Integration using hybrid domains

Mike Ghesquiere
Department of Computer Science
University of Western Ontario
London, Canada N6A 5B7
mghesqui@uwo.ca

Stephen M. Watt
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Canada N2L 3G1
smwatt@uwaterloo.ca

ABSTRACT

The abstract is about 5 lines.
The abstract is about 5 lines.
The abstract is about 5 lines.
The abstract is about 5 lines.
The abstract is about 5 lines.

1. INTRODUCTION

The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.
The introduction might take up the rest of the column.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

2. 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:

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

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, $H = \{\!\!\{\ a^1, a^1, b^{-2}, a^3, b^1\ \}\!\!\} = \{\!\!\{\ a^5, b^{-1}\ \}\!\!\}$. 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 [**?**, **?**, **?**, **?**] 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 $x \in U$:

$$(A \oplus B)(x) = A(x) + B(x) \tag{1}$$
$$(A \ominus B)(x) = A(x) - B(x) \tag{2}$$
$$(A \otimes B)(x) = A(x) \cdot B(x) \tag{3}$$

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

$$(cA)(x) = c \cdot A(x) \tag{4}$$

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.

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:

$$H = P_1 \oplus P_2 \oplus \ldots \oplus P_n \tag{5}$$

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. But this does not hold for a non-strict partition of a reducible hybrid set. Consider the interval $[0,1]$ which can be partitioned by $\{[0,2],\ \ominus(1,2]\}$.

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 $\{a_{ij}\}_{j\in J_i}$

$$P_i = \bigoplus_{j\in J_i} a_{ij} Q_j \tag{6}$$

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 $\operatorname{supp}(Q) = \operatorname{supp}(P)$.

2.1 Oriented Intervals

Definition Given a totally ordered set $(X, \leq)$, 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:

$$\begin{aligned} [a,b]_X &= \{x \in X \mid a \leq x \leq b\} \\ [a,b)_X &= \{x \in X \mid a \leq x < b\} \\ (a,b]_X &= \{x \in X \mid a < x \leq b\} \\ (a,b)_X &= \{x \in X \mid a < x < b\} \end{aligned} \tag{7}$$

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 $[a,b) \oplus [b,c) = [a,b)$.

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

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:

$$\begin{aligned} [\![a,b)\!) &= [a,b) \ominus [b,a) \\ (\!(a,b]\!] &= (a,b] \ominus (b,a] \\ [\![a,b]\!] &= [a,b] \ominus (b,a) \\ (\!(a,b)\!) &= (a,b) \ominus [b,a] \end{aligned} \tag{8}$$

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 2.1. *For all a, b, c,*

$$\begin{aligned} [\![a,b)\!) &= \ominus(\!(b,a]\!] \\ (\!(a,b]\!] &= \ominus[\![b,a)\!) \\ [\![a,b]\!] &= \ominus(\!(b,a)\!) \\ (\!(a,b)\!) &= \ominus[\![b,a]\!] \end{aligned} \tag{9}$$

$$[\![a,b)\!) \oplus [\![b,c)\!) = [\![a,c)\!) \tag{10}$$

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. 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. 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 can think of a hybrid function as a mapping between two Boolean sets decorated with a integer weight on each "arrow". 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:

$$f^H := \bigoplus_{x\in B} H(x)\{\!\!\{ \, (x, f(x))^1 \, \}\!\!\} \tag{11}$$

There is one problem with this definition. Given a hybrid function f^H and a generalized partition $H = P_1 \oplus \ldots \oplus P_n$, we are interested in f^{P_i}. For traditional functions and partitions, this is not an issue. We can be sure that supp$P_i \subseteq H$ however this does not hold for generalized partions and hybrid functions. To remedy this we will 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 $\widetilde{f}^{\,H}$ as:

$$\widetilde{f}^{\,H} = \bigoplus_{x\in U} H(x)\{\!\!\{ \, (x, f)^1 \, \}\!\!\} \tag{12}$$

One should notice the similarity between (12) and (11), however here 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.

