Mapped Regular Pavings*

Jennifer Harlow
University of Canterbury, Christchurch, New Zealand

Raazesh Sainudiin†
University of Canterbury, Christchurch, New Zealand

Warwick Tucker
Uppsala University, Uppsala, Sweden

Abstract

A regular paving is a finite succession of bisections that partition a root box $\boldsymbol{x}$ in $\mathbb{R}^d$ into sub-boxes using a binary tree-based data structure. We extend regular pavings to mapped regular pavings which map sub-boxes in a regular paving of $\boldsymbol{x}$ to elements in some set $\mathbb{Y}$. Arithmetic operations defined on $\mathbb{Y}$ can be extended point-wise over $\boldsymbol{x}$ and carried out in a computationally efficient manner using $\mathbb{Y}$-mapped regular pavings of $\boldsymbol{x}$. The efficiency of this arithmetic is due to recursive algorithms on the algebraic structure of finite rooted binary trees that are closed under union operations. Our arithmetic has many applications in function approximation using tree based inclusion algebras and statistical set-processing.

Keywords: finite rooted binary trees, tree arithmetic, inclusion algebra

AMS subject classifications: 65G40,65Y04,05C05,20F65

1 Introduction

Hierarchical data structures, such as trees, are commonly used to represent and organise multi-dimensional data. A wide range of such data structures has been developed including, for example, binary search trees, quadtrees and octrees, k-d trees and other binary space partitioning trees. These specialisations have been developed to suit particular types of data and to meet the needs of different applications or uses of that data — search and retrieval, range queries, the relationships between 3-d objects for computer graphics, etc [11].

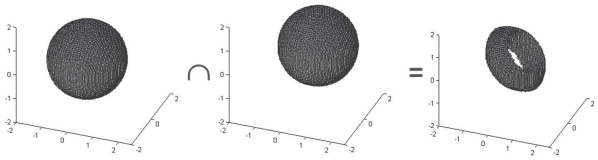
*Submitted: May 16, 2012; Revised: September 5, 2012; Accepted: November 12, 2012.

†Correspond to: Raazesh Sainudiin, Laboratory for Mathematical Statistical Experiments & Department of Mathematics and Statistics, Private Bag 4800, University of Canterbury, Christchurch 8041, New Zealand. Email addresses: raazesh.sainudiin@gmail.com, r.sainudiin@math.canterbury.ac.nz

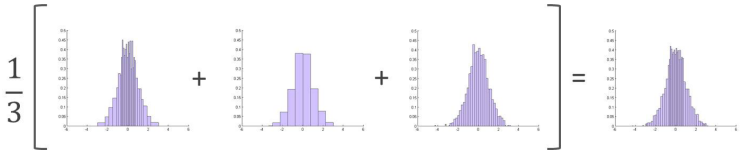
A regular paving [10, 3, 2] is a finite succession of bisections that partition a box $\boldsymbol{x}$ in $\mathbb{R}^d$ into sub-boxes using a tree-based data structure. In this article we present mapped regular pavings, an extension of regular pavings designed to facilitate arithmetical operations on the data structure itself. In a mapped regular paving the sub-boxes in a regular paving of a box $\boldsymbol{x}$ are mapped to elements in some set $\mathbb{Y}$ so that arithmetic operations defined on $\mathbb{Y}$ can be extended point-wise to $\mathbb{Y}$-mapped regular pavings of $\boldsymbol{x}$, and the paving itself is constructed so that these operations can be carried out in a computationally efficient manner.

Figure 1 illustrates some of the many operations that can be carried out using regular mapped pavings.

(a) Arithmetic with coloured spaces.



(b) Intersection of enclosures of two hollow spheres.



(c) Histogram averaging.

Figure 1: Examples of operations with various mapped regular pavings.

In this article we show how the mapped regular pavings allow any arithmetic defined over elements in a general set $\mathbb{Y}$ to be extended to $\mathbb{Y}$-mapped regular pavings. We demonstrate the very wide applicability of this concept by extending and explaining the examples shown in Figure 1.

An interesting set of examples involves the use of mapped regular pavings to represent functions. In this article we explore this in detail, including:

- Arithmetic on piecewise constant functions and interval-valued functions;
- Exploiting the tree-based structure to calculate interval enclosures of real-valued functions efficiently, and hence to find piecewise constant approximations for these functions within required levels of tolerance;

- Obtaining the marginal of a piecewise constant function on a multi-dimensional domain by integrating along any subset of its coordinates, and also for obtaining the conditional function by fixing the values in the domain on a subset of coordinates, and for producing the highest coverage regions of the function domain.

Mapped regular pavings can also be used to extend existing methods of organising and analysing data. An example of this is a computationally efficient representation, endowed with arithmetic for pattern-recognition, of radar-observed flight co-trajectories over a busy airport [14]. In this article we briefly describe how mapped regular pavings can be used in multi-dimensional nonparametric density estimation problems by representing histograms as mapped regular pavings and then applying basic arithmetical operations in order to obtain average histograms as well being able to use the marginalisation and conditional function operations described above.

This article is organized as follows: In Section 2 we give a brief overview of tree-based data structures and their advantages for representing multi-dimensional objects and organising multi-dimensional data. Section 3 describes regular pavings. Section 4 describes mapped regular pavings and operations over them; various algorithms are introduced with illustrative examples. We conclude in Section 5.

2 Tree Structures

Partitions of multi-dimensional space are usually represented using hierarchical data structures such as trees. The main advantages are:

- Operations on the data structures are often well suited to spatial divide and conquer methods and hence to hierarchical data structures.
- A tree provides $\mathcal{O}(\log m)$ access time to any sub-box in a collection of m sub-boxes, regardless of the number of dimensions, without the need to impose a uniform grid partition on the space.
- Trees provide low-cost (constant time) insertion and deletion of sub-boxes, without the need to reallocate existing partitions in memory.
- Algorithms operating on trees can be expressed naturally and succinctly in a recursive form, allowing a simpler and more understandable programming implementation [4].

A review of the use of trees to represent spatial data structures that discusses most of these points, and more, can be found in [12].

A tree-based structure is particularly suitable for mapped regular pavings because an arithmetic operation on two pavings (for example, addition) requires finding 'matching' of pairs of sub-boxes, each pair having one sub-box from each operand paving. If a tree structure is used then the algorithm can easily be expressed and implemented very efficiently in a form that recurses simultaneously on both structures and automatically matches the appropriate boxes.

