \sum_{m \in M} \mathcal{R}_\times \Big(& A^{N_1 \times M_1}\big|_{X=i} \oplus E^{M_1 \times P_1}\big|_{Y=j} \oplus \\ & B^{N_1 \times M_2}\big|_{X=i} \oplus E^{M_2 \times P_1}\big|_{Y=j} \oplus \\ & B^{N_1 \times M_3}\big|_{X=i} \oplus G^{M_3 \times P_1}\big|_{Y=j} \Big) \end{aligned}$$

As this is a small example our curried functions only range over $\{0, 1, 2\}$. This is a small enough domain to express each of the functions as a set of point-wise mappings. So let's expand out each of our terms as formal *hybrid sets* (recall a hybrid function is a special hybrid set of ordered pairs):

$$\begin{aligned} \sum_{m \in M?} \mathcal{R}_\times \Big(& \left\{\!\!\left\{ \left(0 \mapsto \left[\begin{smallmatrix} a_1 \\ a_3 \end{smallmatrix}\right]\right)^{+1}, \left(1 \mapsto \left[\begin{smallmatrix} a_2 \\ a_4 \end{smallmatrix}\right]\right)^{+1} \right\}\!\!\right\} && \oplus \left\{\!\!\left\{ (0 \mapsto [e_1])^{+1}, (1 \mapsto [e_\perp])^{+1} \right\}\!\!\right\} \\ & \oplus \left\{\!\!\left\{ \left(1 \mapsto \left[\begin{smallmatrix} b_\perp \\ b_\perp \end{smallmatrix}\right]\right)^{-1} \right\}\!\!\right\} && \oplus \left\{\!\!\left\{ (1 \mapsto [e_\perp])^{-1} \right\}\!\!\right\} \\ & \oplus \left\{\!\!\left\{ \left(1 \mapsto \left[\begin{smallmatrix} b_\perp \\ b_\perp \end{smallmatrix}\right]\right)^{+1}, \left(2 \mapsto \left[\begin{smallmatrix} b_1 \\ b_2 \end{smallmatrix}\right]\right)^{+1} \right\}\!\!\right\} && \oplus \left\{\!\!\left\{ (2 \mapsto [g_1])^{+1}, (2 \mapsto [g_2])^{+1} \right\}\!\!\right\} \Big) \end{aligned}$$

We are using $e_\perp$ and $b_\perp$ here to represent that the functions E and B are undefined for these points. In reality, we would simply not even attempt to evaluate $B|_{X=x}(1)$ or $E|_{Y=y}(1)$ as the functions are undefined. These points are actually contained in the A and G blocks, once again we must delay evaluation with pseudo-functions.

Applying the $\times$-reduction $\mathcal{R}_\times$, we group terms by their input value (e.g. $1 \mapsto x$ with $1 \mapsto y$) and flatten using the multiplicity to repeat or invert the $\times$ operator. In this case, we are dealing only with multiplicities of $+1$ and -1 which correspond with multiplication and "division".

This is not true division, as $0 \times^{-1} 0 = 1$ without fear of division by zero. Otherwise for non-zero operands, $\times^{-1}$ agrees with the normal understanding of division. This is made possible by working with multiplication as a *group* rather than as a *ring* and so we are not worried about making multiplication "play nice" with addition. Doing this yields:

$$\sum_{M_1 \oplus M_2 \oplus M_3} \Big\{\!\!\Big\{ \left(0 \mapsto \left[\begin{smallmatrix} a_1 \\ a_3 \end{smallmatrix}\right] \times^{+1} [e_1]\right), \\ \left(1 \mapsto \left[\begin{smallmatrix} a_2 \\ a_4 \end{smallmatrix}\right] \times^{+1} [e_\perp] \times^{-1} \left[\begin{smallmatrix} b_\perp \\ b_\perp \end{smallmatrix}\right] \times^{-1} [e_\perp] \times^{+1} \left[\begin{smallmatrix} b_\perp \\ b_\perp \end{smallmatrix}\right]\right) [g_1], \\ \left(2 \mapsto \left[\begin{smallmatrix} b_1 \\ b_2 \end{smallmatrix}\right] \times^{+1} [g_2]\right) \Big\}\!\!\Big\}$$

After some cancellations in the second term, we evaluate $\times^{+1}$ as matrix multiplication and sum over all of M:

$$S_1{}^{N_1 \times P_1} = \begin{bmatrix} a_1 \\ a_3 \end{bmatrix} [e_1] + \begin{bmatrix} a_2 \\ a_4 \end{bmatrix} [g_1] + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} [g_2] = \begin{bmatrix} a_1 e_1 + a_2 g_1 + b_2 g_2 \\ a_3 e_1 + a_4 b_1 + b_2 g_2 \end{bmatrix}$$

As expected we have a $|N_1| \times |P_1| = (2 \times 1)$ matrix which will form the upper left block of S. Ignoring the block structure of Q and R and performing normal matrix multiplication, we also find that these values agree with $S[0,0]$ and $S[1,0]$. Computations for the blocks S_2, S_3 and S_4 are performed identically yielding blocks of varying sizes. Together, theses blocks form a strict partitioning of S as a 2×2 block matrix.

3.5.2 Multiplication with Larger Block Matrices

Extending to larger block matrices is a fairly trivial affair. Once again we will use $\{N_i\}$ to divide the rows of blocks in A and $\{P_j\}$ to divide the block columns of B. M_k and M'_k are two different partitions of the common axis for A and B respectively.

$$A = \left[\begin{array}{c|c|c} A_{1,1}^{N_1\times M_1} & \dots & A_{1,K}^{N_1\times M_K} \\ \hline \vdots & & \vdots \\ \hline A_{I,1}^{N_I\times M_1} & \dots & A_{I,K}^{N_I\times M_K} \end{array}\right] \qquad B = \left[\begin{array}{c|c|c} B_{1,1}^{M_1\times P_1} & \dots & B_{1,J}^{M_1\times P_J} \\ \hline \vdots & & \vdots \\ \hline B_{K',1}^{M_{K'}\times P_1} & \dots & B_{K',J}^{M_{K'}\times P_J} \end{array}\right]$$

$$= \bigoplus_{i\in[\![1,I]\!]} \bigoplus_{k\in[\![1,K]\!]} A_{i,k}^{N_i\times M_k} \qquad = \bigoplus_{k'\in[\![1,K']\!]} \bigoplus_{j\in[\![1,J]\!]} B_{k',j}^{M'_{k'}\times P_j}$$

As before the blocks of C will be of sizes $N_i \times P_j$:

$$C = \bigoplus_{i\in[\![1,I]\!]} \bigoplus_{j\in[\![1,J]\!]} \left(C_{i,j}^{N_i\times P_j}\right) = \left[\begin{array}{c|c|c} C_{1,1}^{N_1\times P_1} & \dots & C_{1,J}^{N_1\times P_J} \\ \hline \vdots & & \vdots \\ \hline C_{I,1}^{N_I\times P_1} & \dots & C_{I,J}^{N_I\times P_J} \end{array}\right] \tag{3.29}$$

and where each $C_{i,j}$ is defined as:

$$C_{i,j} = \sum_M \mathcal{R}_\times \left(\bigoplus_{k\in[\![1,K]\!]} A_{i,k}^{N_i\times M_k}\Big|_{X=x} \oplus \bigoplus_{k\in[\![1,K']\!]} B_{k',j}^{M'_k\times P_j}\Big|_{Y=y} \right) \tag{3.30}$$

Chapter 4

Integration over Hybrid Domains

In many ways, integration provides the inspiration for oriented intervals. As such, many of the techniques we have been using will be very familiar when placed back within their original context. That being said, notationally, sets and orientation are treated as often treated as distinct objects rather than a single entity. Over this chapter we will argue for a "refactoring" to bring orientation and support back together.

As a hopefully illustrative example of this, consider a typical definition of the **definite integral** from an introductory course in calculus. Given a function f with real variable x and an interval $[a, b)$ of the (extended) real line [1] the definite integral

$$\int_a^b f(x)\,\mathrm{d}x$$

is defined as the signed area bounded by f between $x = a$ and $x = b$. Generally after a short exposition about Riemann sums, it would then be revealed that if F is an anti-derivative of the function f then:

$$\int_a^b f(x)\,\mathrm{d}x \;=\; F(b) - F(a) \;=\; -(F(a) - F(b)) \;=\; -\int_b^a f(x)\,\mathrm{d}x$$

[1] The extended real line denoted $\bar{\mathbb{R}}$ is the set of real numbers as well as the points at $+\infty$ and $-\infty$

However this means that previously defining the definite integral using (unoriented) intervals was a bit of a misnomer. As we saw in the previous chapter, when $a \geq b$, the interval $[a, b) = \{x \mid a \leq x < b\}$ is the empty set. So the interval itself cannot really be thought of as a part of the definite integral.

If it were the interval itself that we were concerned with then for $a \leq b$, the interval $[b, a)$ is empty, as are the intervals $[a, a)$, $[\pi, e)$, and $[\infty, -\infty)$. As these are all different representations of the same interval, and if we were truly concerned with the relationship between f and the interval, then one might argue that:

$$\int_b^a f(x)\,\mathrm{d}x \;=\; \int_a^a f(x)\,\mathrm{d}x \;=\; \int_\pi^e f(x)\,\mathrm{d}x \;=\; \int_\infty^{-\infty} f(x)\,\mathrm{d}x$$

Obviously this is not the intent but there is a distinct mismatch between the conceptual usefulness of considering integrals as over an interval. But when actually using said interval, it is treated as an ordered pair of endpoints disregarding the set itself.

This issue is exasperated working with the more general notation

$$\int_X f(x)\,\mathrm{d}x$$

which denotes integrating f over a set X. Now there is nothing stopping X from an interval and one would very much like to say that we can convert between the two notations with a definition like:

$$\int_a^b f(x)\,\mathrm{d}x \;=\; \int_{[a,b)} f(x)\,\mathrm{d}x$$

but there is no analogous translation to $\int_a^b = -\int_b^a$ and so if we made this assertion we would be left with

$$\int_{[a,b)} f(x)\,\mathrm{d}x = \int_a^b f(x)\,\mathrm{d}x = -\int_b^a f(x)\,\mathrm{d}x = -\int_{[b,a)} f(x)\,\mathrm{d}x = -\int_\emptyset f(x)\,\mathrm{d}x = 0$$

This is clearly not a desired outcome; instead what is actually intended is an oriented interval.

What we *intend* is an integral over the the hybrid set $[\![a, b)\!) = \ominus[\![b, a)\!)$ not the set $[a, b)$.

Another advantage to using oriented sets is a more natural language for manipulating domains of integration than sets. We cannot add sets (in the traditional, non-σ-algebra sense) but with the point-wise sum $\oplus$, we *can* add hybrid sets. This allows us to say that the integral operator is *bi-linear*. By this we mean, that integration is a function of two operands, the integrand and the domain. Integration is linear over integrands regardless,

$$\int_X f(x) + g(x)\,\mathrm{d}x = \int_X f(x)\,\mathrm{d}x + \int_X g(x)\,\mathrm{d}x$$

but with summation defined on the domain as well

$$\int_{X\oplus Y} f(x)\,\mathrm{d}x = \int_X f(x)\,\mathrm{d}x + \int_Y f(x)\,\mathrm{d}x$$

In one dimension, many of these changes may seem trivial advances but in higher dimensions, the oriented and measure-theoretic approaches diverge [21]. A Riemann integration foundation is not overly concerned with negatively oriented intervals and the definitions continue to function unperturbed by the fact that the "right-hand" bound is actually less than the "left-hand" bound.

As such extending to higher dimension Riemann integrals, orientation is easily bundled in as well. But measure-based approaches to integration need to fake orientation in one dimension. First one must convert $\int_a^b$ into either $-\int_{[a,b)}$ or $\int_{[a,b)}$ and then integrate one or the other. Once one commits to an orientation, there is no coming back. In higher dimensions, this becomes more of a problem.

This is usually forgivable given the extra power the Lebesgue integral affords over the Riemann. Using hybrid sets as domains of integration allow us to use the best features of both. In this chapter we will investigate integration using hybrid set domains.

4.1 The Riemann Integral on k-rectangles

Definition Let $[\![\boldsymbol{a}, \boldsymbol{b}]\!]$ be a k-rectangle in $\mathbb{R}^n$ where $\boldsymbol{a} = (a_1, \ldots, a_n)$ and $\boldsymbol{b} = (b_1, \ldots, b_n)$. We denote the **volume of $[\![\boldsymbol{a}, \boldsymbol{b}]\!]$** with vol and define it as:

$$\text{vol}(\ [\![\boldsymbol{a}, \boldsymbol{b}]\!]\ ) = (b_1 - a_1) \cdot (b_2 - a_2) \cdot \ldots \cdot (b_n - a_n) \tag{4.1}$$

We say volume, but this depends on the dimension. In 2-dimensions, this would better be described as area and in 1-dimension as a length. For any $k < n$, a k-rectangle will have volume zero. This is fitting, as we wished for example to measure the area of rectangle in 2 dimensions, but the same 2-rectangle in $\mathbb{R}^3$ is flat and can hold no volume. In at least one dimension, the cube will be degenerate (i.e. $a_i = b_i$) and so will contribute zero to the product. Additionally, one can also observe that $\text{vol}(\ominus[\![\boldsymbol{a}, \boldsymbol{b}]\!]) = -\text{vol}([\![\boldsymbol{a}, \boldsymbol{b}]\!])$.

The next task is to partiton the k-rectangle $[\![\boldsymbol{a}, \boldsymbol{b}]\!]$ into a grid of smaller k-rectangles. To do this, for each dimension $[\![a_i, b_i]\!]$, we choose a generalized partition P_i such that P_i is composed of oriented intervals. To build our mesh, we construct smaller k-rectangles $I_{i_1,\ldots,i_n}$ using the Cartesian product of pieces:

$$I_{p_1,\ldots,p_n} = p_1 \times \ldots \times p_n$$

where each p_i is taken from P_i. We are now ready to construct Riemann sums.

Definition Given $P = \{P_j\}_{j=1}^n$ where P_j is an interval generalized partition of $[\![a_j, b_j]\!]$, and $f : [\![\boldsymbol{a}, \boldsymbol{b}]\!] \to \mathcal{R}$ then we define *a* Riemann sum $\text{Rie}(f, P, *)$ to be:

$$\text{Rie}(f, P, *) = \sum_{p_1 \in P_1} \ldots \sum_{p_n \in P_n} f(x^*_{p_1,\ldots,p_n}) \text{vol}(I_{p_1,\ldots,p_n}) \tag{4.2}$$

where $x^*_{p_1,\ldots,p_n}$ is a point from $I_{p_1,\ldots,p_n}$ as chosen by some selector $*$.

Note that we specify *a* Riemann sum, not *the* Riemann sum. There are several ways to choose $x^*_{i_1,\ldots,i_n} \in I_{i_1,\ldots,i_n}$ and different samplings can lead to different Riemann sums for the same

partition and same function. In $\mathbb{R}^1$, several common ways to sample include the left and right Riemann sums (i.e. $\mathrm{Rie}(f, P, \min(x))$ and $\mathrm{Rie}(f, P, \max(x))$), the trapezoidal Riemann sum (i.e. $\mathrm{Rie}(f, P, \min(x)+\max(x)/2)$), and the upper and lower Riemann sums (i.e. $\mathrm{Rie}(f, P, \min(f(x)))$ and $\mathrm{Rie}(f, P, \max(f(x)))$).

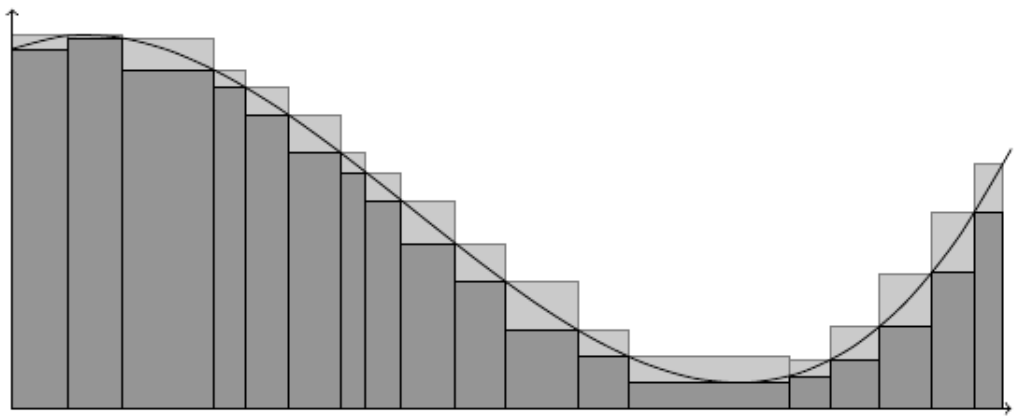


Figure 4.1: A Riemann sum corresponds with the area of a sequence of rectangles. Here, the upper and lower Riemann sums for the same partition are shown with light and dark rectangles respectively. A function over an oriented interval is Riemann integrable if the two sums converge.

Definition The Riemann integral of a function $f : \mathbb{R}^n \to \mathbb{R}$ over a k-rectangle $[\![\boldsymbol{a}, \boldsymbol{b}]\!]$ as $\max(|\mathrm{vol}(I_{p_1,\ldots,p_n})| : p_i \in P_i)$ goes to zero.

$$\mathrm{Rie}\Big(f, P, \min\left(f(x)\right)\Big) \;\leq\; \int_{[\![a,b]\!]} f(x)\,\mathrm{d}x \;\leq\; \mathrm{Rie}\Big(f, P, \max\left(f(x)\right)\Big) \tag{4.3}$$

If these bounds (upper and lower Riemann sums) converge, we say that a function is **Riemann integrable** and define the Riemann integral as the limit.