There is little established use for hybrid functions; so our primary concern is in mapping results back to 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:

$$\mathcal{R}(f^H)(x) = f|_{\mathcal{R}(H)}(x) \tag{13}$$

3.1 Join

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.

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 3.1. *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,*

$$f^A \oplus f^B = f^{A\oplus B} \tag{14}$$

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 $f^A \oplus f^B = f^{A\oplus B}$. $\square$

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 3.2. *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:*

$$f^A \oplus g^B = (f|_{suppA} \oplus g|_{suppB})^{A\oplus B} \tag{15}$$

Notice here the use of $\oplus$ on the right hand side. Here $\oplus$ is the traditional (non-hybrid) function join as defined in (**??**). $(f \oplus g)$ was undefined for $(\mathrm{supp}(A) \cap \mathrm{supp}(B))$ 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 (**??**) 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 $(f|_{\mathrm{supp}A} \oplus g|_{\mathrm{supp}B})$ without undefined points as supp$A \cap$ supp$B = \emptyset$. For $(x, f(x)) \in^m (f^A \oplus g^B)$ with non-zero m, we have $(f|_{\mathrm{supp}A} \oplus g|_{\mathrm{supp}B})(x) = f(x)$ 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 $(f|_{\mathrm{supp}A} \oplus g|_{\mathrm{supp}B})(x) = g(x)$ and $x \in^n A \oplus B$. Thus, $f^A \oplus g^B = (f|_{\mathrm{supp}A} \oplus g|_{\mathrm{supp}B})^{A\oplus B}$. $\square$

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 3.3. *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**.*

3.2 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,

$$x *^n y = (((x \overbrace{* \; y) * y) * \ldots * y)}^{n \text{ times}} \tag{16}$$

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:

$$\mathcal{R}_*(f^A)(x) = \left(*^{A_1(x)} f_1(x) *^{A_2(x)} f_2(x) * \ldots\right)\Big|_{\mathrm{supp}A} \tag{17}$$

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

$$\mathcal{R}(f^A) = \mathcal{R}_*(f^A) \tag{18}$$

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:

$$\mathcal{R}_*(\mathcal{R}_*(f^F) \oplus \mathcal{R}_*(g^G)) = \mathcal{R}_*(f^F \oplus g^G) \tag{19}$$

3.3 Piece-wise Functions

...
...
...
...
...
...
...
...
...
...
...
...
...

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:

$$\begin{aligned} && U &= \bigoplus_j C_j \\ \forall i \in 1 \ldots n: && A_i &= \bigoplus_j a_{i,j} C_j \\ \forall i \in 1 \ldots m: && B_i &= \bigoplus_j b_{i,j} C_j \end{aligned}$$

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

$$C = \Big\{ \; R, \; A_1, \; A_2, \; \ldots, \; A_{n-1}, \; B_1, \; B_2, \ldots, \; B_{m-1} \; \Big\}$$

where $R = U \ominus A_1 \ominus \ldots \ominus A_{n-1} \ominus B_1 \ominus \ldots \ominus B_{n-1}$.

Finally, to generalize the example from **??**, 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

$$\begin{aligned} f * g = \; & \mathcal{R}_*\big((f_1 * g_m)^{A_1} \oplus \ldots \oplus (f_{n-1} * g_m)^{A_{n-1}} \\ & \oplus (f_n * g_1)^{B_1} \oplus \ldots \oplus (f_n * g_{m-1})^{B_{m-1}} \\ & \oplus (f_n * g_m)^{U \ominus (A_1 \oplus \ldots \oplus A_{n-1} \oplus B_1 \oplus \ldots \oplus B_{m-1})} \big) \end{aligned} \tag{20}$$