For these reasons we only discuss mapped regular pavings implemented using a binary tree, and the algorithms given assume the use of such a structure. Other underlying structures for implementing the pavings are possible but we are not aware of any alternative that offers a better combination of simplicity and computational efficiency than a tree.

3 Regular Pavings

Let $\boldsymbol{x} := [\underline{x}, \overline{x}]$ be a compact real interval with lower bound $\underline{x}$ and upper bound $\overline{x}$ where $\underline{x} \leq \overline{x}$. Let the space of such intervals be $\mathbb{IR}$. We can define the width, midpoint and radius of an interval $\boldsymbol{x}$ as $\operatorname{wid}(\boldsymbol{x}) := \overline{x} - \underline{x}$, $\operatorname{mid}(\boldsymbol{x}) := \frac{\underline{x}+\overline{x}}{2}$ and $\operatorname{rad}(\boldsymbol{x}) := \frac{\overline{x}-\underline{x}}{2}$, respectively. We can also define a box of dimension d with coordinates in $\Delta := \{1, 2, \ldots, d\}$ as an interval vector

$$\boldsymbol{x} := [\underline{x}_1, \overline{x}_1] \times \ldots \times [\underline{x}_d, \overline{x}_d] =: \bigboxtimes_{j \in \Delta} [\underline{x}_j, \overline{x}_j] \ .$$

Let $\mathbb{IR}^d$ be the set of all such boxes. Consider a box $\boldsymbol{x}$ in $\mathbb{IR}^d$. Let the index ι be the first coordinate of maximum width, i.e.

$$\iota = \min \left(\operatorname*{argmax}_i (\operatorname{wid}(\boldsymbol{x}_i)) \right) \ .$$

A *bisection* or *split* of $\boldsymbol{x}$ perpendicularly at the mid-point along this first widest coordinate ι gives us the left and right child boxes of $\boldsymbol{x}$ as follows:

$$\boldsymbol{x}_{\mathsf{L}} := [\underline{x}_1, \overline{x}_1] \times \ldots \times [\underline{x}_\iota, \operatorname{mid}(\boldsymbol{x}_\iota)) \times [\underline{x}_{\iota+1}, \overline{x}_{\iota+1}] \times \ldots \times [\underline{x}_d, \overline{x}_d] \ ,$$

$$\boldsymbol{x}_{\mathsf{R}} := [\underline{x}_1, \overline{x}_1] \times \ldots \times [\operatorname{mid}(\boldsymbol{x}_\iota), \overline{x}_\iota] \times [\underline{x}_{\iota+1}, \overline{x}_{\iota+1}] \times \ldots \times [\underline{x}_d, \overline{x}_d] \ .$$

Such a bisection is said to be *regular*.

Note that this bisection gives the right child box a half-open interval $[\underline{x}_\iota, \operatorname{mid}(\boldsymbol{x}_\iota))$ on coordinate ι so that the intersection of the left and right child boxes is empty.

This refinement is a necessary condition for some of the operations described in Section 4 and causes no complications for those operations that do not require it. If $\boldsymbol{x}$ is not a closed box in $\mathbb{IR}^d$ then we can make it closed again, if necessary, by taking its interval or box hull $\Box\boldsymbol{x}$. For example, we can bisect a half-open interval $\boldsymbol{x}$ with the mid-point of its interval hull given by $\operatorname{mid}(\Box\boldsymbol{x})$.

A recursive sequence of selective regular bisections of boxes, with possibly open boundaries, along the first widest coordinate, starting from the root box $\boldsymbol{x}$ in $\mathbb{IR}^d$ is known as a *regular paving (RP)* [3, 2] or n-tree [10] of $\boldsymbol{x}$.

An RP of $\boldsymbol{x}$ can also be seen as a binary tree formed by recursively bisecting the box $\boldsymbol{x}$ at the root node. Each node in the binary tree has either no children or two children. These trees are known as *plane binary trees* in enumerative combinatorics [13, Ex. 6.19(d), p. 220] and as *finite, rooted binary trees (frb-trees)* in geometric group theory [5, Ch. 10]. When the root box $\boldsymbol{x}$ is clear from the context we refer to an RP of $\boldsymbol{x}$ as merely an RP. Each node of an RP is associated with a sub-box of the root box that can be attained by a sequence of selective regular bisections.

Each node in an RP is distinctly labeled by the sequence of child node selections from the root node. We label these nodes and the associated boxes with strings composed of L and R for left and right, respectively.

We illustrate the relationship of trees, labels and partitions in Figure 2 with a simple 1-dimensional example. The root node associated with a 1-dimensional root interval $\boldsymbol{x}_\rho$ is labeled ρ. First, we split ρ into two child nodes and denote it by $\bigtriangledown(\rho) = \{\rho\mathsf{L}, \rho\mathsf{R}\}$. These left child and right child nodes are labeled by $\rho\mathsf{L}$ and $\rho\mathsf{R}$, respectively. The left half of $\boldsymbol{x}_\rho$ that is now associated with node $\rho\mathsf{L}$ is denoted by $\boldsymbol{x}_{\rho\mathsf{L}}$. Similarly, the right half of $\boldsymbol{x}_\rho$ that is associated with the right child node $\rho\mathsf{R}$ is denoted by $\boldsymbol{x}_{\rho\mathsf{R}}$. We say $\rho\mathsf{L}$ and $\rho\mathsf{R}$ are a pair of *sibling nodes* since they share the same parent node ρ. A node with no child nodes is called a *leaf node*. A *cherry node* is a

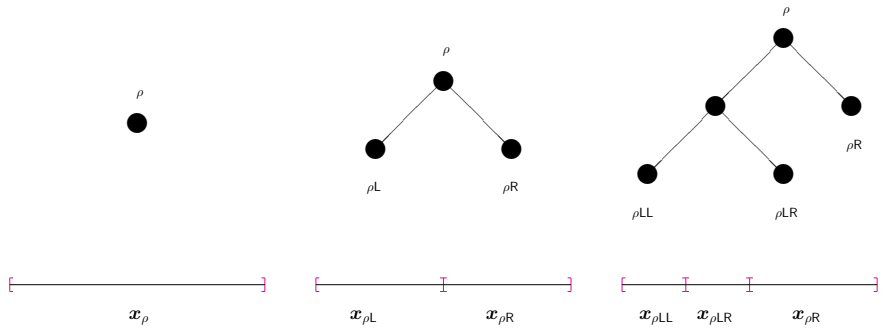


Figure 2: A sequence of selective bisections of boxes (nodes) along the first widest coordinate, starting from the root box (root node), produces an RP.

sub-terminal node with a pair of child nodes that are both leaves. This pair of sibling nodes can be *reunited* or *merged* to its parent cherry node ρ, thereby turning the cherry node into a leaf node. Such a merging operation is denoted by $\bigwedge(\rho\mathsf{L}, \rho\mathsf{R}) = \rho$.