4.2 The Lebesgue Integral on Hybrid Domains

Another common approach to integration is the Lebesgue integral which approaches integration from a measure theoretic perspective. We begin our construction with a σ-algebra (read sigma-algebra) over a set X. By this we mean a collection of subsets of X which are closed

under countable complement, union and intersection. The important part here is that we have a closed universe of sets on which we can perform *nearly* arbitrary set operations but remain within said universe. Next we can attach to a σ-algebra a measure to form a **measure space** (X, Σ, μ) where X is a set Σ is a σ-algebra over subsets of X and μ is a measure defined on the sets in Σ. This measure μ is a function $\mu : \Sigma \to \bar{\mathbb{R}}$ with the following properties:

Non-negative: For all $E \in \Sigma$, $\mu(E) \geq 0$

Empty set has measure 0: $\mu(\emptyset) = 0$

Countably Additive: For $\{E_i\}_i$, a countable set of disjoint sets in Σ, $\mu\left(\bigcup_i E_i\right) = \sum_i \mu(E_i)$

Within the context of integration, the Borel and Lebesgue measure spaces are notable constructions. Both of which give a suitably large universe of sets for which to integrate over. Certainly much more than the set of Riemann integrable domains. Finally, given a measure space (X, Σ, μ), we say that $f : X \to \mathbb{R}$ is a **measurable function** if $\{x \mid f(x) > t\}$ is a measurable set for all t.

If 1_S is indicator function $1_S : X \to \{0, 1\}$ given by $1_S(x) = 1$ if $x \in S$ and $1_S(x) = 0$ otherwise. Clearly if S is a measurable set, then 1_S is a measurable function. We will use this as a base case. Given a measure space (X, Σ, μ) and $S \in \Sigma$, we define the integral:

$$\int 1_S \, \mathrm{d}\mu = \mu(S) \tag{4.4}$$

From this, we consider functions which are the sum of indicator functions. We say that s is a **simple function** if there are finite sets of measurable sets and $\{A_k\}_{k=0}^n$ and matching real coefficients $\{a_k\}_{k=0}^n$ such that:

$$s = \sum_{k=0}^{n} a_k 1_{A_k}$$

The integral of a simple function is then easily defined linearly in terms of integrals of indicator functions:

$$\int s \, \mathrm{d}\mu = \int \left(\sum_{k=0}^{n} a_k 1_{A_k} \right) \mathrm{d}\mu = \sum_{k=0}^{n} a_k \int 1_{A_k} \, \mathrm{d}\mu = \sum_{k=0}^{n} a_k \cdot \mu(A_k) \tag{4.5}$$

But we don't always wish to integrate over the entire measure space but rather some measurable subset of X. This would be done with the notation $\int_B$ instead of $\int$ and replacing $\mu(A_k)$ with $\mu(A_k \cap B)$. If A_k and B are both measurable sets in Σ then their intersection is also a measurable set in Σ. But as already mentioned, integrating over sets is a misnomer; we should be integrating over oriented sets. To do this, we will need to extend μ.

The typical approach would be to construct a **signed measure** over Σ. As a signed measure we lift the non-negative condition and allow for negative values or *charge*. In one dimension, this is analogous to considering $q-p$ (signed measure) as opposed to Euclidean distance: $\|q-p\|$ (unsigned measure). But this does not allow us to integrate sets, rather it allows us to integrate sets *with orientation*. Consider $\int_{[0,1]} \mathrm{d}\mu$, with no extra information regarding the set $[0, 1]$. This integral is either 1 or -1 we must supply additional information in order to distinguish between $\int_0^1 \mathrm{d}\mu$ and $\int_1^0 \mathrm{d}\mu$.

Instead, both orientation and set are contained within our measurable hybrid sets so we will use this instead. Rather than a signed measure over the σ-algebra Σ itself, we will instead use a signed measure over $\mathbb{Z}^\Sigma$, the space of hybrid sets over Σ. Assuming an existing measure μ on Σ, this construction comes for free by extending linearly. Thus, to integrate over a hybrid set $H \in \mathbb{Z}^\Sigma$:

$$\int_H s \,\mathrm{d}\mu = \int H \cdot s \,\mathrm{d}\mu = \sum_k a_k \cdot \mu(A_k \otimes H) \tag{4.6}$$

We then use simple functions to approximate general measurable functions. For a non-negative function f, we say this is the largest simple function that is everywhere less than f:

$$\int_H f \,\mathrm{d}\mu = \sup \left\{ \int_H s \,\mathrm{d}\mu \;\middle|\; s \text{ simple, and } 0 \le \varphi \le f \right\} \tag{4.7}$$

But even if f takes negative values, we can split it into positive and negative parts by:

$$f^+(x) := \max(0, f(x))$$

$$f^-(x) := \max(0, -f(x))$$

Both f^+ and f^- are clearly non-negative but also, observe that $f = f^+ - f^-$. Linearly, this allows us to define for any measurable f:

$$\int_H f \, \mathrm{d}\mu = \int_H f^+ \, \mathrm{d}\mu - \int_H f^- \, \mathrm{d}\mu \tag{4.8}$$

The last issue that remains to be settled is whether such a limit of simple functions even exists. For this we can use the sequence of simple functions ψ_n defined by:

$$\psi_n = \sum_{k=0}^{n2^n-1} \left[\left(\frac{k}{2^n} \right)^{\left[\frac{k}{2^n}, \frac{k+1}{2^n}\right)} \right] + n^{[n,\infty]} \tag{4.9}$$

Notationally, this definition is rather heavy but is easily understood geometrically as seen below in Figure 4.2. In turn this allows us to define for any non-negative f the sequence of functions:

$$\varphi_n = \psi_n \circ f \tag{4.10}$$

We know that φ_n is simple since ψ_n is simple. And since $\psi_n(x) \leq x$ for all x then we also have $0 \leq \varphi_n \leq f$. But, most importantly we have $0 \leq f(x) - \varphi_n(x) \leq 2^{-n}$ and so φ_n, uniformly approaches f as n approaches infinity.

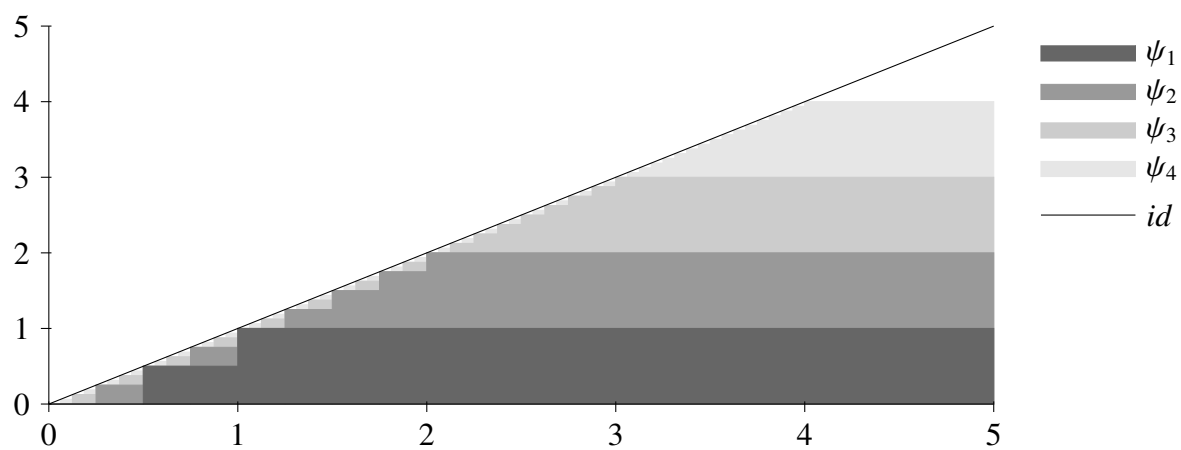


Figure 4.2: The simple functions ψ_n for n from 1 to 4. As n goes to infinity, ψ_n comes arbitrarily close to x over the range 0 to n.

4.2.1 Example: *Integrating the Irrationals*

A Lebesgue measurable hybrid set can now have an orientation like Riemann integral intervals and the power of Lebesgue integration. Consider the indicator function for the set of rational numbers, $1_{\mathbb{Q}}$ which evaluates to 1 for rational numbers and 0 for irrational numbers. Over the interval $[0, 1]$ on the real line, this is not Riemann-integrable.

Suppose we have some generalized partition of $[0, 1]$ made up of intervals. Then for any of these sub-intervals, there will be at least one rational and one irrational number. Hence the upper Riemann sum will be 1 and the lower Riemann sum will be 0. No matter how small we make our intervals, there will always be a rational and irrational number in each. The two sums will never converge and so there is no well-defined Riemann integral.

On the other hand, it is Lebesgue integrable:

$$\int_{[0,1]} 1_{\mathbb{Q}} \, \mathrm{d}\mu \;=\; \mu\Big(\mathbb{Q} \cap [0, 1]\Big)$$

But since $\mathbb{Q}$ is countable, it has measure 0 and so $1_{\mathbb{Q}}$ *is* Lebesgue integrable on $[0, 1]$. Similarly, we could also find that:

$$\int_0^1 1_{\mathbb{R}\setminus\mathbb{Q}} \, \mathrm{d}\mu \;=\; \int_{[0,1]} 1_{\mathbb{R}\setminus\mathbb{Q}} \, \mathrm{d}\mu \;=\; \mu\Big((\mathbb{R} \setminus \mathbb{Q}) \cap [0, 1]\Big) \;=\; \mu\Big((\mathbb{R} \setminus \mathbb{Q}) \cap [0, 1]\Big) \;=\; 1$$

One would expect the same process to hold for when integrating from 0 to 1 but restricted to sets, this is not possible. The statement $[1, 0] = -[0, 1]$ is nonsense. But if we instead take integrating from a to b to mean the Lebesgue integral over the oriented $[\![a, b]\!]$, we can get a result of -1 as we would expect:

$$\int_1^0 1_{\mathbb{R}\setminus\mathbb{Q}} \, \mathrm{d}\mu \;=\; \int_{[\![1,0]\!]} 1_{\mathbb{R}\setminus\mathbb{Q}} \, \mathrm{d}\mu \;=\; \mu((\mathbb{R} \setminus \mathbb{Q}) \otimes [\![1, 0]\!]) \;=\; -\mu((\mathbb{R} \setminus \mathbb{Q}) \otimes [\![0, 1]\!]) \;=\; -1$$

4.3 Differential Forms

Rather than thinking of integrals as functions over n-rectangles, an often more useful language is to use *differential forms*. We define a **(differential) 0-form** β on $\mathbb{R}^n$ as any function $\beta : \mathbb{R}^n \to \mathbb{R}$. And there is very little else to say as they are just functions on $\mathbb{R}^n$.

A **(differental) 1-form** ω on $\mathbb{R}^n$ is an expression of the form:

$$\omega = f_1(\mathrm{x})\,\mathrm{d}x_1 + f_2(\mathrm{x})\,\mathrm{d}x_2 + \ldots + f_n(\mathrm{x})\,\mathrm{d}x_n$$

Now this looks very much like something we're used to integrating. Specifically it is something that be used as an integrand over a 1 dimensional domain. For example, Green's theorem is often introduced using differential forms without even mentioning them as such:

$$\iint_D \left(\frac{\partial f_2}{\partial x} - \frac{\partial f_1}{\partial y}\right)\mathrm{d}x\,\mathrm{d}y = \int_C \Big(f_1(x,y)\,\mathrm{d}x \;+\; f_2(x,y)\,\mathrm{d}y\Big) \qquad \text{(Green's Theorem)}$$

Here C is a closed curve that encloses D a region in the (x, y) plane; hence the right-hand side is an integral of a 1-form over a 1-dimensional curve. Having multiple $\mathrm{d}x_i$ appearing in a single integrand may initially seem unusual when first presented, but is quite intuitively handled. Integration is linear so just as we can separate $\int \Big(f(x) + g(x)\Big)\,\mathrm{d}x$ into $\int f(x)\,\mathrm{d}x + \int g(x)\,\mathrm{d}x$, we can similarly breakup an integral of a 1-form into the sum of integrals over *basic* 1-forms (i.e. 1-forms involving only a single term):

$$\int \Big(f_1(\mathrm{x})\,\mathrm{d}x_1 + f_2(\mathrm{x})\,\mathrm{d}x_2 + \ldots + f_n(\mathrm{x})\,\mathrm{d}x_n\Big) \;=\; \sum_{i=1}^{n}\left(\int f_i\,\mathrm{d}x_i\right)$$

Adding two 1-forms then is quite straight-forward; simply collect terms with matching $\mathrm{d}x_i$. So if we can add differential forms but what about multiplication? For a 0-form β and 1-form ω as defined above, the answer the answer is a simple yes:

$$(\beta \cdot \omega)(\mathrm{x}) = \Big(\beta(\mathrm{x})f_1(\mathrm{x})\Big)\,\mathrm{d}x_1 + \ldots + \Big(\beta(\mathrm{x})f_n(\mathrm{x})\Big)\,\mathrm{d}x_n$$

The result is a 1-form where each basic 1-form term is the product of f_i and β in $\mathbb{R}$. To "multiply" two 1-forms together however we must turn instead to the wedge product $\wedge$.

First of all, the wedge product is primarily defined by being *anti-commutative* or *skew-symmetric*. That is, $\mathrm{d}x \wedge \mathrm{d}y = -\mathrm{d}y \wedge \mathrm{d}x$ and several results will immediately follow. When applied to two identical $\mathrm{d}x$, we have $\mathrm{d}x \wedge \mathrm{d}x = -\mathrm{d}x \wedge \mathrm{d}x$ and so $\mathrm{d}x \wedge \mathrm{d}x = 0$. Additionally, for any permutation σ of $[p]$:

$$\mathrm{d}x_1 \wedge ... \wedge \mathrm{d}x_p = \mathrm{sgn}(\sigma)\,\mathrm{d}x_{\sigma(1)} \wedge ... \wedge \mathrm{d}x_{\sigma(p)}$$

The wedge product of two 1-forms moves us out of the realm of 1-forms which have basis $\mathrm{d}x_i$ and into the realm of 2-forms with basis $\mathrm{d}x_i \wedge \mathrm{d}x_j$.

Definition Given a k-rectangle $\Omega \in \mathbb{R}^n$ with coordinates $\mathrm{x} = (x_1, x_2, \ldots, x_n)$ A **differential p-form** β over Ω has the form:

$$\beta = \sum_{j_1 \in [n]} \ldots \sum_{j_p \in [n]} \left(b_{(j_1,\ldots,j_p)}(\mathrm{x})\,\mathrm{d}x_{j_1} \wedge \ldots \wedge \mathrm{d}x_{j_p} \right) \tag{4.11}$$

Typically, we will take j to be the vector $(j_1, \ldots, j_p)$ and express β instead as a single sum multi-indexed by j. We denote the **space of all p-forms on Ω** as $\Lambda^p(\Omega)$.

Definition Let $\alpha = \sum_i a_i(x)\,\mathrm{d}x_{i_1} \wedge \ldots \wedge \mathrm{d}x_{i_p} \in \Lambda^p(\Omega)$ and $\beta = \sum_j b_j(x)\,\mathrm{d}x_{j_1} \wedge \ldots \wedge \mathrm{d}x_{j_q} \in \Lambda^q(\Omega)$. We extend the wedge product to $\wedge : \Lambda^p(\Omega) \times \Lambda^q(\Omega) \to \Lambda^{p+q}(\Omega)$ by:

$$\alpha \wedge \beta = \sum_{i,j} \left(a_i(x) b_j(x)\ \mathrm{d}x_{i_1} \wedge \ldots \wedge \mathrm{d}x_{i_p} \wedge \mathrm{d}x_{j_1} \wedge \ldots \wedge \mathrm{d}x_{j_q} \right) \tag{4.12}$$

Although we take all possible $\binom{n}{q} \cdot \binom{n}{p}$ pairs of $a_i(x)\,\mathrm{d}x_{i_1} \wedge \ldots \wedge \mathrm{d}x_{i_p}$ and $b_i(x)\,\mathrm{d}x_{i_1} \wedge \ldots \wedge \mathrm{d}x_{i_p}$, most of the possible terms will end up being zero. If *any* of the terms in $\mathrm{d}x_i$ appears in $\mathrm{d}x_j$, then the wedge product will be zero and no term will be contributed. As such, if $q + p > n$, there will be a duplicate in every term and so the entire sum will be zero. When all is said

and done, at most we will have $\binom{n}{p+q}$ terms. Rather than the skew-symmetry we had when commuting $\mathrm{d}x \wedge \mathrm{d}y$, in higher dimensions the sign depends on $p \cdot q$ of the p-form and q-form we are commuting. Specifically,

$$\alpha \wedge \beta = (-1)^{pq} \beta \wedge \alpha \tag{4.13}$$

This can be easily seen by commuting each of $\mathrm{d}x_{j_1}, \ldots \mathrm{d}x_{j_q}$ terms each past $\mathrm{d}x_{i_1} \wedge \ldots \wedge \mathrm{d}x_{i_p}$. So we are commuting q terms each past p terms, reversing the sign each time for a net $(-1)^{pq}$. This result generalizes the earlier skew-symmetry of transposing $\mathrm{d}x \wedge \mathrm{d}y$. When α and β are both 1-forms then clearly $-1^{pq} = -1^{1 \cdot 1} = -1$.