4. INTEGRATION

4.1 Boundary Operator

In one dimension, the boundary of an interval was quite straightforward. In the case of 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 $\llbracket \boldsymbol{a}, \boldsymbol{b} \rrbracket$ 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** $\llbracket \boldsymbol{a}, \boldsymbol{b} \rrbracket$, denoted by the operator ∂ is given by:

$$\begin{aligned} \partial_k \llbracket \boldsymbol{a}, \boldsymbol{b} \rrbracket = \bigoplus_{j=1}^{k} (-1)^j \Bigg(& \Big\llbracket \boldsymbol{a}^{\llbracket 1,n \rrbracket_{\mathbb{N}}}, \boldsymbol{b}^{\llbracket 1,n \rrbracket_{\mathbb{N}} \ominus i_j} \oplus \boldsymbol{a}^{i_j} \Big\rrbracket_{\mathbb{R}^n} \\ & \ominus \Big\llbracket \boldsymbol{a}^{\llbracket 1,n \rrbracket_{\mathbb{N}} \ominus i_j} \oplus \boldsymbol{b}^{i_j}, \boldsymbol{b}^{\llbracket 1,n \rrbracket_{\mathbb{N}}} \Big\rrbracket_{\mathbb{R}^n} \Bigg) \end{aligned} \tag{21}$$

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 $\llbracket 1, i_j \rrparenthesis_{\mathbb{N}}$ and $\llparenthesis i_j, n \rrbracket_{\mathbb{N}}$

are intervals over vector indices just as in Chapter 3. Thus, the term $\boldsymbol{a}^{\llbracket 1, i_j \rrparenthesis_{\mathbb{N}}}$ refers to the vector $(a_1, a_2, \ldots, a_{i_j - 1})$ while the term $\boldsymbol{b}^{\llparenthesis i_j, n \rrbracket_{\mathbb{N}}}$ 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: $\llbracket 1, i_j \rrparenthesis$, $\Lbrace\ i_j\ \Rbrace$, and $\llparenthesis i_j, n \rrbracket$. Formally, we are actually using the hybrid sets $\Lbrace\ (i_j)^1\ \Rbrace$ but we omit multiplicity of one.

Next, using the pointwise sum $\oplus$ we can reconstruct n-dimensional vectors from our pieces. We can construct a $(k-1)$-rectangle using these vectors as shown earlier. 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: $\llbracket a_{i_j}, a_{i_j} \rrbracket$ and $\llbracket b_{i_j}, b_{i_j} \rrbracket$ 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 $\llbracket \boldsymbol{a}, \boldsymbol{b} \rrbracket$. Hence we are indeed left with a $(k-1)$-rectangle.

4.2 Example: *Boundary of a 1-rectangle*

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

$$\begin{aligned}
\partial(\ \llbracket \boldsymbol{a}, \boldsymbol{b} \rrbracket\ ) = (-1)^i (&\llbracket \boldsymbol{a}^{\llbracket 1,1 \rrparenthesis}, \boldsymbol{b}^{\llbracket 1,1 \rrparenthesis} \rrbracket \times \Lbrace\ a_1\ \Rbrace \times \llbracket \boldsymbol{a}^{\llparenthesis 1,1 \rrbracket}, \boldsymbol{b}^{\llparenthesis 1,1 \rrbracket} \rrbracket \\
&\ominus \llbracket \boldsymbol{a}^{\llbracket 1,1 \rrparenthesis}, \boldsymbol{b}^{\llbracket 1,1 \rrparenthesis} \rrbracket \times \Lbrace\ b_1\ \Rbrace \times \llbracket \boldsymbol{a}^{\llparenthesis 1,1 \rrbracket}, \boldsymbol{b}^{\llparenthesis 1,1 \rrbracket} \rrbracket) \\
=\ &\ominus \llbracket \boldsymbol{a}^{\emptyset}, \boldsymbol{b}^{\emptyset} \rrbracket \times \Lbrace\ a_1\ \Rbrace \times \llbracket \boldsymbol{a}^{\emptyset}, \boldsymbol{b}^{\emptyset} \rrbracket \\
&\oplus \llbracket \boldsymbol{a}^{\emptyset}, \boldsymbol{b}^{\emptyset} \rrbracket \times \Lbrace\ b_1\ \Rbrace \times \llbracket \boldsymbol{a}^{\emptyset}, \boldsymbol{b}^{\emptyset} \rrbracket \\
=\ &\Lbrace\ a^{-1}, b^1\ \Rbrace
\end{aligned}$$

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 $\int_{\llbracket a,b \rrbracket} F'(x)\ \mathrm{d}x = \sum(\partial(\llbracket a, b \rrbracket))$. 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 $\mathbb{R}^n$.