Returning to Figure 2, let us further split the left node $\rho\mathsf{L}$ to get its left and right child nodes $\rho\mathsf{LL}$ and $\rho\mathsf{LR}$ with associated sub-intervals $\boldsymbol{x}_{\rho\mathsf{LL}}$ and $\boldsymbol{x}_{\rho\mathsf{LR}}$ respectively, formed by the bisection of interval $\boldsymbol{x}_{\rho\mathsf{L}}$. Because the root interval $\boldsymbol{x}_\rho$ is 1-dimensional, each bisection is always on that single coordinate.

Figure 3 shows a sequence of bisections of a square (2-dimensional) root box. We start with the same sequence as in Figure 2, so the first three trees are identical to those in Figure 2 but in Figure 3 we can see the effect of always bisecting on the first widest coordinate. The first bisection, forming sub-boxes $\boldsymbol{x}_{\rho\mathsf{L}}$ and $\boldsymbol{x}_{\rho\mathsf{R}}$, takes place on the first widest coordinate of $\boldsymbol{x}_\rho$, which is the first coordinate because the box is square. The next bisection, of box $\boldsymbol{x}_{\rho\mathsf{L}}$ to form $\boldsymbol{x}_{\rho\mathsf{LL}}$ and $\boldsymbol{x}_{\rho\mathsf{LR}}$, takes place on the second coordinate because this is the first widest coordinate of $\boldsymbol{x}_{\rho\mathsf{L}}$.

We then extend the sequence with two further bisections. First we split the right child node $\rho\mathsf{R}$ into its child nodes $\rho\mathsf{RL}$ and $\rho\mathsf{RR}$, respectively (again bisecting on the second coordinate of $\boldsymbol{x}_{\rho\mathsf{R}}$). Then we select $\rho\mathsf{LR}$ to do a final split and obtain its child nodes $\rho\mathsf{LRL}$ and $\rho\mathsf{LRR}$. $\boldsymbol{x}_{\rho\mathsf{LR}}$ is square and is bisected on its first coordinate to form the sub-boxes $\boldsymbol{x}_{\rho\mathsf{LRL}}$ and $\boldsymbol{x}_{\rho\mathsf{LRR}}$.

Figures 2 and 3 also illustrate an important point about these regular pavings. Because of our restricted bisection rule (splitting a box only at the mid-point along its first widest coordinate) a tree and its associated root box $\boldsymbol{x}_\rho$ will uniquely describe the partition of $\boldsymbol{x}_\rho$ into sub-boxes. In Figure 2 there is one and only one partition of the root box corresponding to each tree, and similarly in Figure 3. The same would

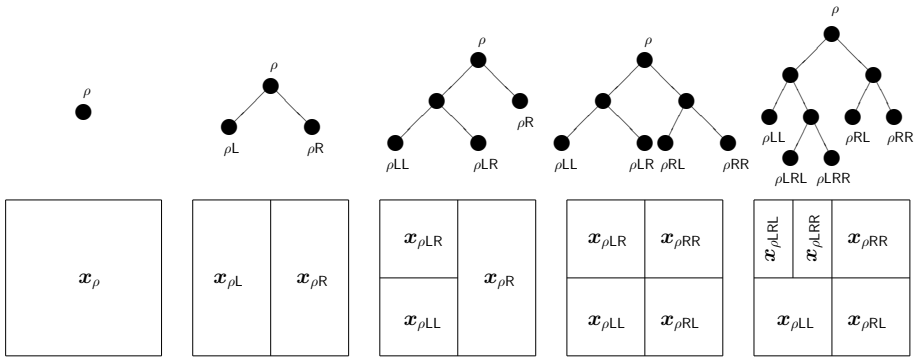


Figure 3: A sequence of selective bisections of boxes (nodes) along the first widest coordinate, starting from the root box (root node), produces an RP.

apply for a root box of any dimension. For the same reason, if we have two RPs with the same root box and two nodes at exactly the same positions in their respective trees (i.e., nodes that would have the same name-label in our visual presentations), then both of these nodes will have exactly the same box and can be considered to be 'equivalent'.

It is this restriction that allows us efficiently to carry out operations on two regular pavings that will result in another regular paving and to extend this to arithmetic on mapped regular pavings. We will discuss this further as we describe operations using regular pavings and mapped regular pavings. We consider the disadvantages of this restriction in Section 5. We now return to a general description of regular pavings.

Let the j-th interval of a box $\boldsymbol{x}_{\rho\mathsf{v}}$ be $[\underline{x}_{\rho\mathsf{v},j}, \overline{x}_{\rho\mathsf{v},j}]$. Then the volume of a d-dimensional box $\boldsymbol{x}_{\rho\mathsf{v}}$ associated with the node $\rho\mathsf{v}$ of an RP of $\boldsymbol{x}_\rho$ is the product of the side-lengths of the box, i.e.

$$\mathrm{vol}\,(\boldsymbol{x}_{\rho\mathsf{v}}) = \prod_{j=1}^{d} (\overline{x}_{\rho\mathsf{v},j} - \underline{x}_{\rho\mathsf{v},j}) \ .$$

The volume may also be associated with the *depth* of a node. A node has depth k if it can be reached by k splits from the root node. Then, the volume of any d-dimensional box $\boldsymbol{x}_{\rho\mathsf{v}}$ associated with node $\rho\mathsf{v}$ having depth k is $\mathrm{vol}\,(\boldsymbol{x}_{\rho\mathsf{v}}) = 2^{-k}\mathrm{vol}\,(\boldsymbol{x}_\rho)$. This is because we will always split a box exactly in half.