The wedge product is only one part of our algebra of differential forms. We have several other nice identities for its behaviour with addition and multiplication. For the following, we consider f to be a function on $\mathbb{R}^n$. Additionally we consider the differential forms ω_1 and ω_2 to be k-forms, α to be a p-form and β to be an q-form. Then we have the following:

$$(\omega_1 + \omega_2) \wedge \alpha = \omega_1 \wedge \alpha + \omega_2 \wedge \alpha \tag{4.14}$$

$$(\omega_1 \wedge \alpha) \wedge \beta = \omega_1 \wedge (\alpha \wedge \beta) \tag{4.15}$$

$$(f \cdot \omega_1) \wedge \alpha = f \cdot (\omega_1 \wedge \alpha) = \omega_1 \wedge (f \cdot \alpha) \tag{4.16}$$

These should all be quite obvious from definitions. We should also note the identities which are *not* present. We have defined the sum of ω_1 and ω_2: two differential forms which are the same dimension but not the sum of α and β: differential forms with different dimension. It is clear how one would add two differential forms of the same dimension as both were defined as sums to begin with. We also do not define the multiplication of $\cdot$ two differential forms but we multiplying a form by a function is simply:

$$(f \cdot \alpha)(x) = \sum_i f(x) \cdot a_i(x) \, \mathrm{d}x_{i_1} \wedge \ldots \wedge \mathrm{d}x_{i_p}$$

Integrating over a k-form is quite simple. First, consider integrating a k-form over a k-rectangle in $\mathbb{R}^k$. Such a k-form is also known as a *top-dimensional form*. As we saw previously, any form of higher degree must be zero. If ω is such a top form then we can always write

$$\omega = f \ \mathrm{d}x_1 \wedge \ldots \wedge \mathrm{d}x_k$$

for some function f. Other presentations of ω exist, but we can always achieve such a presentation by commuting over $\wedge$ to the canonical ordering $x_1, \ldots, x_n$. Once a k-form in this presentation, remove the wedges and evaluate the integral using the integrand $f \, \mathrm{d}x_1 \, \mathrm{d}x_2 \ldots \mathrm{d}x_k$.

Definition Let α be a k-form on $\Omega \subset \mathbb{R}^n$ of the form $\alpha = A(x) \, \mathrm{d}x_1 \wedge ... \wedge \mathrm{d}x_n$. If $A \in \mathcal{L}^1(\Omega, \mathrm{d}x)$ then we define:

$$\int_\Omega \alpha = \int_\Omega A(x) \ \mathrm{d}x \tag{4.17}$$

Where the left-hand side is the integral of a k-form and the right-hand side is a Lebesgue integral. For any $\beta \in \Lambda^k(\Omega)$ we extend this definition linearly as the sum of integrals.

Finally, we extend the differential operator d to act on forms known as the **exterior derivative**. For a function (0-form), it is the 1-form:

$$\mathrm{d}f = \sum_i \frac{\partial f}{\partial x_i} \mathrm{d}x_i \tag{4.18}$$

This will result in equivalent 1-forms regardless of the choice of coordinates $\mathrm{x} = (x_1, \ldots, x_n)$. For higher dimension forms it will similarly map a k-form to a $k+1$-form. This is done by recursively using the identities for p-form α and q-form β.

$$\mathrm{d}(\alpha \wedge \beta) = (\mathrm{d}\alpha) \wedge \beta + (-1)^p \alpha \wedge (\mathrm{d}\beta) \tag{4.19}$$

$$\mathrm{d}(\ \mathrm{d}(\alpha)\ ) = 0 \tag{4.20}$$

and extending linearly for all k-forms.

4.4 Singular Cubes

Up until now we have been dealing with the very small set of axis aligned k-rectangles which is a very limiting class to be restricted to. Instead we would like to be able to integrate over k-rectangles that are deformed by some smooth function. So assume that we have $X \subset \mathbb{R}^m$ and $Y \subset \mathbb{R}^n$ and a smooth map $\Phi : X \to Y$. Not only can we map points from X to points in Y but we can *push foward* vectors from X to vectors in Y and with them, push forward tangent spaces as well.

Definition We denote the **standard k-cube** as the specific k-rectangle $[\![0,1]\!]^k$ in $\mathbb{R}^n$ which is the Cartesian product of k copies of $[\![0,1]\!]$. We also consider $[\![0,1]\!]^0 = \{0\}$. Given an k-dimensional manifold M, a **singular k-cube in M** is a smooth differentiable map c from the standard k-cube to M, $c : [\![0,1]\!]^k \to M$. We will abuse this notation somewhat by also using $c \subseteq M$ to refer to the image of $[\![0,1]\!]^k$ under c.

For example, the hemisphere $H = \{(x,y) \mid x \geq 0, y \geq 0, x^2 + y^2 \leq 1\}$ is a singular 2-cube under the transformation $c : (r,\varphi) \mapsto r\cos(\pi\varphi)x + r\sin(\pi\varphi)y$. Depending on the context we might refer to either H or c as a singular cube. Also, the choice of using specifically the standard k-cube is arbitrary. A differentiable map f from $[\![a,b]\!]$ can always be composed with $g : t \mapsto ta + (1-t)b$ to construct the singular cube $c = f \circ g$.

However if we have a valid integral $\int_\Omega \omega$ with Ω in some space X, if we push forward Ω by some function c to another space Y, then the integral $\int_{c(\Omega)} \omega$ is no longer valid. The differential form ω was expressed in coordinates for X but now that the domain of the integral is in Y, we must perform a change of coordinates. The true reason why we use differential forms is how cleanly they handle this change in coordinates through the use of pull-backs.

Informally, a pullback is a *reversed* function composition. In typical function composition $(f \circ g)(x) = f(g(x))$, for input x one first evaluates the second function g at x before feeding the result of $g(x)$ into the first function f. The pre-composition or pullback would be $f^*g = g(f(x))$. One first evaluates the first function and feeds the result into the second. Gets its name from

pulling f back through g. Using the following identities:

$$F^*(\alpha \wedge \beta) = (F^*\alpha) \wedge (F^*\beta) \tag{4.21}$$

$$F^*(\mathrm{d}\beta) = \mathrm{d}F^*\beta \tag{4.22}$$

one finds a very convenient way to express change of basis inside an integral.

Theorem 4.4.1 *Let $F : X \to \Omega$ be an (orientation-preserving diffeomorphism) and α an integrable n-form on Ω then*

$$\int_{F(X)} \alpha = \int_X F^*\alpha \tag{4.23}$$

To integrate over a manifold M, we first observe that each local chart U_i in the manifold is essentially a singular cube. If the chart is not a map over the standard cube, then there exists a diffeomorphism that can be composed with the local map to transform it into a singular cube. It is then a matter of stitching together these local charts so that points in the manifold are not "double-counted". To do this we use a **partition of unity** on M. That is, a collection of functions $\{\psi_i\}_i$, $\psi_i : U_i \to \mathbb{R}$ which is:

Non-negative: For each ψ_i and for all $x \in \mathrm{supp}(\psi_i)$, $\psi_i \geq 0$.

Sums to one(unity): For all $x \in M$, $\sum_i \psi_i(x) = 1$

Locally finite: for any point in M there are only a finite number of non-zero ψ_i

Given such a parititon of unity for the manifold, we define the integral over all of M as:

$$\int_M \alpha = \sum_i \int_{U_i} \psi_i \alpha \tag{4.24}$$

Chapter 5

Stokes' Theorem

When discussing differential forms an equation called *Green's Theorem* was shown. Green's Theorem allows for one to convert between an integral over a 2-dimensional region and a 1-dimensional integral over a curve that bounds it. This turns out to be just one instance of the more general *Stokes' Theorem* which will work in higher dimensions as well. To do so we will first need to generalize the notion of a bounding region or boundary.

5.1 Boundary Operator

In one dimension, the boundary of an interval was quite straight-forward. For a positively oriented interval, the boundary was composed of two points; the right end-point was positive and the left end-point was negative. From the perspective of k-rectangles, the ∂ operator has mapped an oriented 1-rectangle to a set of oriented 0-rectangles. We will now generalize the boundary to map an oriented n-rectangle to an $(n-1)$-rectangle.

Definition Let $[\![\boldsymbol{a}, \boldsymbol{b}]\!]$ be a a k-rectangle in $\mathbb{R}^n$. Additionally, let $i_1, i_2, \ldots, i_k$ be the unique non-decreasing sequence of indices such that $a_{i_j} \neq b_{i_j}$. The **boundary of $[\![\boldsymbol{a}, \boldsymbol{b}]\!]$** , denoted by the

operator ∂ is given by:

$$\partial\left([\![\boldsymbol{a},\boldsymbol{b}]\!]\right) = \bigoplus_{j=1}^{k}(-1)^j \left(\left[\!\!\left[\left(\boldsymbol{a}^{[\![1,n]\!]_{\mathbb{N}}}\right), \quad \left(\boldsymbol{b}^{[\![1,i_j)\!)_{\mathbb{N}}} \oplus \boldsymbol{a}^{\{\!|\, i_j \,|\!\}} \oplus \boldsymbol{b}^{(\!(i_j,n]\!]_{\mathbb{N}}}\right) \right]\!\!\right]_{\mathbb{R}^n} \right.$$
$$\left. \ominus \left[\!\!\left[\left(\boldsymbol{a}^{[\![1,i_j)\!)_{\mathbb{N}}} \oplus \boldsymbol{b}^{\{\!|\, i_j \,|\!\}} \oplus \boldsymbol{a}^{(\!(i_j,n]\!]_{\mathbb{N}}}\right), \quad \left(\boldsymbol{b}^{[\![1,n]\!]_{\mathbb{N}}}\right) \right]\!\!\right]_{\mathbb{R}^n} \right) \tag{5.1}$$

The above equation will require a bit of unpacking to digest featuring oriented intervals in two different contexts. The first appears in the superscripts of $\boldsymbol{a}$ and $\boldsymbol{b}$. The intervals $[\![1, i_j)\!)_{\mathbb{N}}$ and $(\!(i_j, n]\!]_{\mathbb{N}}$ are intervals over vector indices just as in Chapter 3. Thus, the term $\boldsymbol{a}^{[\![1,i_j)\!)_{\mathbb{N}}}$ refers to the vector $(a_1, a_2, \ldots, a_{i_j-1})$ while the term $\boldsymbol{b}^{(\!(i_j,n]\!]_{\mathbb{N}}}$ refers to $(b_{i_j+1}, b_{i_j+2}, \ldots, b_n)$. This provides a compact notation to partition the original range of indices into 3 pieces: $[\![1, i_j)\!)$, $\{\!|\, i_j \,|\!\}$, and $(\!(i_j, n]\!]$. Formally, we are actually using the hybrid sets $\{\!|\, (i_j)^1 \,|\!\}$ but we omit multiplicity of one.

Next we use the pointwise sum $\oplus$ we reconstruct n-dimensional vectors from our pieces. We then construct a $(k-1)$-rectangle using these vectors as in (5.1). Hence we will have terms of the forms:

$$[\![a_1, b_1]\!]_{\mathbb{R}} \times \ldots \times [\![a_{i_{j-1}}, b_{i_{j-1}}]\!]_{\mathbb{R}} \times [\![a_{i_j}, a_{i_j}]\!]_{\mathbb{R}} \times [\![a_{i_{j-1}}, b_{i_{j-1}}]\!]_{\mathbb{R}} \times \ldots \times [\![a_n, b_n]\!]_{\mathbb{R}}$$

and

$$[\![a_1, b_1]\!]_{\mathbb{R}} \times \ldots \times [\![a_{i_{j-1}}, b_{i_{j-1}}]\!]_{\mathbb{R}} \times [\![b_{i_j}, b_{i_j}]\!]_{\mathbb{R}} \times [\![a_{i_{j-1}}, b_{i_{j-1}}]\!]_{\mathbb{R}} \times \ldots \times [\![a_n, b_n]\!]_{\mathbb{R}}$$

Unlike the previous intervals which were over integers, each of these range continuously over $\mathbb{R}$. In each Cartesian product, the terms at i_j: $[\![a_{i_j}, a_{i_j}]\!]$ and $[\![b_{i_j}, b_{i_j}]\!]$ are both 0-rectangles. Since we defined the sequence i_j by $a_{i_j} \neq b_{i_j}$, these 0-rectangles are replacing 1-rectangles in $[\![\boldsymbol{a}, \boldsymbol{b}]\!]$. Hence we are indeed left with a $(k-1)$-rectangle.

5.1.1 Example: *Boundary of a 1-rectangle*

Let $\boldsymbol{a} = (a_1)$ and $\boldsymbol{b} = (b_1)$ be trivial 1-tuples. Then $[\![\boldsymbol{a}, \boldsymbol{b}]\!] = [\![a_1, b_1]\!]$ It follows that:

$$\begin{aligned}
\partial(\ [\![\boldsymbol{a}, \boldsymbol{b}]\!]\ ) &= (-1)^i([\![\boldsymbol{a}^{[\![1,1)\!)}, \boldsymbol{b}^{[\![1,1)\!)}]\!] \times \{\!\{\ a_1\ \}\!\} \times [\![\boldsymbol{a}^{(\!(1,1]\!]}, \boldsymbol{b}^{(\!(1,1]\!]}]\!] \\
&\quad \ominus [\![\boldsymbol{a}^{[\![1,1)\!)}, \boldsymbol{b}^{[\![1,1)\!)}]\!] \times \{\!\{\ b_1\ \}\!\} \times [\![\boldsymbol{a}^{(\!(1,1]\!]}, \boldsymbol{b}^{(\!(1,1]\!]}]\!]) \\
&= \ominus [\![\boldsymbol{a}^{\emptyset}, \boldsymbol{b}^{\emptyset}]\!] \times \{\!\{\ a_1\ \}\!\} \times [\![\boldsymbol{a}^{\emptyset}, \boldsymbol{b}^{\emptyset}]\!] \oplus [\![\boldsymbol{a}^{\emptyset}, \boldsymbol{b}^{\emptyset}]\!] \times \{\!\{\ b_1\ \}\!\} \times [\![\boldsymbol{a}^{\emptyset}, \boldsymbol{b}^{\emptyset}]\!] \\
&= \left\{\!\!\left\{\ a^{-1}, b^{1}\ \right\}\!\!\right\}
\end{aligned}$$

One may notice the similarity between this result and the (second) fundamental theorem of calculus:

$$\int_a^b F'(x)\ \mathrm{d}x = F(b) - F(a)$$

Which one could easily rewrite as $\int_{[\![a,b]\!]} F'(x)\ \mathrm{d}x = \sum(\partial([\![a, b]\!]))$. Indeed, this is why we have defined the boundary function as such, but more general statements await. We defined the boundary for not just intervals on $\mathbb{R}$ but k-rectangle in $\mathbb{R}^n$.

5.1.2 Example: *Boundary of a 3-rectangle*

Let $\boldsymbol{a} = (0, 0, 0)$ and $\boldsymbol{b} = (1, 1, 1)$. Omitting the intermediate step, we find the boundary of $[\![\boldsymbol{a}, \boldsymbol{b}]\!]$ to be:

$$\begin{aligned}
\partial(\ [\![\boldsymbol{a}, \boldsymbol{b}]\!]\ ) = &\ \ominus\ (\{\!\{\ 0\ \}\!\} \times [\![0, 1]\!] \times [\![0, 1]\!])\ \oplus\ (\{\!\{\ 1\ \}\!\} \times [\![0, 1]\!] \times [\![0, 1]\!]) \\
&\oplus\ ([\![0, 1]\!] \times \{\!\{\ 0\ \}\!\} \times [\![0, 1]\!])\ \ominus\ ([\![0, 1]\!] \times \{\!\{\ 1\ \}\!\} \times [\![0, 1]\!]) \\
&\ominus\ ([\![0, 1]\!] \times [\![0, 1]\!] \times \{\!\{\ 0\ \}\!\})\ \oplus\ ([\![0, 1]\!] \times [\![0, 1]\!] \times \{\!\{\ 1\ \}\!\})
\end{aligned}$$

This may not be the most enlightening expression on its own. In Figure 5.1 below, the 3-rectangle given by $[\![\boldsymbol{a}, \boldsymbol{b}]\!]$ can be seen as a cube in three dimensions. Physically, the 3-rectangle is a solid cube and includes all interior points. The boundary meanwhile are just the rectangular

outer faces, which conveniently, there are also six to match the six terms of $\partial[\![\boldsymbol{a}, \boldsymbol{b}]\!]$.

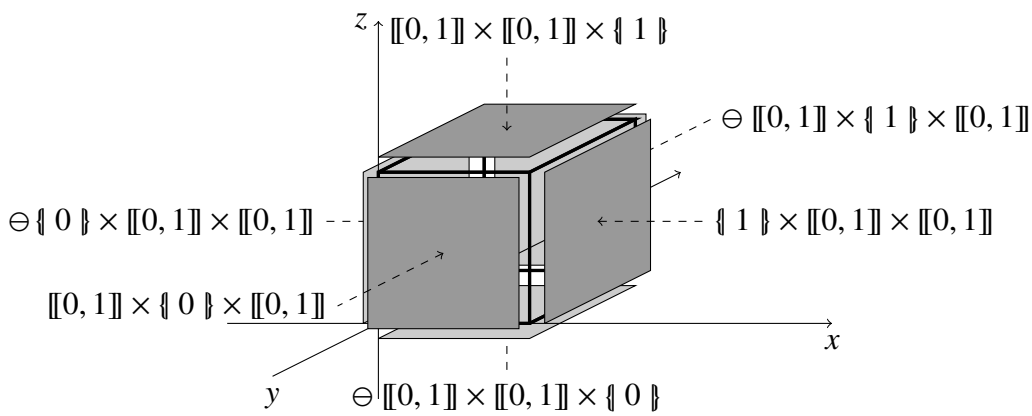


Figure 5.1: The unit cube in $\mathbb{R}^3$ with positive orientation can be represented as the 3-rectangle: $[\![(0,0,0),(1,1,1)]\!]$ is shown as a wire-frame. The six faces that make up its boundary are shaded and labeled with their corresponding terms.

There are several ways to interpret and visualize the $\oplus$ and $\ominus$ sign associated with each face. Most naturally in $\mathbb{R}^3$ for 2-rectangles is to give each a front and back side with the sign determining which to use. Alternatively, a 2-rectangle has a boundary formed by 1-rectangles which when drawn as arrows, will all meet head-to-tail. This induces a clockwise or counter-clockwise cycle around the edge of the rectangle and so $\circlearrowright$ and $\circlearrowleft$ are also commonly used. This can be seen in Figure 5.2. One may even notice that the normals produced by both are the same and choose to use that. These are all conceptual tools, which are convenient to use particularly in $\mathbb{R}^2$ and $\mathbb{R}^3$. There may not be such a nice physical interpretation in other spaces.