4.3 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 just:

$$\int_c f = \sum_i \lambda_i \int_{c_i} f \tag{22}$$

and the boundary operator ∂_k defined as:

$$\partial_k(c) = \bigoplus_{i=1}^{k} \lambda_i \partial_k(c_i)$$

now maps from k-chains to $(k-1)$-chains:

$$\begin{tikzcd} \ldots & C_{k-1} \arrow[l, "\partial_{k-1}"'] & C_k \arrow[l, "\partial_k"'] & C_{k+1} \arrow[l, "\partial_{k+1}"'] & \ldots \arrow[l, "\partial_{k+2}"'] \end{tikzcd}$$

One obvious question that arises, is what is the result of chaining the boundary operator with itself. That is, what is $\partial(\partial(c))$. If we consider $\llbracket \boldsymbol{a}, \boldsymbol{b} \rrbracket$ be a k-rectangle in $\mathbb{R}^n$. Then we have:

$$\begin{aligned}
\partial\partial\left(\llbracket \boldsymbol{a}, \boldsymbol{b} \rrbracket\right) = \bigoplus_{j=1}^{k} (-1)^j \Big(\ &\partial_{k-1}\left(\llbracket \boldsymbol{a}^{\llbracket 1,n \rrbracket},\ \boldsymbol{b}^{\llbracket 1,n \rrbracket \ominus i_j} \oplus \boldsymbol{a}^{i_j} \rrbracket\right) \\
&\ominus\ \partial_{k-1}\left(\llbracket \boldsymbol{a}^{\llbracket 1,n \rrbracket \ominus i_j} \oplus \boldsymbol{b}^{i_j},\ \boldsymbol{b}^{\llbracket 1,n \rrbracket} \rrbracket\right)\Big)
\end{aligned}$$

$$\begin{aligned}
= \bigoplus_{j=1}^{k} \bigoplus_{\ell=1}^{k-1} (-1)^{j+\ell}\ &\llbracket \boldsymbol{a}^{\llbracket 1,n \rrbracket},\ \boldsymbol{b}^{\llbracket 1,n \rrbracket \ominus i_j \ominus i_{\neq j,\ell}} \oplus \boldsymbol{a}^{i_j \oplus i_{\neq j,\ell}} \rrbracket \\
&\ominus \llbracket \boldsymbol{a}^{\llbracket 1,n \rrbracket \ominus i_{j,\ell}} \oplus \boldsymbol{b}^{i_{\neq j,\ell}},\ \boldsymbol{b}^{\llbracket 1,n \rrbracket \ominus i_j} \oplus \boldsymbol{a}^{i_j} \rrbracket \\
&\ominus \llbracket \boldsymbol{a}^{\llbracket 1,n \rrbracket \ominus i_j} \oplus \boldsymbol{b}^{i_j},\ \boldsymbol{b}^{\llbracket 1,n \rrbracket \ominus i_{\neq j,\ell}} \oplus \boldsymbol{a}^{i_{\neq j,\ell}} \rrbracket \\
&\oplus \llbracket \boldsymbol{a}^{\llbracket 1,n \rrbracket \ominus i_j \ominus i_{\neq j,\ell}} \oplus \boldsymbol{b}^{i_j \oplus i_{\neq j,\ell}},\ \boldsymbol{b}^{\llbracket 1,n \rrbracket} \rrbracket
\end{aligned}$$

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 sequences $i_{\neq j}$, $\{i_{\neq j,\ell}\}_{\ell=1}^{k-1}$ given by:

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

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:

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

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.