We use the nodes of the final RP in Figure 3 for illustration purposes. Assume that the root box $\boldsymbol{x}_\rho$ is a unit hypercube. Then the root node ρ has depth 0 and $\mathrm{vol}\,(\boldsymbol{x}_\rho) = 1$, the nodes $\rho\mathsf{L}$ and $\rho\mathsf{R}$ have depth 1 and volume 2^{-1}, the nodes $\rho\mathsf{LL}, \rho\mathsf{LR}, \rho\mathsf{RL}, \rho\mathsf{RR}$ have depth 2 and volume 2^{-2}, and finally the nodes $\rho\mathsf{LRL}$, $\rho\mathsf{LRR}$ have depth 3 and volume 2^{-3}.

We can now label each leaf node of a tree by its depth. The leaf nodes of the final RP in Figure 3, listed in left-right order, are $[\rho\mathsf{LL}, \rho\mathsf{LRL}, \rho\mathsf{LRR}, \rho\mathsf{RL}, \rho\mathsf{RR}]$. We can then also uniquely identify or label an RP with its *ordered leaf-depth string*. The final RP in Figure 3 has 23322 as its *ordered leaf-depth string*. Thus this RP can be denoted by s_{23322}.

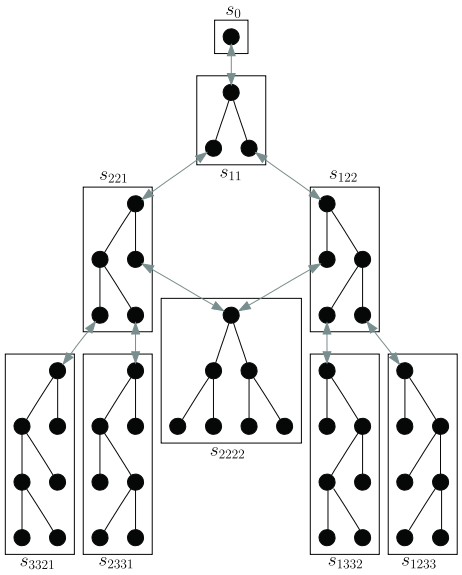


Figure 4: Transition diagram over $\mathbb{S}_{0:3}$ with split/reunion transitions from one RP state to another.

We denote the label set of all nodes of a regular paving by $\mathbb{V} := \rho \cup \{\rho\{\mathsf{L},\mathsf{R}\}^j : j \in \mathbb{N}\}$ and the set of all leaf nodes of a regular paving by $\mathbb{L}$. For the final regular paving in the sequence represented in Figure 3, $\mathbb{L}(s_{23322}) = \{\rho\mathsf{LL}, \rho\mathsf{RL}, \rho\mathsf{RR}, \rho\mathsf{LRL}, \rho\mathsf{LRR}\}$ and $\mathbb{V}(s_{23322}) = \{\rho, \rho\mathsf{L}, \rho\mathsf{R}, \rho\mathsf{LL}, \rho\mathsf{LR}, \rho\mathsf{RL}, \rho\mathsf{RR}, \rho\mathsf{LRL}, \rho\mathsf{LRR}\}$. The list of cherry nodes of s_{23322} is $c(s_{23322}) := [\rho\mathsf{LR}, \rho\mathsf{R}]$ and $\boldsymbol{x}_{c(s_{23322})} = \{\boldsymbol{x}_{\rho\mathsf{LR}}, \boldsymbol{x}_{\rho\mathsf{R}}\}$ is the set of boxes associated with them.

Having seen a particular RP s_{23322} let us study the space of all RPs. Let $\mathbb{S}_k$ be the set of all RPs of $\boldsymbol{x}_\rho$ made of k splits. Note that $|\mathbb{L}(s)| = k+1$ if $s \in \mathbb{S}_k$. The number of distinct binary trees with k splits is equal to the Catalan number

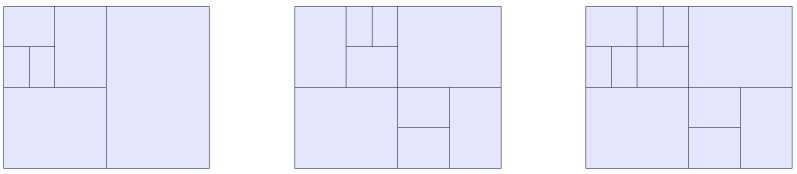
$$C_k = \frac{1}{k+1}\binom{2k}{k} = \frac{(2k)!}{(k+1)!(k!)} \ . \tag{1}$$

For $i, j \in \mathbb{Z}_+$, where $\mathbb{Z}_+ := \{0, 1, 2, \ldots\}$ and $i \leq j$, let $\mathbb{S}_{i:j}$ be the set of RPs with k splits where $k \in \{i, i+1, \ldots, j\}$. The space of all RPs is then $\mathbb{S}_{0:\infty} := \lim_{j\to\infty} \mathbb{S}_{0:j}$. Figure 4 displays the transition diagram over $\mathbb{S}_{0:3}$ where the gray arrows represent the transition from one RP state to another through a split or reunion. Each sequence of splits and merges of an RP s with root node ρ returns a partition of its root box $\boldsymbol{x}_\rho$ given by the set of its leaf boxes $\boldsymbol{x}_{\mathbb{L}(s)}$.

There may be more than one path from the root node to a particular RP in $\mathbb{S}_k$, i.e. more than one distinct sequence of k splits may result in the same RP in $\mathbb{S}_k$. In Figure 4, for example, there are two paths to s_{2222}.

Randomised algorithms of interest here are Markov chains on $\mathbb{S}_{0:\infty}$. In Section 4 we shall describe two such algorithms that use a randomised priority queue to rigorously approximate a real-valued function that has an interval inclusion function.

The union of two RPs $s^{(1)}$ and $s^{(2)}$ in $\mathbb{S}_{0:\infty}$ with the same root box $\boldsymbol{x}_\rho$ is denoted by $s^{(1)} \cup s^{(2)}$. Intuitively, the leaf boxes of the union of two RPs can be seen as being obtained from overlaying or superimposing the partitions of the operand RPs as shown in Figure 5.