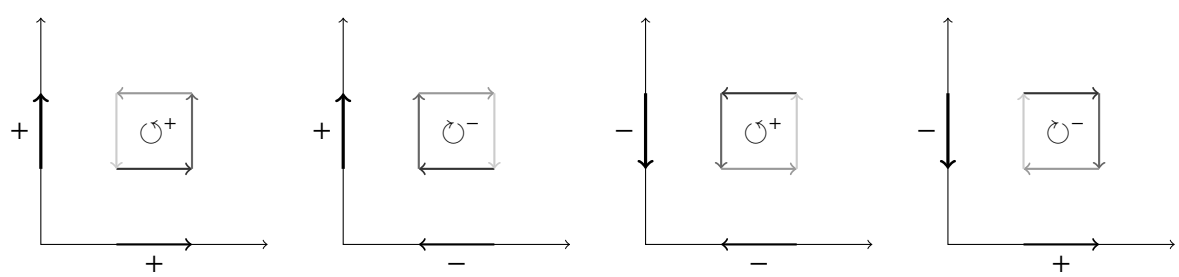


Figure 5.2: One way of visualizing the orientation of 2-rectangles using clockwise and counter-clockwise cycles of arrows for 1-rectangles. The boundary of $[\![a,b]\!]\times[\![c,d]\!]$ becomes the cycle: $(a,c) \to (b,c) \to (b,d) \to (a,d) \to (a,c)$. Showing the relationship between $[\![a,b]\!] \times [\![c,d]\!]$ and $[\![b,a]\!] \times [\![d,c]\!]$

5.2 Chains

In fact, we have already seen k-chains without mentioning them explicitly. The boundary of a $(k+1)$-rectangle was the sum $\oplus$, of $2(k+1)$, k-rectangles. Chains are not restricted to being boundaries of some larger $(k+1)$ rectangle; any linear combination of k-rectangles will do.

Definition We denote the Abelian group of of all k-rectangles in X as $C_k(X)$ (omitting X when obvious by context). An element $c \in C_k(\mathrm{X})$ is called a ***k*-chain on** X and is of the form:

$$c = \bigoplus_{c_i \in X} \lambda_i c_i$$

with integer coefficients λ_i and k-rectangles in c_i. If coefficients λ_i are ± 1 and c is *locally finite* (i.e. each c_i intersects with only finitely many c_j that have non-zero coefficients) then we say that c is a **domain of integration**.

Since k-chains are just linear combinations of k-rectangles, we naturally extend many of our definitions linearly as well. The integral $\int_c f$ of a k-chain $c = \bigoplus_i \lambda_i c_i$ is defined as $\lambda_i \int_{c_i} f + \lambda_2 \int_{c_2} f + \ldots$. Doing the same for the boundary operator ∂ we have:

$$\partial_k : C_k \to C_{k-1}$$
$$\partial_k(c) = \bigoplus_{i=1}^{k} \lambda_i \partial_k(c_i)$$

Elegantly, the boundary operator now maps k-chains to $(k-1)$-chains!

$$\ldots \xleftarrow{\partial_{k-1}} C_{k-1} \xleftarrow{\partial_k} C_k \xleftarrow{\partial_{k+1}} C_{k+1} \xleftarrow{\partial_{k+2}} \ldots$$

5.2.1 Example: *Boundary of a boundary*

This implies should be able to compute the *boundary of a boundary* of some chain $c \in C_{k+1}$ by composing the boundary function with itself as in:

$$\partial_k(\partial_{k+1}(c)) = ?$$

For $c \in C_1$ (i.e c is a 1-rectangle), then $\partial_1(c) \in C_0$ is a set of points. Since the boundary of any isolated point is empty, ∂_0 *always* maps to $\emptyset$. So instead let us consider the case when $c \in C_2$ is a 2-rectangle given by: $[\![a_1, b_1]\!] \times [\![a_2, b_2]\!]$ for $a_1 \neq b_1$ and $a_2 \neq b_2$:

$$\begin{aligned}
\partial_1(\partial_2([\![a_1, b_1]\!] \times [\![a_2, b_2]\!])) = &\ \ominus\, \partial_1(\{\!|\ 0\ |\!\} \times [\![0, 1]\!]) \ \oplus\ \partial_1(\{\!|\ 1\ |\!\} \times [\![0, 1]\!]) \\
&\oplus\, \partial_1([\![0, 1]\!] \times \{\!|\ 0\ |\!\}) \ \ominus\ \partial_1([\![0, 1]\!] \times \{\!|\ 1\ |\!\}) \\
= &\ \ominus\, (\ominus \{\!|\ (0,0)\ |\!\} \oplus \{\!|\ (0,1)\ |\!\}) \ \oplus\ (\ominus \{\!|\ (1,0)\ |\!\} \oplus \{\!|\ (1,1)\ |\!\}) \\
&\oplus\, (\ominus \{\!|\ (0,0)\ |\!\} \oplus \{\!|\ (1,0)\ |\!\}) \ \ominus\ (\ominus \{\!|\ (0,1)\ |\!\} \oplus \{\!|\ (1,1)\ |\!\}) \\
= &\ \emptyset
\end{aligned}$$

Geometrically this can be seen below that the boundary of a rectangle are its edges. The boundary of these edges are the corners; but each corner occurs with both postive and negative sign cancelling. Often stated as "$\partial\partial = 0$", this identity is not unique to 2-rectangles but holds for higher dimensions as well.

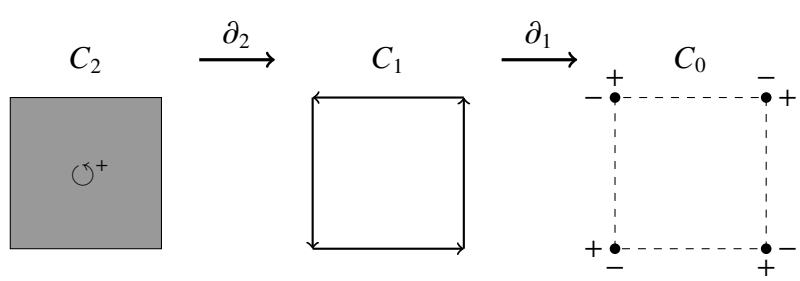


Figure 5.3: The boundary of rectangle gives a cycle of oriented edges. Taking the boundary of again, at each corner, the negative boundary of one edge will be canceled by the positive boundary of the preceding edge.

Let $[\![\boldsymbol{a}, \boldsymbol{b}]\!]$ be a k-rectangle in $\mathbb{R}^n$. Then we have:

$$\begin{aligned}\partial_k \partial_{k-1} \left([\![\boldsymbol{a}, \boldsymbol{b}]\!]\right) = \bigoplus_{j=1}^{k} (-1)^j \Big(& \partial_{n-1} \left([\![\boldsymbol{a}^{[\![1,n]\!]}, \quad \boldsymbol{b}^{[\![1,i_j)\!]} \oplus \boldsymbol{a}^{[\![i_j]\!]} \oplus \boldsymbol{b}^{(\!(i_j,n]\!]}]\!] \right) \\ & \ominus \partial_{n-1} \left([\![\boldsymbol{a}^{[\![1,i_j)\!]} \oplus \boldsymbol{b}^{[\![i_j]\!]} \oplus \boldsymbol{a}^{(\!(i_j,n]\!]}, \quad \boldsymbol{b}^{[\![1,n]\!]}]\!] \right) \Big)\end{aligned}$$

$$\begin{aligned}= \bigoplus_{j=1}^{k} \bigoplus_{\ell=1}^{k-1} (-1)^{j+\ell} \; & [\![(\boldsymbol{a}^{[\![1,n]\!]}), \quad (\boldsymbol{b}^{[\![1,i_j)\!] \oplus (\!(i_j,i_{j,\ell})\!] \oplus (\!(i_{j,\ell},n]\!]} \oplus \boldsymbol{a}^{[\![i_j]\!] \oplus [\![i_{j,\ell}]\!]})]\!] \\ & \ominus [\![(\boldsymbol{a}^{[\![1,i_{j,\ell})\!] \oplus (\!(i_{j,\ell},n]\!]} \oplus \boldsymbol{b}^{[\![i_{j,\ell}]\!]}), \quad (\boldsymbol{b}^{[\![1,i_j)\!] \oplus (\!(i_j,n]\!]} \oplus \boldsymbol{a}^{[\![i_j]\!]})]\!] \\ & \ominus [\![(\boldsymbol{a}^{[\![1,i_j)\!] \oplus (\!(i_j,n]\!]} \oplus \boldsymbol{b}^{[\![i_j]\!]}), \quad (\boldsymbol{b}^{[\![1,i_{j,\ell})\!] \oplus (\!(i_{j,\ell},n]\!]} \oplus \boldsymbol{a}^{[\![i_{j,\ell}]\!]})]\!] \\ & \oplus [\![(\boldsymbol{a}^{[\![1,i_j)\!] \oplus (\!(i_j,i_{j,\ell})\!] \oplus (\!(i_{j,\ell},n]\!]} \oplus \boldsymbol{b}^{[\![i_j]\!] \oplus [\![i_{j,\ell}]\!]}), \quad (\boldsymbol{b}^{[\![1,n]\!]})]\!]\end{aligned}$$

Note that we have i_j and i'_ℓ; after applying the first boundary operator, one dimension of the k-rectangle is degenerate. Hence for each sequence: $\{i_j\}_{j=1}^{k}$ we construct $\{i_{j,\ell}\}_{\ell=1}^{k-1}$ given by:

$$i_{j,1}, \ldots, i_{j,k-1} = i_1, \ldots, \widehat{i_j}, \ldots, i_k$$

The double sum iterates over all pairs but $\oplus$ commutes so the $(k-2)$-rectangle with degenerate dimensions $[\![i_j]\!] \oplus [\![i_{j,\ell}]\!]$ will be iterated over twice. The sequences depend on one another so it is not as simple as simply swapping ℓ and j:

$$[\![i_j]\!] \oplus [\![i_{j,\ell}]\!] = \begin{cases} [\![i_\ell]\!] \oplus [\![i_{\ell,j-1}]\!] & j > \ell \\ [\![i_{\ell+1}]\!] \oplus [\![i_{\ell+1,j}]\!] & j \leq \ell \end{cases}$$

So each term representing a $(k-2)$-rectangle will occur twice in the sum. Once with the iteration (j, ℓ) and once with $(\ell, j-1)$ or $(\ell+1, j)$. In either case, $(-1)^{j+\ell}$ is inverted meaning the two cubes will cancel. Leaving us with the boundary of a boundary being empty. By linearity this extends to all chains as well as the sum of empty sets is of course still empty.

5.3 Stokes' Theorem

This result mirrors the earlier $\mathrm{d}\,\mathrm{d} = 0$ and this duality goes deeper. The boundary ∂ maps a $k+1$-chain to a k-chain while the exterior derivative d mapped a k-form to a $k+1$-form. Sometimes this is written out as:

$$\ldots \xleftarrow{\partial} C_{k-1} \xleftarrow{\partial} C_k \xleftarrow{\partial} C_{k+1} \xleftarrow{\partial} \ldots$$

$$\ldots \xrightarrow{\mathrm{d}} \Lambda^{k-1} \xrightarrow{\mathrm{d}} \Lambda^k \xrightarrow{\mathrm{d}} \Lambda^{k+1} \xrightarrow{\mathrm{d}} \ldots$$

Stokes' Theorem is an important result which links the two even closer and generalizes many classical theorems including the fundamental theorem of calculus, Green's theorem and the divergence theorem.

Given a $k-1$-form ω and k chain M:

$$\int_{\partial M} \omega = \int_M d\omega \qquad \text{(Stokes' Theorem)}$$

Proof First we will consider Stokes theorem for the standard cube $I^k = [0,1]^k \subset \mathbb{R}^k$. In the previous section we saw how cumbersome representing the faces in ∂I^k, could be. We will denote the faces of I^k by $I^k_{i=0}$ and $I^k_{i=1}$ for the i-th faces of I^k. This allows us to rewrite the boundary more succinctly as:

$$\partial(I^k) = \bigoplus_{i=1}^{k} (-1)^i \left(I^k_{i=0} \ominus I^k_{i=1} \right)$$

A $k-1$-form ω can be written as the sum

$$\omega = \sum_{i=1}^{k} \omega_i = \sum_{i=1}^{k} f_i \ \mathrm{d}x_1 \wedge \ldots \wedge \widehat{\mathrm{d}x_i} \wedge \ldots \wedge \mathrm{d}x_k$$

but since everything: the integrals, d and ∂ are all linear, we can work using just one of these

terms. Assuming Stokes' theorem holds for ω_i then we immediately have it for ω as well:

$$\begin{aligned}
\int_{\partial\Omega} \omega &= \int_{\partial\Omega} (\omega_1 + \ldots + \omega_k) \\
&= \int_{\partial\Omega} \omega_1 + \ldots + \int_{\partial\Omega} \omega_k \\
&= \int_{\Omega} d\omega_1 + \ldots + \int_{\Omega} d\omega_k \\
&= \int_{\Omega} (d\omega_1 + \ldots + d\omega_k) \\
&= \int_{\Omega} d(\omega_1 + \ldots + \omega_k) = \int_{\Omega} d\omega
\end{aligned}$$

To compute $d\omega$, we have for each term in the sum:

$$\begin{aligned}
d\left(f_i \, \mathrm{d}x_1 \wedge \ldots \wedge \widehat{\mathrm{d}x_i} \wedge \ldots \wedge \mathrm{d}x_k\right) &= \mathrm{d}f_i \wedge \mathrm{d}x_1 \wedge \ldots \wedge \widehat{\mathrm{d}x_i} \wedge \ldots \wedge \mathrm{d}x_k \\
&= \left(\sum_{j=1}^{k} \frac{\partial f_i}{\partial x_j} \, \mathrm{d}x_j\right) \wedge \mathrm{d}x_1 \wedge \ldots \wedge \widehat{\mathrm{d}x_i} \wedge \ldots \wedge \mathrm{d}x_k
\end{aligned}$$

But for $j \neq i$, there will be a duplicate $\mathrm{d}x_j$ term and this collision will cause the term to go to zero. Hence only one term in the sum, $i = j$ will actually result in a non-zero term:

$$\begin{aligned}
\mathrm{d}\left(f_i \, \mathrm{d}x_1 \wedge \ldots \wedge \widehat{\mathrm{d}x_i} \wedge \ldots \wedge \mathrm{d}x_k\right) &= \left(\frac{\partial f_i}{\partial x_i} \, \mathrm{d}x_i\right) \wedge \mathrm{d}x_1 \wedge \ldots \wedge \hat{\mathrm{d}x_i} \wedge \ldots \wedge \mathrm{d}x_k \\
&= (-1)^{i-1} \frac{\partial f_i}{\partial x_i} \, \mathrm{d}x_1 \wedge \ldots \wedge \mathrm{d}x_k
\end{aligned}$$

Since this is an integral over the canonical basis $\mathrm{x} = (x_1, \ldots, x_k)$ we can remove the wedge

products and integrate as normal.

$$\begin{aligned}
\int_{[0,1]^k} \mathrm{d}\omega_i &= (-1)^{i-1} \int_{[0,1]^k} \frac{\partial f_i}{\partial x_i} \, \mathrm{d}(x_1, \ldots, x_k) \\
&= (-1)^{i-1} \int_{[0,1]^{k-1}} \left(\int_0^1 \frac{\partial f_i}{\partial x_i} \, \mathrm{d}x_i \right) \mathrm{d}(x_1, \ldots, \widehat{x_i}, \ldots, x_k) \\
&= (-1)^{i-1} \left(\int_{[0,1]^{k-1}} f_i(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_k) \, \mathrm{d}(x_1, \ldots, \widehat{x_i}, \ldots, x_k) \right. \\
&\quad \left. - \int_{[0,1]^{k-1}} f_i(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_k) \, \mathrm{d}(x_1, \ldots, \widehat{x_i}, \ldots, x_k) \right)
\end{aligned}$$

The trick here being Fubini's Theorem allowing us to evaluate the iterated integral in whichever order we choose. On the other side of the equality we have:

$$\int_{\partial I^k} \omega_i = \sum_{j=1}^{k} (-1)^j \int_{I^k_{j=0}} \omega_i - \int_{I^k_{j=1}} \omega_i$$

but $I^k_{j=0}$ is just a $k-1$-rectangle embedded in $\mathbb{R}^k$ by the map:

$$(x_1, \ldots x_{k-1}) \mapsto (x_1, \ldots, x_{j-1}, 0, x_j, \ldots x_{k-1})$$

So we could alternatively think of $I^k_{j=0} : I^{k-1} \to I^k$ as just a change in coordinates:

$$\begin{aligned}
\int_{\partial I^k} \omega_i &= \sum_{j=1}^{k} (-1)^j \left(\int_{I^{k-1}} (I^k_{j=0})^* \omega_i - \int_{I^{k-1}} (I^k_{j=1})^* \omega_i \right) \\
&= \sum_{j=1}^{k} (-1)^j \left(\int_{I^{k-1}} f_i(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_k) \, \mathrm{d}x_1 \wedge \ldots \wedge \widehat{\mathrm{d}x_i} \wedge \ldots \wedge \mathrm{d}x_k \right. \\
&\quad \left. - \int_{I^{k-1}} f_i(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_k) \, \mathrm{d}x_1 \wedge \ldots \wedge \widehat{\mathrm{d}x_i} \wedge \ldots \wedge \mathrm{d}x_k \right) \\
&= (-1)^i \left(\int_{I^{k-1}} f_i(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_k) \, \mathrm{d}(x_1, \ldots, \widehat{x_i}, \ldots, x_k) \right. \\
&\quad \left. - \int_{I^{k-1}} f_i(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_k) \, \mathrm{d}(x_1, \ldots, \widehat{x_i}, \ldots, x_k) \right)
\end{aligned}$$

In the final step we observe that all terms in the sum for $i \neq j$ end up disappearing leaving

us with just the term for $i = j$. Clearly, both sides of the equation are the same and so Stokes' theorem holds for the standard cube. From here, the remaining cases build on one another are quite straight-forward. For a singular cube c, we have:

$$\begin{aligned}
\int_{\partial c} \omega &= \int_{c(\partial([0,1]^k))} \omega \\
&= \int_{\partial([0,1]^k)} c^* \omega \\
&= \int_{[0,1]^k} \mathrm{d}c^* \omega \\
&= \int_{[0,1]^k} c^* \, \mathrm{d}\omega \\
&= \int_c d\omega
\end{aligned}$$

And for a chain $C = a_1 c_1 + \ldots + a_n c_n$ made up of singular cubes:

$$\begin{aligned}
\int_C d\omega &= \int_{a_1 c_1 + \ldots + a_n c_n} \mathrm{d}\omega \\
&= a_1 \int_{c_1} \mathrm{d}\omega + \ldots + a_n \int_{c_n} \mathrm{d}\omega \\
&= a_1 \int_{\partial c_1} \omega + \ldots + a_n \int_{\partial c_n} \omega \\
&= \int_{a_1 \partial c_1 + \ldots a_n \partial c_n} \omega \\
&= \int \partial(a_1 c_1 + \ldots + a_n c_n)\omega \\
&= \int_{\partial C} \omega
\end{aligned}$$

And so we have Stokes' theorem on general chains.