4.4 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:

$$\begin{tikzcd} \ldots & C_{k-1} \arrow[l, "\partial"'] & C_k \arrow[l, "\partial"'] & C_{k+1} \arrow[l, "\partial"'] & \ldots \arrow[l, "\partial"'] \end{tikzcd}$$

$$\begin{tikzcd} \ldots \arrow[r, "\mathrm{d}"] & \Lambda^{k-1} \arrow[r, "\mathrm{d}"] & \Lambda^{k} \arrow[r, "\mathrm{d}"] & \Lambda^{k+1} \arrow[r, "\mathrm{d}"] & \ldots \end{tikzcd}$$

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:

$$\int_{\partial M} \omega = \int_M d\omega \tag{Stokes' Theorem}$$

...

Transition into proof

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:

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

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:

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

Since this is an integral over the canonical basis $\mathrm{x} = (x_1, \ldots, x_k)$ we can remove the wedge products and integrate with respect to x_i using Fubini's Theorem:

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

where $\mathrm{d}\widehat{x}_i = \mathrm{d}(x_i, \ldots, \widehat{x}_i, \ldots, x_k)$.

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:

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

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.

Next, we extend this result from the standard cube I^k to k-dimensional singular cubes c:

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

and for a chains $C = a_1 c_1 + \ldots + a_n c_n$ 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. $\square$

...
...
...
...
...
...

5. CONVOLUTION

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 1.

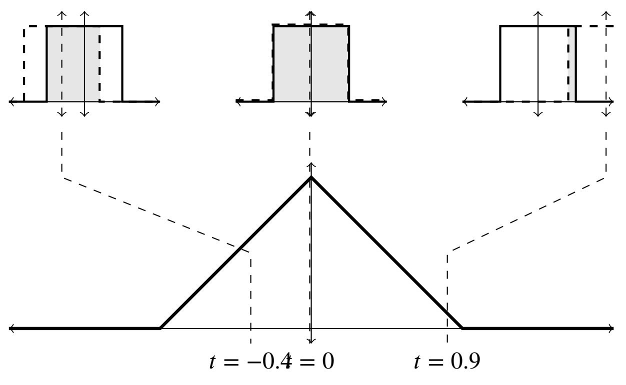

Figure 1: The convolution of the box signal $f(t) = g(t) = \left(0^{(\!(-\infty,-0.5)\!)} \oplus 1^{[\![-0.5,0.5]\!]} \oplus 0^{(\!(0.5,\infty)\!)}\right)$ with itself.

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

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

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

$$(F * G)(t) = \int_{-\infty}^{\infty} F(\tau)\, G(t-\tau)\,\mathrm{d}\tau \tag{23}$$

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.

5.1 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. Rather, we are interested in *Symbolic Linear Convolution* (of piecewise continuous functions).

The typical approach is to first consider for convolution of "one piece" functions [**?**, **?**]. By "one-piece" functions F and G of the form:

$$F(x) = \mathcal{R}_+\left(f^{[a_f,b_f)}\right)(x) = \begin{cases} f(x) & a_f \le x < b_f \\ 0 & \text{otherwise} \end{cases} \tag{24}$$

$$G(x) = \mathcal{R}_+\left(g^{[a_g,b_g)}\right)(x) = \begin{cases} g(x) & a_g \le x < b_g \\ 0 & \text{otherwise} \end{cases} \tag{25}$$

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 $b_f - a_f \le b_g - a_g$. 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 $G * F$.

$$\begin{aligned}(F * G)(t) &= \int_{-\infty}^{\infty} F(\tau)G(t-\tau)\,\mathrm{d}\tau \\ &= \int_{\infty}^{-\infty} F(t-\tau')G(\tau')\,(-1)\,\mathrm{d}\tau' \qquad (\tau' = t-\tau) \\ &= \int_{-\infty}^{\infty} G(\tau')F(t-\tau')\,\mathrm{d}\tau' \\ &= (G * F)(t)\end{aligned} \tag{26}$$