(a) $s^{(1)}$ (b) $s^{(2)}$ (c) $s^{(1)} \cup s^{(2)}$

Figure 5: Union of two regular pavings of a root box in $\mathbb{R}^2$.

Let $\rho\mathsf{v}$ be a node of an RP s and let the Boolean function `IsLeaf`$(\rho\mathsf{v})$ return true if $\rho\mathsf{v}$ is a leaf node and false otherwise. Let `Copy`$(\rho\mathsf{v})$ return a copy of the RP tree rooted at node $\rho\mathsf{v}$. If $\rho\mathsf{v}$ is not a leaf node then let $\rho\mathsf{vL}$ and $\rho\mathsf{vR}$ be the left and right child nodes of $\rho\mathsf{v}$, respectively. Consider two RPs $s^{(1)}$ and $s^{(2)}$ with root nodes $\rho^{(1)}$ and $\rho^{(2)}$ respectively and the same root box $\boldsymbol{x}_\rho$. Then, `RPUnion`$(\rho^{(1)}, \rho^{(2)})$ given by Algorithm 1 returns the union of the two RP trees. Figure 6 shows two RPs of the same box $\boldsymbol{x}_\rho \in \mathbb{IR}^2$ and their union.

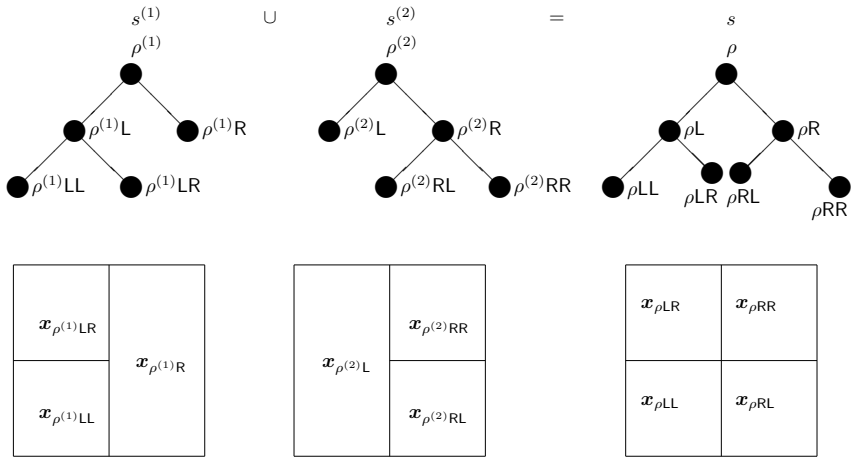


Figure 6: Union on the RPs $s^{(1)}$ and $s^{(2)}$.

Observe that the union operation $\cup$ is subject only to the restriction that the two operand RPs have the same root box. Remarkably, RPs are closed under such unions, i.e., if $s^{(1)}, s^{(2)} \in \mathbb{S}_{0:\infty}$ then $s^{(1)} \cup s^{(2)} =: s \in \mathbb{S}_{0:\infty}$. This can be easily seen as a process of overhead transparencies: take two transparencies, one with the RP tree $s^{(1)}$ drawn on it and the other with the tree $s^{(2)}$; lay one transparency over the other, aligning the roots, and you have produced $s := s^{(1)} \cup s^{(2)}$. Thus, since each $s^{(i)} \in \{s^{(1)}, s^{(2)}\}$

is a subtree of the union, each $s^{(i)}$ can be expanded to s by adding splits. For more details on the above argument and connections with Thompson's group see proof of [5, Prop. 10.3].

Note that the tree structure is exploited to give an algorithm that recurses simultaneously on pairs of nodes, first left-child pairs and then right-child pairs. The structure thus provides a very simple way of matching up sub-boxes of the RPs 'layer by layer' in the tree hierarchies in order to find the finest partitioning of any part of the shared root box and copy it into the result of the union operation.

Note also that the reason why the union operation can be carried out on any two RPs $s^{(1)}, s^{(2)} \in \mathbb{S}_{0:\infty}$ provided only that they have the same root box is that we restrict the bisection of a box to bisection perpendicularly at the mid-point along the first widest coordinate. This restriction means that if some equivalent node $\rho*$ exists in both $s^{(1)}$ and $s^{(2)}$ (for example, $\rho^{(1)}\mathsf{L}$ and $\rho^{(2)}\mathsf{L}$ in Figure 6) then the boxes associated with these nodes ($\boldsymbol{x}_{\rho^{(1)}\mathsf{L}}$ and $\boldsymbol{x}_{\rho^{(2)}\mathsf{L}}$ in Figure 6) will be identical. Similarly, if some part of $s^{(1)}$, for instance, is 'more partitioned' than that part of $s^{(2)}$ (as the left half of $s^{(1)}$ is in Figure 6), then that same partition can be exactly replicated in $s^{(2)}$ or a copy of $s^{(2)}$ by simply following this general bisection rule.

If we did not restrict bisections of a box in this way we might record, in each node that has been split, the split coordinate and split point on that coordinate used to divide the box. This would not be enough, however, to give us such a general union operation: we would only be able to guarantee being able to carry out the union of two pavings and obtain another such paving where both pavings had identical split dimensions and split points in each pair of equivalent nodes in their respective trees. The restriction on the method of bisection for RPs so that each tree shape can represent just one partition of the root box means that the union of the trees is equivalent to the overlay of the partitions and the result will be another RP.

We emphasise these points here because the union operation is fundamental to many of the arithmetic operations on mapped regular pavings described in the next section.

4 Mapped Regular Pavings

Let $s \in \mathbb{S}_{0:\infty}$ be an RP with root node ρ and root box $\boldsymbol{x}_\rho \in \mathbb{IR}^d$ and let $\mathbb{Y}$ be a non-empty set. Let $\mathbb{V}(s)$ and $\mathbb{L}(s)$ denote the set of all nodes and leaf nodes of s, respectively. Let $f : \mathbb{V}(s) \to \mathbb{Y}$ map each node of s to an element in $\mathbb{Y}$ as follows:

$$\{\rho\mathsf{v} \mapsto f_{\rho\mathsf{v}} : \rho\mathsf{v} \in \mathbb{V}(s), f_{\rho\mathsf{v}} \in \mathbb{Y}\} \ .$$

Such a map f is called a *$\mathbb{Y}$-mapped regular paving ($\mathbb{Y}$-MRP)*. Thus, a $\mathbb{Y}$-MRP f is obtained by augmenting each node $\rho\mathsf{v}$ of the RP tree s with an additional data member $f_{\rho\mathsf{v}}$.