5.3.1 Example: *Contour Integral*

Generalized partitions can extend past the original domain but so far this has been done through relatively obvious extensions. Sometimes the additional regions that are added and negated can

be quite unexpected. *Contour integration* is a method for solving integrals that can involve the usage of curious looking regions to solve. Consider the following function defined on the real numbers:

$$f(x) = \frac{1}{x^4 + 1}$$

and suppose we wish to compute the integral over the entire real line:

$$\int_{-\infty}^{\infty} f(x)\,\mathrm{d}x$$

This is a difficult integral to evaluate directly so instead of treating f as a real function, $f : \mathbb{R} \to \mathbb{R}$, we instead consider it as a complex function: $f : \mathbb{C} \to \mathbb{C}$.

Let C_1 be the straight line curve from $-r$ to r: $C_1(t) = rt - (1-t)r$ and C_2 the circular arc with radius r from r to $-r$: $C_2(t) = re^{it}$. Finally let C the combined curve: $C = C_1 \oplus C_2$ Then our desired value is the integral over C_1 as r goes to infinity:

$$\int_{C_1} f(x)\,\mathrm{d}x = \oint_C f(z)\,\mathrm{d}z - \int_{C_2} f(z)\,\mathrm{d}z$$

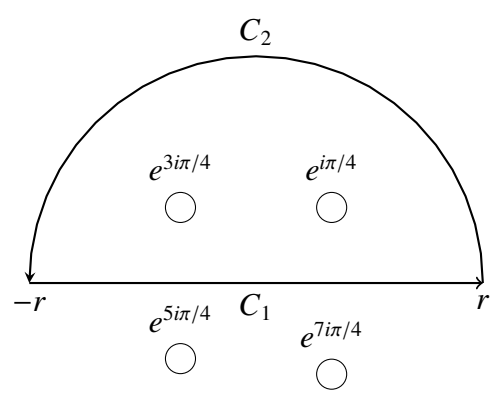


Figure 5.4: The contours C_1 and C_2 and with the four singularities of f marked.

To solve the integral on C we can use the Cauchy residue theorem: a special case of the generalized Stokes' theorem. C is a closed simple path and f is holomorphic everywhere

except for a finite set of points $\{a_k\}_{i=1}^n$ and so we have:

$$\oint_C f(z)\,\mathrm{d}z = 2\pi i \sum_{k=1}^{n} \mathrm{Res}(f, a_k)$$

And so we must compute the residues of the two singularities above the real line: $W_8^1 = e^{i\pi/4}$ and $W_8^3 = e^{3i\pi/4}$. These are both simple poles and can be computed as follows:

$$\mathrm{Res}(f, W_8^1) = \lim_{z\to W_8^1} \frac{z - W_8^1}{z^4 + 1} = \lim_{z\to W_8^1} \frac{1}{4z^3} = (1/4)W_8^{-3} = (1/4)W_8^5$$

$$\mathrm{Res}(f, W_8^3) = \lim_{z\to W_8^3} \frac{z - W_8^3}{z^4 + 1} = \lim_{z\to W_8^3} \frac{1}{4z^3} = (1/4)W_8^{-9} = (1/4)W_8^7$$

Converting these out of polar coordinates, we have $W_8^7 = \left(\frac{\sqrt{2}}{2} - \frac{\sqrt{2}}{2}i\right)$ and $W_8^5 = \left(-\frac{\sqrt{2}}{2} - \frac{\sqrt{2}}{2}i\right)$:

$$\oint_C f(z)\,\mathrm{d}z = 2\pi i\left(\frac{1}{4}\left(\frac{\sqrt{2}}{2} - \frac{\sqrt{2}}{2}i\right) + \frac{1}{4}\left(-\frac{\sqrt{2}}{2} - \frac{\sqrt{2}}{2}i\right)\right) = \frac{\pi}{\sqrt{2}}$$

To compute the integral over C_2, we make the substitutions, $z = re^{i\theta}$ and $dz = ire^{i\theta}d\theta$:

$$\int_{C_2} f(z)\,\mathrm{d}z = \int_0^{\pi} \frac{ire^{i\theta}}{r^4e^{4i\theta} + 1}\,\mathrm{d}\theta$$

And then simply observe that as r goes to infinity, this integral goes to zero and the term for C_2 drops off. Leaving us with just:

$$\int_{-\infty}^{\infty} f(x)\,\mathrm{d}x = \frac{\pi}{\sqrt{2}}$$

Chapter 6

Convolution of Piecewise Functions

Convolution is an operation which takes two functions and produces a third. It takes one of the two input functions, and modifies one by mirroring and translating. The resulting function is then the overlap between one of these functions as a function of the translation. Visually, this can be seen below in Figure 6.1.

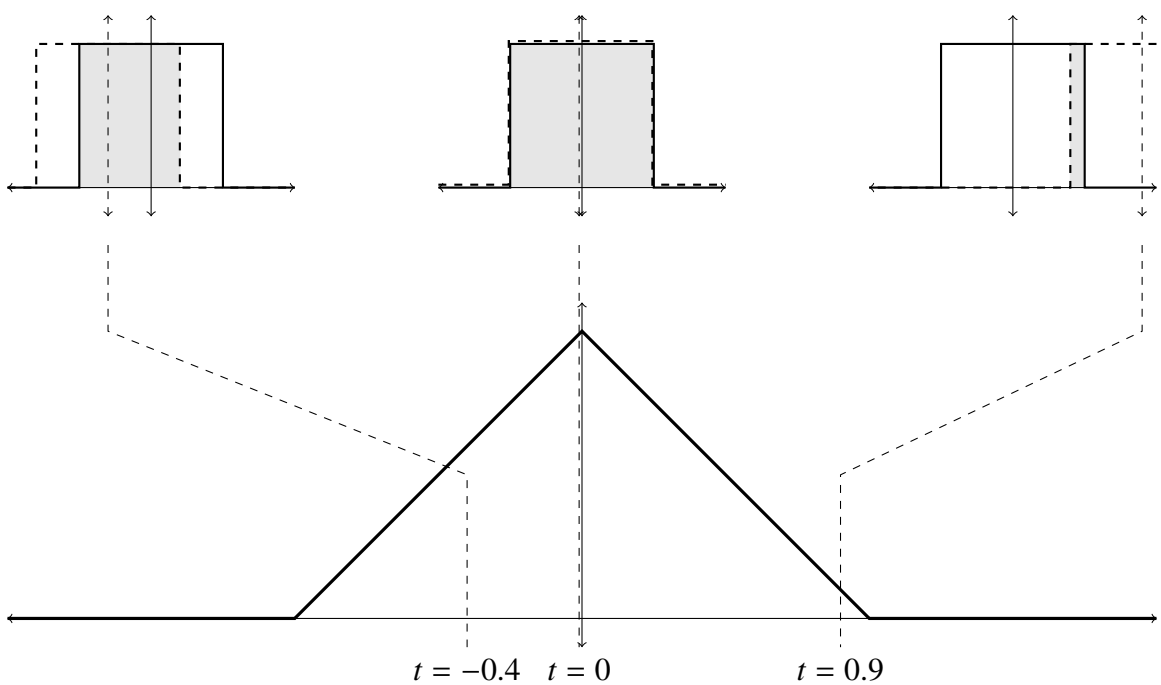


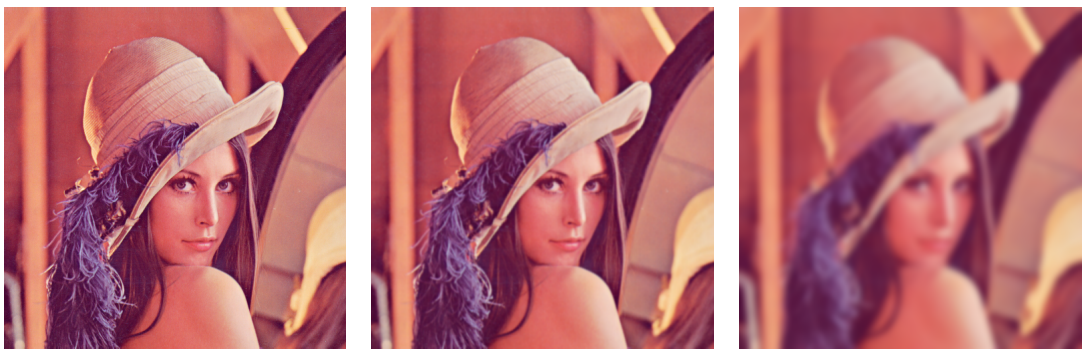
Figure 6.1: The convolution of the box signal $f(t) = g(t) = \left(0^{((-\infty,-0.5))} \oplus 1^{[\![-0.5,0.5]\!]} \oplus 0^{((0.5,\infty))}\right)$ with itself.

Formally this, equates to the following definition for convolution over continuous domains:

Definition The **convolution** $*$, of two functions F and G is defined as:

$$(F * G)(t) = \int_{-\infty}^{\infty} F(\tau)\, G(t - \tau)\, \mathrm{d}\tau \tag{6.1}$$

and in the case of discrete linear convolution, summation would replace integration. In this equation, t represents the translation of G as well as the input for $(F * G)$ while τ is internal to the integral and varies over the real line.



(a) Original Image (b) 1px Gaussian blur (c) 5px Gaussian blur

Figure 6.2: 512x512px "Lena"(a) with a 1px (b) and 5px (c) Gaussian blur applied. Gaussian blurring is accomplished by convolving an image with a Gaussian kernel and is commonly used in image processing to reduce noise prior to edge detection.

Convolution has applications in many areas of mathematics and engineering. One very common use in image processing is in blurring. *Gaussian blurring* is the result of a convolution in 2-dimensions of an image with the Gaussian distribution function:

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{6.2}$$

Blurring an image in this way reduces noise and greatly increases the efficacy of subsequent edge detection. In statistics, a (simple) *moving average* can be represented as a convolution by a rectangular pulse while more generally, weighted moving averages can be made by convolving with other functions.

6.1 Convolution of Piecewise Functions

CAS such as Maple and Mathematica are quite adept at solving integrals. Convolution of elementary functions generally poses no problem. When convolving two piecewise continuous functions, many possible intervals arise and the conditionals that arise can quickly overwhelm them unaided.

We are interested in *Symbolic Linear Convolution* (of piecewise continuous functions). The typical approach is to first consider for convolution of "one piece" functions [10, 22]. By "one-piece" functions we mean functions which are restricted to a single interval and zero everywhere else. We will consider two functions, F and G defined as:

$$F(x) = f^{[a_f,b_f)}(x) = \begin{cases} f(x) & a_f \leq x < b_f \\ 0 & \text{otherwise} \end{cases} \tag{6.3}$$

$$G(x) = g^{[a_g,b_g)}(x) = \begin{cases} g(x) & a_g \leq x < b_g \\ 0 & \text{otherwise} \end{cases} \tag{6.4}$$

for which we would like to compute the convolution $(F * G)$. To reduce the total number of cases generated, it is generally also assumed that $b_f - a_f \leq b_g - a_g$. Assuming that F is non-zero over a shorter interval is not that strong an assumption as convolution is commutative; if it is not the case we can rearrange $F * G$ to $G * F$. To see this, simply apply the substitute $\tau' = t - \tau$ in equation (7.1):

$$\begin{aligned} (F * G)(t) &= \int_{-\infty}^{\infty} F(\tau)G(t-\tau)\,\mathrm{d}\tau \\ &= \int_{\infty}^{-\infty} F(t-\tau')G(\tau')\,(-1)\,\mathrm{d}\tau' \\ &= \int_{-\infty}^{\infty} G(\tau')F(t-\tau')\,\mathrm{d}\tau' \\ &= (G * F)(t) \end{aligned} \tag{6.5}$$

Thus we can assume that our static function is also the function with the shorter interval. Since F and G are zero outside of their respective intervals, we do not need to integrate over the entire real line. F is our static function, so $[a_f, b_f)$ would be sufficient. For a tight boundary, we have the following:

$$
\begin{aligned}
(F * G)(t) &= \int_{-\infty}^{\infty} F(\tau)\, G(t-\tau)\, \mathrm{d}\tau \\
&= \int_{a_f}^{b_f} f(\tau)\, G(t-\tau)\, \mathrm{d}\tau \\
&= \begin{cases} \int_{a_f}^{t-a_g} f(\tau)\, g(t-\tau)\, \mathrm{d}\tau & (a_f + a_g) \le t < (b_f + a_g) \\ \int_{a_f}^{b_f} f(\tau)\, g(t-\tau)\, \mathrm{d}\tau & (b_f + a_g) \le t < (a_f + b_g) \\ \int_{t-b_g}^{b_f} f(\tau)\, g(t-\tau)\, \mathrm{d}\tau & (a_f + b_g) \le t < (b_f + b_g) \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
\tag{6.6}
$$

These regions can be visualized as b in Figure 6.3 where two rectangular pulses are convolved. If both functions had equal length non-zero intervals (i.e. $b_f - a_f = b_g - a_g$), then the central plateau would be empty (as in Figure 6.1).

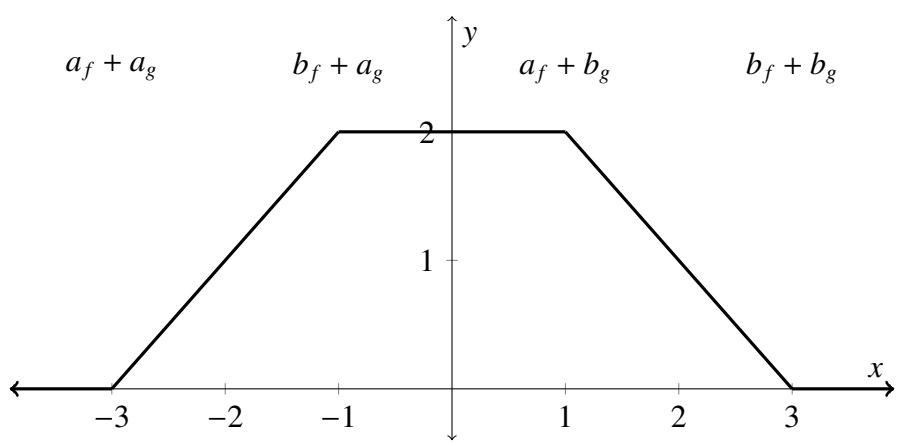


Figure 6.3: Convolution of length 1 and 2 rectangular pulses. Given the functions $F = 1^{[-1,1)}$ and $G = 1^{[-2,2)}$, there are three non-zero regions in $(F * G)$.

Another formulation presented by Cîrnu [9] and Cavicchi [8] is to use:

$$(F * G)(t) = \begin{cases} \int_{\max(a_f,\, t-b_g)}^{\min(b_f, t-a_g)} f(\tau) \cdot g(t-\tau)\,\mathrm{d}\tau & (a_f + a_g) \le t < (b_f + b_g) \\ 0 & \text{otherwise} \end{cases} \tag{6.7}$$

Although this may appear to reduce the number of cases, expanding the min and max will cause just as many cases to reappear.

To extend this to piecewise continuous function, we simply treat each piecewise function as the sum of "one-piece" functions. Given functions, $F = \sum_i f_i^{P_i}$ and $G = \sum_j g_j^{Q_j}$ where $\{P_i\}$, $\{Q_j\}$ are each sets of disjoint intervals, and $f_i^{P_i}$, $g_j^{Q_j}$ are all "one-piece" functions. The convolution of $F * G$ is the sum of pairwise convolution:

$$\begin{aligned} \left(\left(\sum_i f_i^{P_i}\right) * \left(\sum_j g_j^{Q_j}\right)\right)(t) &= \int_{-\infty}^{\infty} \left(\sum_i f_i^{P_i}\right)(\tau) \cdot \left(\sum_j g_j^{Q_j}\right)(t-\tau)\,\mathrm{d}\tau \\ &= \sum_i \sum_j \int_{-\infty}^{\infty} f_i^{P_i}(\tau) \cdot g_j^{Q_j}(t-\tau)\,\mathrm{d}\tau \\ &= \sum_i \sum_j \left(f_i^{P_i} * g_j^{P_j}\right) \end{aligned} \tag{6.8}$$

To summarize, the algorithm for convolution of two piecewise functions is as follows:

1. Each function is converted into a sum of disjoint function intervals.
2. Each function interval in one function is convolved with each function interval in the other.
3. The final result is the sum of all function interval convolutions.