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:

$$(F * G)(t) = \begin{cases} \int_{a_f}^{t-a_g} f(\tau)\, g(t-\tau)\,\mathrm{d}\tau & t \in A \\ \int_{a_f}^{b_f} f(\tau)\, g(t-\tau)\,\mathrm{d}\tau & t \in B \\ \int_{t-b_g}^{b_f} f(\tau)\, g(t-\tau)\,\mathrm{d}\tau & t \in C \\ 0 & \text{otherwise} \end{cases} \tag{27}$$

with the intervals A, B, C given as follows:

$$\begin{aligned} A &= [a_f + a_g,\ b_f + a_g) \\ B &= [b_f + a_g,\ a_f + b_g) \\ C &= [a_f + b_g,\ b_f + b_g) \end{aligned}$$

These regions can be visualized as b in Figure **??** 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 1). Another formulation presented by Cîrnu [**?**] and Cavicchi [**?**]

$$(F * G)(t) = \int_{\max(a_f,\ t-b_g)}^{\min(b_f, t-a_g)} f(\tau)\cdot g(t-\tau)\,\mathrm{d}\tau \tag{28}$$

whenever

$$(a_f + a_g) \le t < (b_f + b_g)$$

and 0 elsewhere. 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:

$$\left(\left(\sum_i f_i^{P_i}\right) * \left(\sum_j g_j^{Q_j}\right)\right)(t) = \sum_i \sum_j \left(f_i^{P_i} * g_j^{P_j}\right) \tag{29}$$

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 [**?**]. 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 5.2. Concerns with intervals where one boundary point is at infinity have also been raised [**?**]. Techniques to handle this will be investigated in Section 5.3.

5.2 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** $\boldsymbol{f^{[a_f,b_f)}}$ **and** $\boldsymbol{g^{[a_g,b_g)}}$ is:

$$\begin{aligned}
&(f^{[a_f,b_f)} \;*\; g^{[a_g,b_g)})(t) = \\
&\quad \mathcal{R}_+\Bigg(\left(\int_{[\![a_f,\, t-a_g]\!)} f(\tau)\, g(t-\tau)\,\mathrm{d}\tau\right)^{[\![a_f+a_g,\, b_f+a_g]\!)} \\
&\qquad \oplus \left(\int_{[\![a_f,\, b_f]\!)} f(\tau)\, g(t-\tau)\,\mathrm{d}\tau\right)^{[\![b_f+a_g,\, a_f+b_g]\!)} \\
&\qquad \oplus \left(\int_{[\![t-b_g,\, b_f]\!)} f(\tau)\, g(t-\tau)\,\mathrm{d}\tau\right)^{[\![a_f+b_g,\, b_f+b_g]\!)} \Bigg)(t) \qquad (30)
\end{aligned}$$

The first thing one should note is the similarity between this expression and (27). But, *we do not enforce* $b_f - a_f \leq b_g - a_g$ as we did in Section 5.1. 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. The ordering of f and g is no longer dictated by the relative length of their respective intervals.

As an example, suppose we wish to evaluate the convolution at a point t where $(a_f + b_g) \leq t < (b_f + a_g)$:

$$\begin{aligned}
[\![a_f + a_g,\; b_f + a_g]\!)(t) &= 1 \\
[\![b_f + a_g,\; a_f + b_g]\!)(t) &= -1 \\
[\![a_f + b_g,\; b_f + b_g]\!)(t) &= 1
\end{aligned}$$

Simplifying the $+$-reduction we then get:

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

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:

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

5.3 Infinite Intervals

Evans and McClellan [**?**] 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 end-points 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 (30) 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 3. 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:

$$\begin{aligned}
&(f^{[a_f,\infty)} \;*\; g^{[-\infty,b_g)})(t) = \\
&\quad \mathcal{R}_+\Bigg(\left(\int_{[\![a_f,\, \infty]\!)} f(\tau)\, g(t-\tau)\,\mathrm{d}\tau\right)^{[\![-\infty,\, a_f+b_g]\!)} \\
&\qquad \oplus \left(\int_{[\![t-b_g,\, \infty]\!)} f(\tau)\, g(t-\tau)\,\mathrm{d}\tau\right)^{[\![a_f+b_g,\, \infty]\!)} \Bigg)(t) \qquad (31)
\end{aligned}$$

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 (30) without modification is correct in all cases. A full enumeration of this can be found in Appendix **??**.