Let $\eta(x)$ be the node label of the unique leaf box $\boldsymbol{x}_{\eta(x)}$ in $\boldsymbol{x}_{\mathbb{L}(s)}$ that contains the point $x \in \boldsymbol{x}_\rho \in \mathbb{IR}^d$.

Due to the recursive partitioning structure of $\mathbb{Y}$-MRP trees it is computationally efficient to find $\eta(x)$ for a given $x \in \boldsymbol{x}_\rho$ and then look up its image $f_{\eta(x)}$ according to Algorithm 2.

Recall that in Section 3 we carefully defined the bisection of a box in a RP s so that the intersection between sibling child boxes is empty. Algorithm 2 relies on this, i.e. we know that x can never be in both $\boldsymbol{x}_{\rho\mathsf{vL}}$ and $\boldsymbol{x}_{\rho\mathsf{vR}}$ of any node $\rho\mathsf{v} \in \mathbb{V}(s)$.

Algorithm 1: $\texttt{RPUnion}(\rho^{(1)}, \rho^{(2)})$

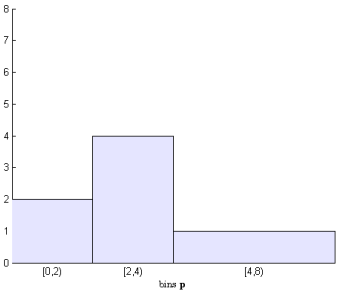
input : Root nodes $\rho^{(1)}$ and $\rho^{(2)}$ of RPs $s^{(1)}$ and $s^{(2)}$, respectively, with root box $\boldsymbol{x}_{\rho^{(1)}} = \boldsymbol{x}_{\rho^{(2)}}$.
output : Root node ρ of RP $s = s^{(1)} \cup s^{(2)}$.

if $\texttt{IsLeaf}(\rho^{(1)})$ & $\texttt{IsLeaf}(\rho^{(2)})$ **then**
  $\rho \leftarrow \texttt{Copy}(\rho^{(1)})$
  return ρ
end
else if $!\texttt{IsLeaf}(\rho^{(1)})$ & $\texttt{IsLeaf}(\rho^{(2)})$ **then**
  $\rho \leftarrow \texttt{Copy}(\rho^{(1)})$
  return ρ
end
else if $\texttt{IsLeaf}(\rho^{(1)})$ & $!\texttt{IsLeaf}(\rho^{(2)})$ **then**
  $\rho \leftarrow \texttt{Copy}(\rho^{(2)})$
  return ρ
end
else
  $!\texttt{IsLeaf}(\rho^{(1)})$ & $!\texttt{IsLeaf}(\rho^{(2)})$
end
Make ρ as a node with $\boldsymbol{x}_\rho \leftarrow \boldsymbol{x}_{\rho^{(1)}}$
Graft onto ρ as left child the node $\texttt{RPUnion}(\rho^{(1)}\mathsf{L}, \rho^{(2)}\mathsf{L})$
Graft onto ρ as right child the node $\texttt{RPUnion}(\rho^{(1)}\mathsf{R}, \rho^{(2)}\mathsf{R})$
return ρ

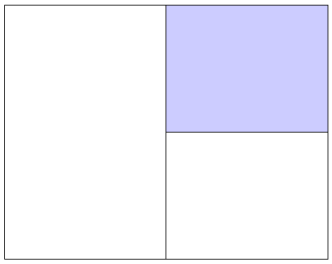
Algorithm 2: $\texttt{PointWiseImage}(\rho, x)$

input : ρ with box $\boldsymbol{x}_\rho$, the root node of $\mathbb{Y}$-MRP f with RP s, and a point $x \in \boldsymbol{x}_\rho$.
output : Return $f_{\eta(x)}$ at the leaf node $\eta(x)$ that is associated with the box $\boldsymbol{x}_{\eta(x)}$ containing x.

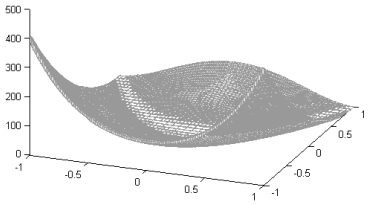
if $\texttt{IsLeaf}(\rho)$ **then**
  return f_ρ
end
else
  if $x \in \boldsymbol{x}_{\rho\mathsf{R}}$ **then**
    $\texttt{PointWiseImage}(\rho\mathsf{R}, x)$
  end
  else
    $\texttt{PointWiseImage}(\rho\mathsf{L}, x)$
  end
end



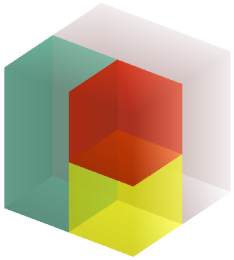
(a) $\mathbb{R}$-MRP over s_{221} with $x_\rho = [0, 8]$



(b) $\mathbb{B}$-MRP over s_{122} with $x_\rho = [0, 1]^2$



(c) $\mathbb{IR}$-MRP enclosure of the Rosenbrock function with $x_\rho = [-1, 1]^2$



(d) $[0, 1]^3$-MRP over s_{3321} with $x_\rho = [0, 1]^3$

Figure 7: Examples of mapped regular pavings.

Then from a given $\mathbb{Y}$-MRP f with root node ρ we can define a *pointwise extension* $f : \boldsymbol{x}_\rho \to \mathbb{Y}$ from the *piecewise constant map* $x \mapsto f_{\eta(x)}$ as follows:

$$f(x) = f_{\eta(x)} \quad \text{if } x \in \boldsymbol{x}_\rho \ ,$$

and easily obtain $f_{\eta(x)}$ from `PointWiseImage`(ρ, x). Note that $f(x)$ is undefined if $x \notin \boldsymbol{x}_\rho$.

We give some examples of mapped regular pavings in Figure 7. Figure 7(a) shows an $\mathbb{R}$-MRP $f(x) : [0, 8] \to \mathbb{R}$ over RP s_{221} ($\mathbb{R}$-MRPs are depicted in 'histogram' form to relate the mapped values to the-sub boxes of the partition).