This is the typical approach to convolution of piecewise functions as presented by West and McClellan [22]. When the boundaries between regions is symbolic, then we may not be able to determine which interval is longer. Another approach involving hybrid functions will be presented in Section 6.2. Concerns with intervals where one boundary point is at infinity have also been raised [10]. Techniques to handle this will be investigated in Section 6.3.

6.2 Hybrid Function Convolution

Our exposition for hybrid set convolution will appear very similar to that from the previous section and can be seen as a replacement for step 2 in the algorithm. Again we will be interested in the convolution of "one-piece" functions which we will use to build up piecewise continuous functions. Assuming two hybrid one-piece functions $f^{[a_f,b_f)}$ and $g^{[a_g,b_g)}$ which are 0 outside of the intervals $[a_f, b_f)$ and $[a_g, b_g)$ respectively. Then the **hybrid convolution of $f^{[a_f,b_f)}$ and $g^{[a_g,b_g)}$** is:

$$
\begin{aligned}
(f^{[a_f,b_f)} \ * \ g^{[a_g,b_g)})(t) = \mathcal{R}_+ \Bigg(& \left(\int_{[\![a_f,\, t-a_g)\!]} f(\tau)\ g(t-\tau)\,\mathrm{d}\tau \right)^{[\![a_f+a_g,\, b_f+a_g)\!]} \\
& \oplus \left(\int_{[\![a_f,\, b_f)\!]} f(\tau)\ g(t-\tau)\,\mathrm{d}\tau \right)^{[\![b_f+a_g,\, a_f+b_g)\!]} \\
& \oplus \left(\int_{[\![t-b_g,\, b_f)\!]} f(\tau)\ g(t-\tau)\,\mathrm{d}\tau \right)^{[\![a_f+b_g,\, b_f+b_g)\!]} \Bigg)(t)
\end{aligned}
\tag{6.9}
$$

The first thing one should note is the similarity between this expression and (6.6). But, *we do not enforce* $b_f - a_f \leq b_g - a_g$ as we did in Section 6.1. Instead both cases will be handled by our generalized partition structure. If the integral $f(t)g(t-\tau)$ is is easier to compute than $g(t)f(t-\tau)$ then the convolution can, of course, still be commuted. This could be due to the nature of functions for f and g or if f and g are a part of a larger sum with identical sub-functions on different regions. Then these integrals could potentially be combined, provided they have the same integrand, resulting in fewer overall integrals to compute. The ordering of f and g is no longer dictated by the relative length of their respective intervals.

When $b_f - a_f \leq b_g - a_g$ then the three oriented intervals will be disjoint and the two equations are identical. Otherwise, if $b_f - a_f > b_g - a_g$, then the interval $[\![b_f + a_g,\ a_f + b_g)\!]$ will have a negative orientation. The intervals $[\![a_f+a_g,\ b_f+a_g)\!]$, $[\![b_f+a_g,\ a_f+b_g)\!]$ and $[\![a_f+b_g,\ b_f+b_g)\!]$ still forms a reducible (i.e. everywhere multiplicity one) generalized partition over $[a_f+a_g,\ b_f+b_g)$. Outside this region the function is zero as expected.

Suppose we are in this second case and we wish to evaluate the convolution at a point t which is in all three intervals. This occurs when $(a_f + b_g) \leq t < (b_f + a_g)$ and we have:

$$[\![a_f + a_g,\ b_f + a_g)\!](t) = 1$$

$$[\![b_f + a_g,\ a_f + b_g)\!](t) = -1$$

$$[\![a_f + b_g,\ b_f + b_g)\!](t) = 1$$

Simplifying the +-reduction we then get:

$$\begin{aligned}(f^{[a_f,b_f)} \ * \ g^{[a_g,b_g)})(t) = &\left(\int_{[\![a_f,\ t-a_g)\!]} f(\tau)\ g(t-\tau)\,\mathrm{d}\tau\right) \\ &-\left(\int_{[\![a_f,\ b_f)\!]} f(\tau)\ g(t-\tau)\,\mathrm{d}\tau\right) \\ &+\left(\int_{[\![t-b_g,\ b_f)\!]} f(\tau)\ g(t-\tau)\,\mathrm{d}\tau\right) \end{aligned} \tag{6.10}$$

All three of these integrals have the same integrand so we can use bi-linearity to move the sum to be over the domains of integration. These domains then cancel nicely to leave us with:

$$\begin{aligned}(f^{[a_f,b_f)} \ * \ g^{[a_g,b_g)})(t) &= \int_{[\![a_f,\ t-a_g)\!] \ominus [\![a_f,\ b_f)\!] \oplus [\![t-b_g,\ b_f)\!]} (\tau)\ g(t-\tau)\,\mathrm{d}\tau \\ &= \int_{[\![t-b_g,\ t-a_g)\!]} f(\tau)\ g(t-\tau)\,\mathrm{d}\tau \end{aligned} \tag{6.11}$$

Let us now look at a concrete example with some actual numbers.

6.2.1 Example: *Hybrid Convolution*

In Figure 6.3 we saw the convolution of $1^{[-1,1)}$ with $1^{[-2,2)}$. We know that convolution is commutative so computing $1^{[-2,2)} * 1^{[-1,1)}$ we already know what to expect. We will label these as 1_f and 1_g to differentiate and to prevent confusion by reminding us that the object we are dealing with is $x \mapsto 1$ rather than the number 1 itself.

$$
\begin{aligned}
(1_f^{[-2,2)} \ * \ 1_g^{[-1,1)})(t) = \mathcal{R}_+ \Bigg(& \left(\int_{[\![-2,\, t-1)\!]} f(\tau)\, g(t-\tau)\, \mathrm{d}\tau \right)^{[\![-3,\, 1)\!]} \\
& \oplus \left(\int_{[\![-2,\, 1)\!]} f(\tau)\, g(t-\tau)\, \mathrm{d}\tau \right)^{[\![1,\, -1)\!]} \\
& \oplus \left(\int_{[\![t-1,\, 2)\!]} f(\tau)\, g(t-\tau)\, \mathrm{d}\tau \right)^{[\![-1,\, 3)\!]} \Bigg)(t)
\end{aligned}
\tag{6.12}
$$

Already this is promising as we can see the set of end-points: $\{-3, -1, 1, 3\}$ agrees with our previous example. Let us consider three points $t_1 \in [-3, -1)$, $t_2 \in [-1, 1)$ and $t_3 \in [1, 3)$. We omit the derivations but encourage the reader to convince themselves that each is correct.

$$
(1_f^{[-2,2)} \ * \ 1_g^{[-1,1)})(t_1) \ = \ \int_{[\![-2,\, t_1-(-1))\!]} 1_f(\tau)\ 1_g(t_1-\tau)\, \mathrm{d}\tau \ = \ t_1 + 3
$$

First we should note that at no point in the integral do we attempt to evaluate 1_f or 1_g outside of their original domains $[-2, 2]$ and $[-1, 1]$ respectively. Thus it is safe to replace $1_f(\tau) \cdot 1_g(t-\tau)$ with 1 inside the integral. From here, the integral is trivially evaluated and is as expected.

For t_3, only the third term has non-zero multiplicity and by an identical argument as for t_1 we have:

$$
(1_f^{[-2,2)} \ * \ 1_g^{[-1,1)})(t_3) \ = \ \int_{[\![t-1,2)\!]} 1_f(\tau)\ 1_g(t_3-\tau)\, \mathrm{d}\tau \ = \ 3 - t_3
$$

Finally, t_2 deviates from this pattern slightly as t_2 is in *all three* oriented intervals. By the same derivation as we used in the previous section we can use equation (6.11):

$$
(1_f^{[a_f,b_f)} \ * \ 1_g^{[a_g,b_g)})(t_2) \ = \ \int_{[\![t_2-1,\, t_2-(-1))\!]} f(\tau)\, g(t-\tau)\, \mathrm{d}\tau \ = \ 2
$$

For any other point t which is not in $[-3, 3)$, then all three oriented intervals will have multiplicity zero. Simplifying the +-reduction we have, $\mathcal{R}_+(\emptyset) = e_+ = 0$ and so (6.12) evaluates correctly everywhere.

6.3 Infinite Intervals

The method presented in Section 6.1 behaves correctly assuming that all the end points are finite. But Evans and McClellan [10] raise two issues that arise when we allow for interval end points at infinity. Namely,

1. Interval lengths can no longer be compared to ensure the first interval is shorter than the second.
2. Indeterminant arithmetic may occur in computing new endpoints of the form $+\infty - \infty$

We have already shown that hybrid function convolution is not concerned with the relative length of functions. Clearly, the first point should not be a concern for hybrid convolution. Less clear is that, invalid arithmetic on interval endpoints can be ignored as well! Unlike the 16 different cases used by Evans and McClellan, we can extend hybrid convolution from finite-only to mixed finite and infinite end points with no additional logic.

The first important observation is that indeterminate arithmetic can only occur in *internal* end-points. By this we mean that of the four end-points: $\{a_f + a_g,\ b_f + a_g,\ a_f + b_g,\ b_f + b_g\}$, the points $a_f + a_g$ and $b_f + b_g$ can always safely be evaluated. If b_f were $-\infty$ or a_f were ∞ then the function interval $f^{[a_f, b_f)}$ is actually $f^{\emptyset}$ and can be ignored (similarly $b_g \neq -\infty$ and $a_g \neq \infty$). Now, recall that the three hybrid set intervals that occurred in equation (6.9) were:

$$[\![a_f + a_g,\ b_f + a_g)\!], \qquad [\![b_f + a_g,\ a_f + b_g)\!], \qquad \text{and} \qquad [\![a_f + b_g, b_f + b_g)\!]$$

So if indeterminate arithmetic does occur among end-points it would occur at least twice. This will prove useful in allowing us to have these points cancel each other out. For example, suppose $b_f + a_g = \infty + (-\infty) = \perp$ is undefined. Even though, $[\![a_f + a_g,\ b_f + a_g)\!]$ and $[\![b_f + a_g,\ a_f + b_g)\!]$ may separately be undefined, their sum is not:

$$[\![a_f + a_g,\ b_f + a_g)\!] \oplus [\![b_f + a_g,\ a_f + b_g)\!] = [\![a_f + a_g,\ a_f + b_g)\!]$$

Before we can just add these intervals, each is of course attached to a function so we return to the discussion of compatibility from section 2.2. After substituting the previously assumed $b_f = \infty$ and $a_g = -\infty$, we can see that both the integrands are identical and the domains nearly so as well:

$$\left(\int_{[\![a_f,\, t+\infty)\!]} f(\tau)\; g(t-\tau)\,\mathrm{d}\tau\right)^{[\![-\infty,\, \perp)\!]}$$

and

$$\left(\int_{[\![a_f,\, \infty)\!]} f(\tau)\; g(t-\tau)\,\mathrm{d}\tau\right)^{[\![\perp,\, a_f+b_g)\!]}$$

For $t \neq -\infty$, the domains of integration match as well. Since the contained functions are identical, these two hybrid functions are compatible. Putting this all together:

$$\begin{aligned}(f^{[a_f,\infty)} \;*\; g^{[-\infty,b_g)})(t) = \mathcal{R}_+\Bigg(&\left(\int_{[\![a_f,\, \infty)\!]} f(\tau)\; g(t-\tau)\,\mathrm{d}\tau\right)^{[\![-\infty,\, a_f+b_g)\!]} \\ &\oplus \left(\int_{[\![t-b_g,\, \infty)\!]} f(\tau)\; g(t-\tau)\,\mathrm{d}\tau\right)^{[\![a_f+b_g,\, \infty)\!]}\Bigg)(t)\end{aligned} \tag{6.13}$$

Each end-point can be either finite or infinite; left end-points are either finite or $-\infty$ while right end-points are finite or $+\infty$. So with 4 end-points and 2 possible cases for each, this leads to 16 possible cases to convolve one-piece functions. It can be shown that equation (6.9) without modification is correct in all cases. A full enumeration of this can be found in Appendix A.

6.4 Discrete Convolution

Until now we have assumed F and G are continuous functions. While the continuous is of more historical interest, the discrete case is more widely used in digital signal processing. From a theoretical perspective, very little is different between the continuous and discrete case; it is primarily a swap from integrals $\int$ to sums $\sum$ and evaluating functions at points $f(x)$ to indexing in an array $f[x]$.

Definition The **discrete convolution** of two sequences F and G defined as:

$$(F * G)[t] = \sum_{\tau=-\infty}^{\infty} F[\tau] \cdot G[t-\tau] \tag{6.14}$$

or for sequences with non-negative indexing:

$$(F * G)[t] = \sum_{\tau=0}^{\infty} F[\tau] \cdot G[t-\tau] \tag{6.15}$$

When convolving sequences, limiting the boundaries which need to be summed over is just as important as in the continuous case. In particular, when working with digital inputs represented by arrays, it is very important to not attempt to index outside of the bounds of the arrays. Therefore it is important to get tight bounds on the range of τ. Again, this is a simple swap as well:

$$\begin{aligned}(f^{[a_f,b_f]} * g^{[a_g,b_g]})[t] = \mathcal{R}_+ \Bigg(& \left(\sum_{\tau \in [\![a_f,\, t-a_g]\!]} f[\tau]\, g[t-\tau] \right)^{[\![a_f+a_g,\, b_f+a_g)\!)} \\ & \oplus \left(\sum_{\tau \in [\![a_f,\, b_f]\!]} f[\tau]\, g[t-\tau] \right)^{[\![b_f+a_g,\, a_f+b_g)\!)} \\ & \oplus \left(\sum_{\tau \in [\![t-b_g,\, b_f]\!]} f[\tau]\, g[t-\tau] \right)^{[\![a_f+b_g,\, b_f+b_g]\!]} \Bigg)[t]\end{aligned} \tag{6.16}$$

One must be even more careful with bounds in the discrete case compared to the continuous. Excepting the Dirac function and similar constructions, including or excluding the end-points will not usually affect the evaluation of an integral:

$$\int_{(\!(a,b)\!)} f(x)\,\mathrm{d}x = \int_{(\!(a,b]\!]} f(x)\,\mathrm{d}x = \int_{[\![a,b)\!)} f(x)\,\mathrm{d}x = \int_{[\![a,b]\!]} f(x)\,\mathrm{d}x$$

The difference between each interval is measure 0 and so excepting pathological functions, the integrals should be equal. The same does not hold for summation; the openness or "closed-

ness" of an interval matters even in the typical use case.

That being said, the boundary points between intervals are safe to fall in either direction. For $t = b_f + a_g$, evaluating either the sum

$$\sum_{\tau \in [\![a_f,\, t-a_g]\!]} f[\tau]\, g[t-\tau] \qquad \text{or} \qquad \sum_{\tau \in [\![a_f,\, b_f]\!]} f[\tau]\, g[t-\tau]$$

equates to the same thing. So we can have the intervals $[\![a_f + a_g, b_f + a_g)\!)$ and $[\![b_f + a_g, a_f + b_g)\!)$ or the intervals $[\![a_f + a_g, b_f + a_g]\!]$ and $(\!(b_f + a_g, a_f + b_g)\!)$; either would be correct. Similarly we can also move the point at $a_f + b_g$ from the third term or the second term.

Here we are using closed intervals for f and g. This is at odds with typical array iteration, like Python's `range(..)` function which tend to use *closed-open* intervals. Unlike the contiuous case, there is no structural difference between an open or closed interval; the choice of using one over the other is usually a matter of whichever gives the nicest indices by avoiding $+1$'s or -1's. So we can handle the difference by converting $[a, b) = [a, b-1]$

$$\begin{aligned}(f^{[a_f,b_f)} * g^{[a_g,b_g)})[t] = \mathcal{R}_+\Bigg(&\left(\sum_{\tau \in [\![a_f,\, t-a_g]\!]} f[\tau]\, g[t-\tau]\right)^{[\![a_f+a_g,\, b_f+a_g)\!)} \\ &\oplus \left(\sum_{\tau \in [\![a_f,\, b_f)\!)} f[\tau]\, g[t-\tau]\right)^{[\![b_f+a_g,\, a_f+b_g)\!)} \\ &\oplus \left(\sum_{\tau \in (\!(t-b_g,\, b_f)\!)} f[\tau]\, g[t-\tau]\right)^{[\![a_f+b_g,\, b_f+b_g-1)\!)}\Bigg)[t]\end{aligned} \tag{6.17}$$

6.5 Implementation

Implementation for continuous symbolic convolution was done in *Maple*. Maple is able to correctly determine many cancellations but requires assistance for a few cases with symbolic and infinite end-points. For the most part, the cases enumerated in Appendix A follow a few patterns:

1. Merging terms with indeterminate end-points. (Cases 6, 7, 9, 11, 13, 14, and 15)

2. Remove any terms over empty intervals. (Cases 1, 2, 3, 5, 7, 10, 11, 13, and 14)

3. Manipulate intervals to find a disjoint partition and combine integrals using linearity of domains (Cases 4, 8, and 12)

To merge indeterminate end-points, we use the local variables `afbg` and `bfag` to represent the sums $a_f + b_g$ and $b_f + a_g$ respectively. If the sums are well defined, then we can apply the substitution, otherwise `afbg` and `bfag` are left symbolic.

```
Convolve := proc(f, af, bf, g, ag, bg)
 local afbg, bfag, out, temp;
 if(af+bg != undefined) then afbg := af+bg; fi;
 if(bf+ag != undefined) then bfag := bf+ag; fi;
 ...
```

In cases where these sums are undefined, these terms will disappear (cancellation shown in Appendix A). However, to assist Maple in finding this cancellation we must leave the sum as a symbolic term. We can avoid any potentially undefined integrals by delaying the evaluation of the functions inside the integral. We use the symbolic `fg(x)` to represent `f(x)*g(t-x)` to prevent Maple from unwrapping the integrals until we have determined ranges which the integrals will be evaluated much like pseudo-functions from section 2.2:

```
 ...
 out:=(int(fg(x),x=af..t-ag))*(OrientedInterval(af+ag,bfag))(t)
   +(int(fg(x),x=af..bf))*(OrientedInterval(bfag,afbg))(t)
   +(int(fg(x),x=t-bg..bf))*(OrientedInterval(afbg,bf+bg))(t);
 ...
```

To ensure that future steps function smoothly, we remove any $t - \infty$ or $t + \infty$ terms that can arise in the bounds of integration. This is just a simple substitution:

```
 ...
out:=subs(t-infinity=-infinity,out);
out:=subs(t+infinity=infinity,out);
...
```

If `afbg` or `bfag` are symbolic ($a_f + b_g$ or $b_f + a_g$ are undefined respectively) then these symbolic terms then disappear when the piecewise OrientedIntervals are converted to Heaviside functions: `convert(%, Heaviside, t)`. Converting this back to a piecewise function with `convert(%, piecewise,t)` results in a more readable expression.