Let $\mathbb{B} := \{\mathsf{True}, \mathsf{False}\}$ denote the set of Boolean values. Figure 7(b) shows a $\mathbb{B}$-MRP $a(x) : [0, 1]^2 \to \mathbb{B}$ representing a subset A of the root box $\boldsymbol{x} = [0, 1]^2$ by mapping the leaf boxes of the RP tree s_{122} with Boolean values that indicate membership in A. A True-mapped leaf box is shown as filled space. Figure 7(c) shows an $\mathbb{IR}$-MRP enclosure of the Rosenbrock function $f(x) : \mathbb{R}^2 \to \mathbb{R}$, $f(x) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2$ on $[-1, 1]^2$. Figure 7(d) shows a $[0, 1]^3$-MRP $g(x) : [0, 1]^3 \to [0, 1]^3$ over s_{3321} in RGB colouring scheme.

As an example of the flexibility of mapped regular pavings, we can use Boolean-mapped regular pavings $\mathbb{B}$-MRP to represent the regular subpavings of [2]. A *regular subpaving* of $\boldsymbol{x}_\rho$ is formed by a finite succession of selective bisections and deletions,

whereby some of the sub-boxes (leaf nodes) are deleted. Regular subpavings are used in interval analysis to approximate compact sets in a guaranteed way (eg. [2]). In this article we are concerned with pavings, not subpavings: in a paving no sub-box is deleted — equivalently, each node in the tree either has two children or none. By mapping Boolean values to the leaves of our regular pavings we can include the regular subpavings of [2] within our framework. (Strictly speaking, the leaf boxes in the regular pavings of [2] do not partition $\boldsymbol{x}_\rho$ due to non-empty intersections between the child boxes. Thus the elements of $\boldsymbol{x}_{\mathbb{L}(s)}$ in our RP are not necessarily elements of $\mathbb{IR}^d \cap \boldsymbol{x}_\rho$. However, we can always take the closure of each leaf box $\boldsymbol{x}_{\rho\mathsf{v}}$ in our RP that may be partly open to obtain compact leaf boxes that belong to $\mathbb{IR}^d \cap \boldsymbol{x}_\rho$ as in [2], when necessary.)

Let the class of $\mathbb{Y}$-mapped regular pavings over the leaf boxes of regular pavings of a root box $\boldsymbol{x}_\rho \in \mathbb{IR}^d$ be :

$$\mathcal{F} := \{\{\rho\mathsf{v} \mapsto f_{\rho\mathsf{v}} : \rho\mathsf{v} \in \mathbb{V}(s), f_{\rho\mathsf{v}} \in \mathbb{Y}\} : s \in \mathbb{S}_{0:\infty}\} \quad .$$

Suppose we are interested in representing a constant function over $\boldsymbol{x}_\rho$ that maps each $x \in \boldsymbol{x}_\rho$ to the constant $c \in \mathbb{Y}$ using a $\mathbb{Y}$-MRP in $\mathcal{F}$. Then any $\mathbb{Y}$-MRP f in $\{\{\rho\mathsf{v} \mapsto c : \rho\mathsf{v} \in \mathbb{V}(s)\} : s \in \mathbb{S}_{0:\infty}\} \subset \mathcal{F}$, an equivalence class in $\mathcal{F}$ under pointwise function equality, may be used to represent our constant function. We make the canonical choice

$$\curlywedge(f) = \{\rho \mapsto c : \rho \in \mathbb{V}(s_0)\} \ ,$$

with the smallest possible RP tree s_0, to uniquely represent this equivalence class.

More generally, elements of $\mathcal{F}$ are piecewise constant functions where the pieces are the leaf boxes of the RP tree. Once again we use the smallest possible tree to uniquely represent the equivalence class in $\mathcal{F}$ under function equality for each piecewise constant function. Let `IsCherry`$(\rho\mathsf{v})$ return true if $\rho\mathsf{v}$ is a sub-terminal (cherry) node and false otherwise and let `Prune`$(\rho\mathsf{v})$ prune the node $\rho\mathsf{v}$ and its descendants from the MRP tree. Let ρ be the root node of a $\mathbb{Y}$-MRP f. We can now obtain a unique minimal representative $\curlywedge(f)$ of f by calling `MinimiseLeaves`(ρ) in Algorithm 3.

Algorithm 3: `MinimiseLeaves`(ρ)

input : ρ, the root node of $\mathbb{Y}$-MRP f.
output : Modify f into $\curlywedge(f)$, the unique $\mathbb{Y}$-MRP with fewest leaves.

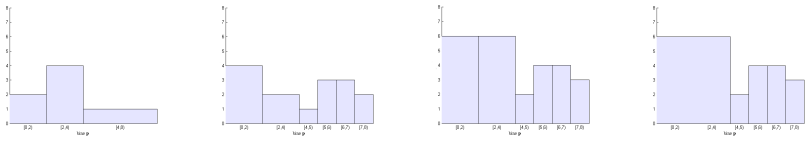
```
if !IsLeaf(ρ) then
    MinimiseLeaves(ρL)
    MinimiseLeaves(ρR)
    if IsCherry(ρ) & ( f_ρL = f_ρR ) then
        f_ρ ← f_ρL
        Prune(ρL)
        Prune(ρR)
    end
end
```

Let f and g be two $\mathbb{Y}$-MRPs with the same root box $\boldsymbol{x}_\rho$ and let $\mathbb{Y}$ be a field. Let us extend arithmetic from $\mathbb{Y}$ to $\mathcal{F}$ to obtain a field of functions through pointwise operations, i.e., $(f \star g)(x) = f(x) \star g(x)$, where $\star \in \{+, -, \cdot, /\}$, up to the operation

$\star$ being well-defined in $\mathbb{Y}$. The additive and multiplicative identities, say 0 and 1 in $\mathbb{Y}$, immediately yield the constant $\mathbb{Y}$-MRPs $\rho \mapsto 0$ and $\rho \mapsto 1$ as the additive and multiplicative identities in $\mathcal{F}$, respectively.

Before we describe how arithmetic over $\mathbb{Y}$-MRPs is carried out, let us look at two simple examples. Let $\mathbb{1}_A(x)$ be the indicator function of a set A, i.e., $\mathbb{1}_A(x) = 1$ if $x \in A$ and 0 otherwise.