However, since the Heaviside function is undefined at 0, this can lead to point discontinuities in cases where $a_f + b_g$ and $b_f + a_g$ *are* perfectly well defined. To remedy this, before we convert to Heaviside and back, we should first attempt to convert it to a piecewise function. Converting to piecewise will fail if `bfag` or `afbg` are incomparable (undefined):

```
...
try temp := convert(out,piecewise,t);
catch:
   try temp:=convert(convert(out,Heaviside),piecewise,t);
   catch: temp := out;
   end try;
finally out := temp;
end try;
...
```

By this point, Maple has already removed any terms with empty intervals so all that remains is combining integrals by linearity. In most cases this can be handled by `Combine` in the `IntegrationTools` package. `Combine` is unable to combine when end-points are infinite so again we will substitute a symbolic term:

```
...
try temp:=Combine(subs(infinity=infty,out));
   catch temp:=out;
finally out:=subs(infty=infinity,temp); end try;
...
```

The final step is to replace `fg(x)` with `f(x)*g(t-x)`

```
...
out:=subs(fg(x)=f(x)*g(t-x), out)
```

Hopefully this process strikes the reader as being quite *simple* as most of the functionality to reduce the hybrid function equation is already provided by Maple. This is intentional, not only is it easier to implement but it illustrates the point that hybrid convolution can in fact be a simpler framework with less case-based logic. Admittedly, some strange looking hacks are required to "make it work", such as the conversion to Heaviside and back to piecewise. With the exception of deciding to leave `afbf` and `bfag` symbolic, many of these steps are entirely optional. They result in a simplified, nicer looking expression but the equation can be correctly interpretted without them.

Chapter 7

Conclusions

The primary objective of this thesis was to extend [7] by investigating further applications of hybrid sets and functions. In this we focused on three main examples: block matrix algebra, integration and convolution of piecewise functions. Along the way, several smaller examples were considered as well. As primarily a demonstration of notation, rational numbers and polynomials were shown as examples of hybrid sets. Hybrid relations and functions were also demonstrated with maximum flow problems and arithmetic with piecewise defined functions.

Piecewise function arithmetic was the first demonstration of the strength of generalized partitions. Whereas the naive approach to adding an n piece to an m piece function leads to $O(n \cdot m)$ piece function in the general case. Using generalized partitions and careful selection of a generalized partition, we could reduce this to $O(n + m - 1)$. This also required the use of pseudo-functions and leaving functions unevaluated to allow for cancellations to occur first. If this is not done, we run the risk of attempting to evaluate a function outside of its defined domain.

Addition of symbolic block matrices and vectors can be performed in the same fashion as piecewise function arithmetic as was previously shown in [7]. Filling the role of the $\xi(i, j, k)$ function from [17], oriented intervals were introduced. Thus far, this notation has proved very intuitive and flexible. In particular, this sufficiently lightened the notation to allow for the

development of block matrix multiplication. Doing so we could formulate an expression using "wrong ordering" of breakpoints along the mutual axis of the multiplicands but still result in the correct product. Thus we can work with matrices which have symbolically sized blocks without resorting to a case-based approach.

Hybrid sets were then shown to be a good model for domains of integration. Numeric integration using hybrid sets as was then shown for both the Riemann and Lebesgue integral. This allows for more natural manipulation of domains turning $\int_a^b + \int_b^c = \int_a^c$ from a theorem to a trivial result of bi-linearity. As we moved to integration on differential forms, hybrid sets also showed up naturally with the boundary operator. In conjunction with Stokes' theorem, there are even more opportunities to manipulate the domains of an integral.

Finally, we applied generalized partitions towards convolution of piecewise functions. The typical approach for convolving one-piece interval functions involves two cases depending on which interval is longer. Whether one uses two distinct expressions or commutes the convolution, both methods are unsatisfactory when interval bounds are symbolic and relative length cannot be compared. With oriented intervals we can use the same equations in a length oblivious manner. This method handles infinite end points as well without resorting to 6 equations and a 16 case table.

Generalized partitions present a more algebraic way to subdivide an object. The usefulness of this has long been half-realized for domains of integration but there are many areas where they could aptly be applied. By moving from traditional sets to hybrid sets, we gain a proper notion of a negative set. Despite the sometimes overlapping notation, this notion should not be confused with set complement And so the normally imposed condition of disjointness for partitions can be dropped for generalized partitions. Assuming a good mapping for what is meant by a negative set (generally through the $*$-reduction $\mathcal{R}_*$), generalized partitions present a useful language. Even if the examples presented in this thesis are not of direct interest to the reader, hopefully the techniques are and that they may be applied towards many more endeavors.

Bibliography

[1] D. Anelli, S. Loeb, E. Damiani, and O. D'Antona. Getting results with negative thinking. *arXiv preprint math/9502214*, 2008.

[2] Colin G. Bailey and Joseph S. Gull, Dean W .and Oliveira. Hypergraphic oriented matroid relational dependency flow models of chemical reaction networks. *arXiv preprint arXiv:0902.0847*, 2009.

[3] Jean-Pierre Banâtre, Pascal Fradet, Yann Radenac, et al. Generalised multisets for chemical programming. *Mathematical Structures in Computer Science*, 16(4):557–580, 2006.

[4] Wayne D. Blizard. Multiset theory. *Notre Dame Journal of Formal Logic*, 30(1):36–66, 12 1988.

[5] Wayne D. Blizard. Negative membership. *Notre Dame Journal of Formal Logic*, 31(3):346–368, 06 1990.

[6] George Boole. *An investigation of the laws of thought: on which are founded the mathematical theories of logic and probabilities*, volume 2. Walton and Maberly, 1854.

[7] Jacques Carette, Alan P. Sexton, Volker Sorge, and Stephen M. Watt. Symbolic domain decomposition. In *Proceedings of 17th Symposium on the Integration of Symbolic Computation and Mechanised Reasoning, (Calculemus 2010)*, pages 172–188, Paris, France, July 2010. Springer Verlag LNAI 6167.

[8] T.J. Cavicchi. Simplified method for analytical evaluation of convolution integrals. *IEEE Transactions on Education*, 45(2):142–144, May 2002.

[9] Mircea I. Cîrnu. Calculation of convolution products of piecewise defined functions and some applications. *Journal of Information Systems and Operations Management*, 6:41–52, 2012.

[10] Brian L. Evans and James H. McClellan. Algorithms for symbolic linear convolution. In *Proceedings of Asilomar Conference on Signals, Systems and Computers*, volume 2, pages 948–953. IEEE, 1994.

[11] K.P. Girish and John Sunil Jacob. On multiset topologies. *Theory and Applications of Mathematics & Computer Science*, 2(1):37–52, 2012.

[12] Theodore Hailperin. *Boole's logic and probability: a critical exposition from the standpoint of contemporary algebra, logic and probability theory*. Elsevier, 1986.

[13] Donald E. Knuth. *The Art of Computer Programming, Volume 2: Seminumerical Algorithms*. Addison-Wesley, 2014.

[14] Monica D. Lam, Edward E. Rothberg, and Michael E. Wolf. The cache performance and optimizations of blocked algorithms. *ACM SIGOPS Operating Systems Review*, 25(Special Issue):63–74, 1991.

[15] Daniel Loeb. Sets with a negative number of elements. *Advances in Mathematics*, 91(1):64–74, 1992.

[16] Wolfgang Reisig. *Petri nets: an introduction*. Springer-Verlag, 1985.

[17] Alan P. Sexton, Volker Sorge, and Stephen M. Watt. Abstract matrix arithmetic. In *10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, 2008. SYNASC'08*, pages 61–68, Timişoara, Romania, September 2008.

[18] D. Singh, A. Ibrahim, T. Yohanna, and J. Singh. An overview of the applications of multisets. *Novi Sad Journal of Mathematics*, 37(3):73–92, 2007.

[19] D. Singh, A.M. Ibrahim, T. Yohana, and J.N. Singh. Complementation in multiset theory. *International Mathematical Forum*, 6(38):1877–1884, 2011.

[20] D. Singh, A.M. Ibrahim, T. Yohanna, and J.N. Singh. A systematization of fundamentals of multisets. *Lecturas Matematicas*, 29:33–48, 2008.

[21] T. Tao. Differential forms and integration. Technical report, Tech. Rep., Department of Mathematics, UCLA, 2007.

[22] Kevin A. West and J.H. McClellan. Symbolic convolution. *IEEE Transactions on Education*, 36(4):386–393, 1993.

[23] Hassler Whitney. Characteristic functions and the algebra of logic. *Annals of Mathematics*, 34:405–414, 1933.

[24] N.J. Wildberger. A new look at multisets. *preprint*, University of New South Wales, Sydney, 2003.

Appendix A

Convolution with Infinite End-Points

Table A.1: All possible cases of finite and infinite end-points. Finite end-points are denoted with F. Infinite left end-points are denoted $-\infty$ and infinite right end-points are denoted ∞.

	a_f	b_f	a_g	b_g			a_f	b_f	a_g	b_g
Case 0:	F	F	F	F		Case 8:	$-\infty$	F	F	F
Case 1:	F	F	F	∞		Case 9:	$-\infty$	F	F	∞
Case 2:	F	F	$-\infty$	F		Case 10:	$-\infty$	F	$-\infty$	F
Case 3:	F	F	$-\infty$	∞		Case 11:	$-\infty$	F	$-\infty$	∞
Case 4:	F	∞	F	F		Case 12:	$-\infty$	∞	F	F
Case 5:	F	∞	F	∞		Case 13:	$-\infty$	∞	F	∞
Case 6:	F	∞	$-\infty$	F		Case 14:	$-\infty$	∞	$-\infty$	F
Case 7:	F	∞	$-\infty$	∞		Case 15:	$-\infty$	∞	$-\infty$	∞

When convolving one-piece functions with infinite end-points, there are 4 end-points which can each be either finite or infinite. As such there are 2^4 possible combinations of end-point types shown in the table above. Throughout all calculations in this section, the integrands will not change, only the domains. So we define the function C as a sort of restricted convolution:

$$C_{[\![x,y)\!]}(t) = \int_{[\![x,y)\!]} f(\tau)g(t-\tau)\, d\tau \tag{A.1}$$

Which can be used to condense equation (6.9), the definition for hybrid convolution, into:

$$(f^{[a_f,b_f)} \ * \ g^{[a_g,b_g)}) = \mathcal{R}_+ \Big(C_{[\![a_f,\, t-a_g)\!]}{}^{[\![a_f+a_g,\, b_f+a_g)\!]} \oplus C_{[\![a_f,\, b_f)\!]}{}^{[\![b_f+a_g,\, a_f+b_g)\!]} \oplus C_{[\![t-b_g,\, b_f)\!]}{}^{[\![a_f+b_g,\, b_f+b_g)\!]} \Big)$$

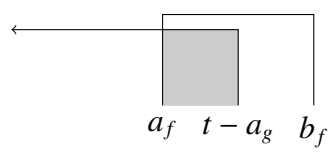
Case 0 has all finite points and was already shown to be correct in Section 6.2 but is listed for completeness. The exposition will not be repeated here.

Case 1 has one infinite point, $b_g = \infty$ and results in an empty interval for the third term. Since it is impossible for t to be in $[\![\infty, \infty)\!]$, we can safely remove this term altogether.

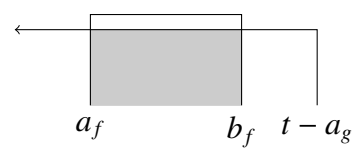
$$
\begin{aligned}
(f^{[a_f,b_f)} \ * \ g^{[a_g,\infty)}) &= \mathcal{R}_+ \left(C_{[\![a_f,\, t-a_g)\!]}{}^{[\![a_f+a_g,\, b_f+a_g)\!]} \oplus C_{[\![a_f,\, b_f)\!]}{}^{[\![b_f+a_g,\, \infty)\!]} \oplus C_{[\![-\infty,\, b_f)\!]}{}^{[\![\infty,\, \infty)\!]} \right) \\
&= \mathcal{R}_+ \left(C_{[\![a_f,\, t-a_g)\!]}{}^{[\![a_f+a_g,\, b_f+a_g)\!]} \oplus C_{[\![a_f,\, b_f)\!]}{}^{[\![b_f+a_g,\, \infty)\!]} \right)
\end{aligned}
$$

Throughout this section we will use sets of diagrams like the one below to verify that our expressions are sensible. The region where the intervals, $[a_f, b_f)$ and $[t - b_g, t - a_g)$ overlap, (shaded in gray) is where the convolution will be non-zero. The bounds of this intersection may depend on t; each case that results in a non-empty intersection will be shown separately. To verify an expression, each diagram should correspond to a term in the convolution and the bounds of the shaded region should correspond to the domain on that term's integral.

(a) $t \in [\![a_f + a_g,\ b_f + a_g)\!]$



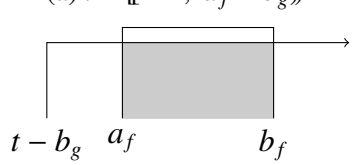
(b) $t \in [\![b_f + a_g, \infty)\!]$



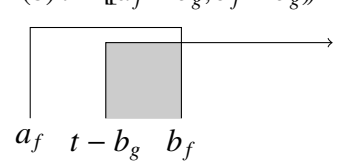
Case 2 also has only one infinite point, $a_g = -\infty$ and also yields an empty interval:

$$
\begin{aligned}
(f^{[a_f,b_f)} \ * \ g^{[-\infty,b_g)}) &= \mathcal{R}_+ \left(C_{[\![a_f,\, \infty)\!]}{}^{[\![-\infty,\, -\infty)\!]} \oplus C_{[\![a_f,\, b_f)\!]}{}^{[\![-\infty,\, a_f+b_g)\!]} \oplus C_{[\![t-b_g,\, b_f)\!]}{}^{[\![a_f+b_g,\, b_f+b_g)\!]} \right) \\
&= \mathcal{R}_+ \left(C_{[\![a_f,\, b_f)\!]}{}^{[\![-\infty,\, a_f+b_g)\!]} \oplus C_{[\![t-b_g,\, b_f)\!]}{}^{[\![a_f+b_g,\, b_f+b_g)\!]} \right)
\end{aligned}
$$

(a) $t \in [\![-\infty,\ a_f + b_g)\!]$



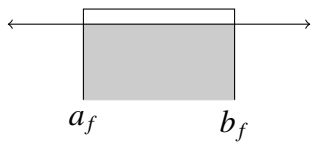
(b) $t \in [\![a_f + b_g, b_f + b_g)\!]$



Case 3 is a combination of both case 1 and 2, resulting in only a single term for the entire real line since both the first and third terms are over empty intervals.

$$
\begin{aligned}
(f^{[a_f,b_f)} \ * \ g^{[-\infty,\infty)}) &= \mathcal{R}_+\Big(C_{[\![a_f,\,\infty)\!]}{}^{[\![-\infty,\,-\infty)\!]} \oplus C_{[\![a_f,\,b_f)\!]}{}^{[\![-\infty,\,\infty)\!]} \oplus C_{[\![-\infty,\,b_f)\!]}{}^{[\![\infty,\,\infty)\!]} \Big) \\
&= \mathcal{R}_+\Big(C_{[\![a_f,\,b_f)\!]}{}^{[\![-\infty,\,\infty)\!]} \Big) = \int_{[\![a_f,\,b_f)\!]} f(\tau)\, g(t-\tau)\, d\tau
\end{aligned}
$$

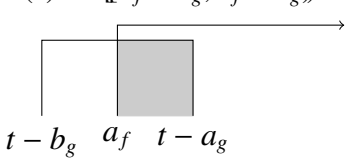
(a) $t \in [\![-\infty,\ \infty)\!]$



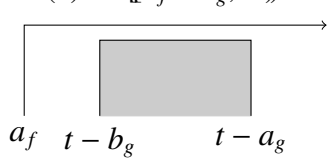
Case 4 (i.e. $b_f = \infty$) is a bit more involved:

$$
\begin{aligned}
(f^{[a_f,\infty)} \ * \ g^{[-\infty,b_g)}) &= \mathcal{R}_+\Big(C_{[\![a_f,\,t-a_g)\!]}{}^{[\![a_f+a_g,\,\infty)\!]} \oplus C_{[\![a_f,\,\infty)\!]}{}^{[\![\infty,\,a_f+b_g)\!]} \oplus C_{[\![t-b_g,\,\infty)\!]}{}^{[\![a_f+b_g,\,\infty)\!]} \Big) \\
&= \mathcal{R}_+\Big(C_{[\![a_f,\,t-a_g)\!]}{}^{[\![a_f+a_g,\,a_f+b_g)\!]\oplus[\![a_f+b_g,\,\infty)\!]} \ominus C_{[\![a_f,\,\infty)\!]}{}^{[\![a_f+b_g,\,\infty)\!]} \oplus C_{[\![t-b_g,\,\infty)\!]}{}^{[\![a_f+b_g,\,\infty)\!]} \Big) \\
&= \mathcal{R}_+\Big(C_{[\![a_f,\,t-a_g)\!]}{}^{[\![a_f+a_g,\,a_f+b_g)\!]} \oplus C_{[\![a_f,\,t-a_g)\!]\ominus[\![a_f,\,\infty)\!]\oplus[\![t-b_g,\,\infty)\!]}{}^{[\![a_f+b_g,\,\infty)\!]} \Big) \\
&= \mathcal{R}_+\Big(C_{[\![a_f,\,t-a_g)\!]}{}^{[\![a_f+a_g,\,a_f+b_g)\!]} \oplus C_{[\![t-b_g,\,t-a_g)\!]}{}^{[\![a_f+b_g,\,\infty)\!]} \Big)
\end{aligned}
$$

(a) $t \in [\![a_f + a_g, a_f + b_g)\!]$



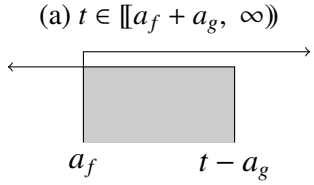
(b) $t \in [\![a_f + b_g, \infty)\!]$



Case 5 $b_f = \infty$, $b_g = \infty$

$$
\begin{aligned}
(f^{[a_f,\infty)} \ * \ g^{[a_g,\infty)}) &= \mathcal{R}_+\Big(C_{[\![a_f,\,t-a_g)\!]}{}^{[\![a_f+a_g,\,\infty)\!]} \oplus C_{[\![a_f,\,\infty)\!]}{}^{[\![\infty,\,\infty)\!]} \oplus C_{[\![-\infty,\,\infty)\!]}{}^{[\![\infty,\,\infty)\!]} \Big) \\
&= \mathcal{R}_+\Big(C_{[\![a_f,\,t-a_g)\!]}{}^{[\![a_f+a_g,\,\infty)\!]} \Big)
\end{aligned}
$$

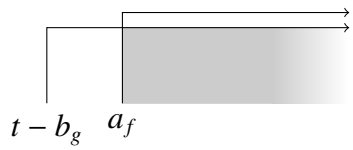
(a) $t \in [\![a_f + a_g,\ \infty)\!]$



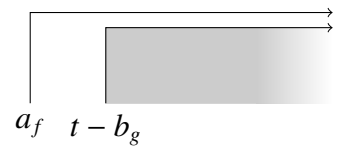
Case 6: $b_f = \infty$, $a_g = -\infty$

$$
\begin{aligned}
(f^{[a_f,\infty)} \ * \ g^{[-\infty,b_g)}) &= \mathcal{R}_+\left(\mathcal{C}_{[\![a_f,\,\infty)\!]}{}^{[\![-\infty,\,\perp)\!]} \oplus \mathcal{C}_{[\![a_f,\,\infty)\!]}{}^{[\![\perp,\,a_f+b_g)\!]} \oplus \mathcal{C}_{[\![t-b_g,\,\infty)\!]}{}^{[\![a_f+b_g,\,\infty)\!]} \right) \\
&= \mathcal{R}_+\left(\mathcal{C}_{[\![a_f,\,\infty)\!]}{}^{[\![-\infty,\,a_f+b_g)\!]} \oplus \mathcal{C}_{[\![t-b_g,\,\infty)\!]}{}^{[\![a_f+b_g,\,\infty)\!]} \right)
\end{aligned}
$$

(a) $t \in [\![-\infty, a_f + b_g)\!]$



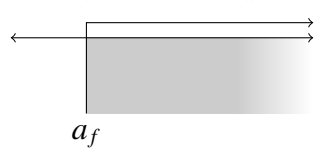
(b) $t \in [\![a_f + b_g, \infty)\!]$



Case 7: $b_f = \infty$, $a_g = -\infty$, $b_g = \infty$

$$
\begin{aligned}
(f^{[a_f,\infty)} \ * \ g^{[-\infty,\infty)}) &= \mathcal{R}_+\left(\mathcal{C}_{[\![a_f,\,\infty)\!]}{}^{[\![-\infty,\,\perp)\!]} \oplus \mathcal{C}_{[\![a_f,\,\infty)\!]}{}^{[\![\perp,\,\infty)\!]} \oplus \mathcal{C}_{[\![-\infty,\,\infty)\!]}{}^{[\![\infty,\,\infty)\!]} \right) \\
&= \mathcal{R}_+\left(\mathcal{C}_{[\![a_f,\,\infty)\!]}{}^{[\![-\infty,\,\infty)\!]} \right)
\end{aligned}
$$

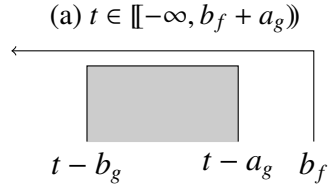
(a) $t \in [\![-\infty,\ \infty)\!]$



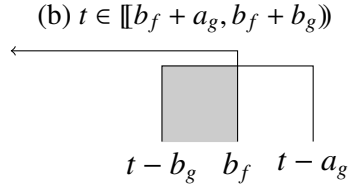
Case 8: $a_f = -\infty$

$$
\begin{aligned}
(f^{[-\infty,b_f)} \ * \ g^{[a_g,b_g)}) &= \mathcal{R}_+\left(\mathcal{C}_{[\![-\infty,\,t-a_g)\!]}{}^{[\![-\infty,\,b_f+a_g)\!]} \oplus \mathcal{C}_{[\![-\infty,\,b_f)\!]}{}^{[\![b_f+a_g,\,-\infty)\!]} \oplus \mathcal{C}_{[\![t-b_g,\,b_f)\!]}{}^{[\![-\infty,\,b_f+b_g)\!]} \right) \\
&= \mathcal{R}_+\left(\mathcal{C}_{[\![-\infty,\,t-a_g)\!]}{}^{[\![-\infty,\,b_f+a_g)\!]} \ominus \mathcal{C}_{[\![-\infty,\,b_f)\!]}{}^{[\![-\infty,\,b_f+a_g)\!]} \oplus \mathcal{C}_{[\![t-b_g,\,b_f)\!]}{}^{[\![-\infty,\,b_f+a_g)\!]\oplus[\![b_f+a_g,b_f+b_g)\!]} \right) \\
&= \mathcal{R}_+\left(\mathcal{C}_{[\![-\infty,\,t-a_g)\!]\ominus[\![-\infty,\,b_f)\!]\oplus[\![t-b_g,\,b_f)\!]}{}^{[\![-\infty,\,b_f+a_g)\!]} \oplus \mathcal{C}_{[\![t-b_g,\,b_f)\!]}{}^{[\![b_f+a_g,b_f+b_g)\!]} \right) \\
&= \mathcal{R}_+\left(\mathcal{C}_{[\![t-b_g,\,t-a_g)\!]}{}^{[\![-\infty,\,b_f+a_g)\!]} \oplus \mathcal{C}_{[\![t-b_g,\,b_f)\!]}{}^{[\![b_f+a_g,b_f+b_g)\!]} \right)
\end{aligned}
$$

(a) $t \in [\![-\infty, b_f + a_g)\!]$



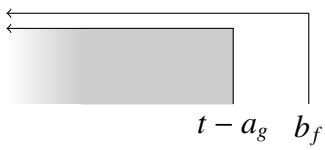
(b) $t \in [\![b_f + a_g, b_f + b_g)\!]$



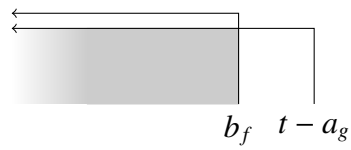
Case 9: $a_f = -\infty$, $b_g = \infty$

$$
\begin{aligned}
(f^{[-\infty,b_f)} \ * \ g^{[a_g,\infty)}) &= \mathcal{R}_+\left(C_{[\![-\infty,\, t-a_g)\!)}{}^{[\![-\infty,\, b_f+a_g)\!)} \oplus C_{[\![-\infty,\, b_f)\!)}{}^{[\![b_f+a_g,\, \perp)\!)} \oplus C_{[\![-\infty,\, b_f)\!)}{}^{[\![\perp,\, \infty)\!)} \right) \\
&= \mathcal{R}_+\left(C_{[\![-\infty,\, t-a_g)\!)}{}^{[\![-\infty,\, b_f+a_g)\!)} \oplus C_{[\![-\infty,\, b_f)\!)}{}^{[\![b_f+a_g,\, \infty)\!)} \right)
\end{aligned}
$$

(a) $t \in [\![-\infty, b_f + a_g)\!)$



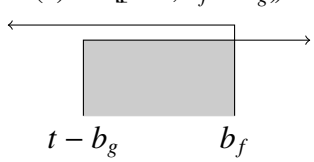
(b) $t \in [\![b_f + a_g, \infty)\!)$



Case 10: $a_f = -\infty$, $a_g = -\infty$

$$
\begin{aligned}
(f^{[-\infty,b_f)} \ * \ g^{[-\infty,b_g)}) &= \mathcal{R}_+\left(C_{[\![-\infty,\, \infty)\!)}{}^{[\![-\infty,\, -\infty)\!)} \oplus C_{[\![-\infty,\, b_f)\!)}{}^{[\![-\infty,\, -\infty)\!)} \oplus C_{[\![t-b_g,\, b_f)\!)}{}^{[\![-\infty,\, b_f+b_g)\!)} \right) \\
&= \mathcal{R}_+\left(C_{[\![t-b_g,\, b_f)\!)}{}^{[\![-\infty,\, b_f+b_g)\!)} \right)
\end{aligned}
$$

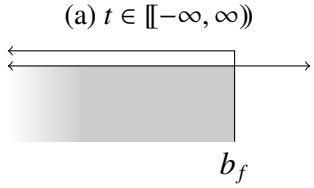
(a) $t \in [\![-\infty, b_f + b_g)\!)$



Case 11: $a_f = -\infty$, $a_g = -\infty$, $b_g = \infty$

$$
\begin{aligned}
(f^{[-\infty,b_f)} \ * \ g^{[-\infty,\infty)}) &= \mathcal{R}_+\left(C_{[\![-\infty,\, \infty)\!)}{}^{[\![-\infty,\, -\infty)\!)} \oplus C_{[\![-\infty,\, b_f)\!)}{}^{[\![-\infty,\, \perp)\!)} \oplus C_{[\![-\infty,\, b_f)\!)}{}^{[\![\perp,\, \infty)\!)} \right) \\
&= \mathcal{R}_+\left(C_{[\![-\infty,\, b_f)\!)}{}^{[\![-\infty,\, \infty)\!)} \right)
\end{aligned}
$$

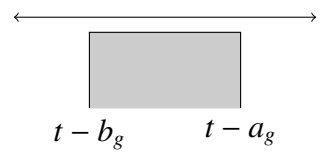
(a) $t \in [\![-\infty, \infty)\!)$



Case 12: $a_f = -\infty$, $b_f = \infty$

$$
\begin{aligned}
(f^{[-\infty,\infty)} \ * \ g^{[a_g,b_g)}) &= \mathcal{R}_+\Big(C_{[\![-\infty,\, t-a_g)\!]}{}^{[\![-\infty,\, \infty)\!]} \oplus C_{[\![-\infty,\, \infty)\!]}{}^{[\![\infty,\, -\infty)\!]} \oplus C_{[\![t-b_g,\, \infty)\!]}{}^{[\![-\infty,\, \infty)\!]} \Big) \\
&= \mathcal{R}_+\Big(C_{[\![-\infty,\, t-a_g)\!] \ominus [\![-\infty,\, \infty)\!] \oplus [\![t-b_g,\, \infty)\!]}{}^{[\![-\infty,\, \infty)\!]} \Big) \\
&= \mathcal{R}_+\Big(C_{[\![[t-b_g,\, t-a_g)\!]}{}^{[\![-\infty,\, \infty)\!]} \Big)
\end{aligned}
$$

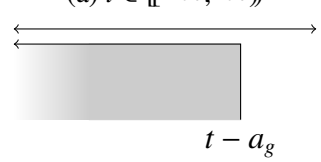
(a) $t \in [\![-\infty,\ \infty)\!]$



Case 13: $a_f = -\infty$, $b_f = \infty$, $b_g = \infty$

$$
\begin{aligned}
(f^{[-\infty,\infty)} \ * \ g^{[a_g,\infty)}) &= \mathcal{R}_+\Big(C_{[\![-\infty,\, t-a_g)\!]}{}^{[\![-\infty,\, \infty)\!]} \oplus C_{[\![-\infty,\, \infty)\!]}{}^{[\![\infty,\, \bot)\!]} \oplus C_{[\![-\infty,\, \infty)\!]}{}^{[\![\bot,\, \infty)\!]} \Big) \\
&= \mathcal{R}_+\Big(C_{[\![-\infty,\, t-a_g)\!]}{}^{[\![-\infty,\, \infty)\!]} \oplus C_{[\![-\infty,\, \infty)\!]}{}^{[\![\infty,\, \infty)\!]} \Big) \\
&= \mathcal{R}_+\Big(C_{[\![-\infty,\, t-a_g)\!]}{}^{[\![-\infty,\, \infty)\!]} \Big)
\end{aligned}
$$

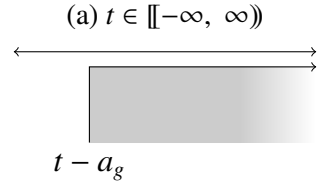
(a) $t \in [\![-\infty,\ \infty)\!]$



Case 14: $a_f = -\infty$, $b_f = \infty$, $a_g = -\infty$

$$
\begin{aligned}
(f^{[-\infty,\infty)} \ * \ g^{[-\infty,b_g)}) &= \mathcal{R}_+\Big(C_{[\![-\infty,\, \infty)\!]}{}^{[\![-\infty,\, \bot)\!]} \oplus C_{[\![-\infty,\, \infty)\!]}{}^{[\![\bot,\, -\infty)\!]} \oplus C_{[\![t-b_g,\, \infty)\!]}{}^{[\![-\infty,\, \infty)\!]} \Big) \\
&= \mathcal{R}_+\Big(C_{[\![-\infty,\, \infty)\!]}{}^{[\![-\infty,\, -\infty)\!]} \oplus C_{[\![t-b_g,\, \infty)\!]}{}^{[\![-\infty,\, \infty)\!]} \Big) \\
&= \mathcal{R}_+\Big(C_{[\![t-b_g,\, \infty)\!]}{}^{[\![-\infty,\, \infty)\!]} \Big)
\end{aligned}
$$

(a) $t \in [\![-\infty,\ \infty)\!]$



Case 15: $a_f = -\infty$, $b_f = \infty$, $a_g = -\infty$, $b_g = \infty$ has all infinite points and is not really what one would think of as a one-piece function at all. The definition of convolution already holds for such functions. That being said:

$$
\begin{aligned}
(f^{[-\infty,\infty)} \ * \ g^{[-\infty,\infty)}) &= \mathcal{R}_+\left(C_{[\![-\infty,\,\infty)\!]}{}^{[\![-\infty,\,\perp_1)\!]} \oplus C_{[\![-\infty,\,\infty)\!]}{}^{[\![\perp_1,\,\perp_2)\!]} \oplus C_{[\![-\infty,\,\infty)\!]}{}^{[\![\perp_2,\,\infty)\!]} \right) \\
&= \mathcal{R}_+\left(C_{[\![-\infty,\,\infty)\!]}{}^{[\![-\infty,\,\infty)\!]} \right) \\
&= \int_{[\![-\infty,\,\infty)\!]} f(\tau)g(t-\tau)\, d\tau
\end{aligned}
$$

```
Convolve := proc(f,af,bf,g,ag,bg)
 local afbg,bfag,out,temp;
 if(af+bg != undefined) then afbg := af+bg; fi;
 if(bf+ag != undefined) then bfag := bf+ag; fi;
 out:=(int(fg(x),x=af..t-ag))*(OrientedInterval(af+ag,bfag))(t)
    +(int(fg(x),x=af..bf))*(OrientedInterval(bfag,afbg))(t)
    +(int(fg(x),x=t-bg..bf))*(OrientedInterval(afbg,bf+bg))(t);
 out:=subs(t-infinity=-infinity,out);
 out:=subs(t+infinity=infinity,out);
 try temp := convert(out,piecewise,t);
 catch:
    try temp:=convert(convert(out,Heaviside),piecewise,t);
    catch: temp := out; end try;
 finally out := temp; end try;
 try temp := Combine(subs(infinity=infty,out));
    catch temp:=out;
 finally out:=subs(infty=infinity,temp); end try;
 out := subs(fg(x)=f(x)*g(t-x), out);
end proc;
```

```
> Convolve(sin, 0, Pi, t->exp(-t), 0, 1);
```

$$\begin{cases} 0 & t\sim < 0 \\ \int_{t\sim}^{0} \sin(x) e^{-t\sim+x}\, dx & t\sim < 1 \\ \int_{-1+t\sim}^{t\sim} \sin(x) e^{-t\sim+x}\, dx & t\sim < \pi \\ \int_{-1+t\sim}^{t\sim} \sin(x) e^{-t\sim+x}\, dx & t\sim < \pi + 1 \\ 0 & \pi + 1 \leq t\sim \end{cases}$$

Case 1:

```
> assume(t::real); additionally(af<bf); additionally(ag<bg);
> Convolve(f,af,bf,g,ag,infinity);
```

$$\begin{cases} 0 & t\sim < af\sim + ag\sim \\ \int_{af\sim}^{t\sim - ag\sim} f(x) g(t\sim - x)\, dx & t\sim < bf\sim + ag\sim \\ \int_{af\sim}^{bf\sim} f(x) g(t\sim - x)\, dx & bf\sim + ag\sim \leq t\sim \end{cases}$$

Curriculum Vitae

Name:	Mike Ghesquiere
Post-Secondary Education and Degrees:	University of Western Ontario 2013 – 2015 M.Sc. (Computer Science) University of Western Ontario 2008 – 2013 B.Sc. Honours (Computer Science)
Honours and Awards:	Undergraduate Summer Research Award (Computer Science) 2012 Undergraduate Summer Research Award (Mathematics) 2011
Related Work Experience:	hyperPad 2015 – Software Developer The University of Western Ontario 2013 – 2015 Teaching Assistant Cyborg Trading Systems 2013 Software Developer