Example 1 ($\mathbb{R}$-MRP arithmetic)**:** $\mathbb{R}$-MRP arithmetic can be demonstrated by adding two real-valued simple functions over the root box $\boldsymbol{x}_\rho = [0, 8]$. The arguments $f(x) = 2\mathbb{1}_{\boldsymbol{x}_{\rho\mathsf{LL}}}(x) + 4\mathbb{1}_{\boldsymbol{x}_{\rho\mathsf{LR}}}(x) + \mathbb{1}_{\boldsymbol{x}_{\rho\mathsf{R}}}(x)$ and $g(x) = 4\mathbb{1}_{\boldsymbol{x}_{\rho\mathsf{LL}}}(x) + 2\mathbb{1}_{\boldsymbol{x}_{\rho\mathsf{LR}}}(x) + \mathbb{1}_{\boldsymbol{x}_{\rho\mathsf{RLL}}}(x) + 3\mathbb{1}_{\boldsymbol{x}_{\rho\mathsf{RLR}}}(x) + 3\mathbb{1}_{\boldsymbol{x}_{\rho\mathsf{RRL}}}(x) + 2\mathbb{1}_{\boldsymbol{x}_{\rho\mathsf{RRR}}}(x)$ over s_{221} and s_{223333}, are shown in Figures 8(a) and 8(b), respectively. Their sum and its unique representative with the smallest RP tree are shown in Figure 8(c) and 8(d), respectively.



(a) f (b) g (c) $f + g$ (d) $\rightarrow(f + g)$

Figure 8: Addition of two $\mathbb{R}$-MRPs over $[0, 8]$ (see text for description).

Example 2 ($\mathbb{B}$-MRP arithmetic)**:** Figure 9 shows the results of Boolean arithmetic operations between two $\mathbb{B}$-MRPs A_1 and A_2 to represent subsets of $\mathbb{IR}^2$. As in Figure 7(b), True-mapped leaf boxes are showed as filled. We can similarly use $\mathbb{B}$-MRPs of $\boldsymbol{x}_\rho \in \mathbb{IR}^3$ to represent and manipulate voxel images (as in Figure 1(b)).

Algorithm 4 is an extension of Algorithm 1 used to perform binary operations over $\mathbb{Y}$-MRPs.

Note that, like Algorithm 1, Algorithm 4 can be expressed very simply because it recurses simultaneously on pairs of nodes and the pairs are easily found as a result of the tree structure of the pavings. Also like Algorithm 1, Algorithm 4 can be used on any two $\mathbb{Y}$-MRPs subject only to the condition that they have the same root box $\boldsymbol{x}_\rho$ because of the restrictive bisection rule used to create a regular paving.

Note also that (again because of the restrictive bisection rule), the boxes of any two equivalent nodes in the trees rooted at $\rho^{(1)}$ and $\rho^{(2)}$ will be exactly the same, including having exactly the same closed or partly open interval boundaries. When Algorithm 4 copies boxes for new nodes to make the new MRP h, they are always the 'right' (only possible) boxes for those nodes, and the possible mixture of closed or partly open intervals that may make up any box causes no complications in the algorithm or for its implementation.

Let the function `MRPTransform`(ρ, τ) apply the unary transformation $\tau : \mathbb{Y} \to \mathbb{Y}$ to a given $\mathbb{Y}$-MRP f with root node ρ and return the transformed $\mathbb{Y}$-MRP $g = \tau(f)$ by recursively descending through the tree and setting $f_\rho \leftarrow \tau(f_\rho)$ for each node ρ.

At any step during such a process of binary operations and/or unary transformations over $\mathbb{Y}$-MRPs we can use Algorithm 3 to reduce the $\mathbb{Y}$-MRPs to their unique minimal representatives and use Algorithm 2 to find the point-wise image of any point in the root box. These Algorithms on $\mathbb{Y}$-MRPs and their compositions enable us to computationally conduct arithmetical expressions directly over $\mathbb{Y}$-MRPs.

Algorithm 4: $\texttt{MRPOperate}(\rho^{(1)}, \rho^{(2)}, \star)$

input : two root nodes $\rho^{(1)}$ and $\rho^{(2)}$ with same root box $\boldsymbol{x}_{\rho^{(1)}} = \boldsymbol{x}_{\rho^{(2)}}$ and binary operation $\star$.

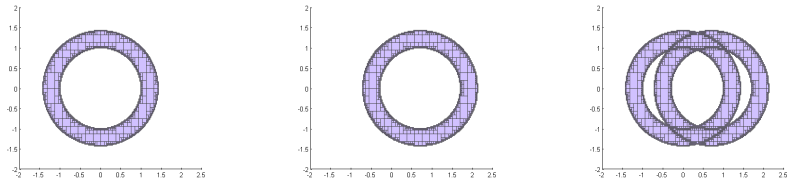
output : the root node ρ of $\mathbb{Y}$-MRP $h = f \star g$.

Make a new node ρ with box and image
$\boldsymbol{x}_\rho \leftarrow \boldsymbol{x}_{\rho^{(1)}}$; $h_\rho \leftarrow f_{\rho^{(1)}} \star g_{\rho^{(2)}}$

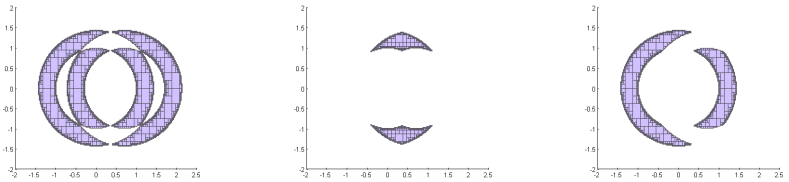
if $\texttt{IsLeaf}(\rho^{(1)})$ & $\texttt{!IsLeaf}(\rho^{(2)})$ **then**
  Make temporary nodes L', R'
  $\boldsymbol{x}_{\mathsf{L}'} \leftarrow \boldsymbol{x}_{\rho^{(2)}\mathsf{L}}$; $\boldsymbol{x}_{\mathsf{R}'} \leftarrow \boldsymbol{x}_{\rho^{(2)}\mathsf{R}}$
  $f_{\mathsf{L}'} \leftarrow f_{\rho^{(1)}}$, $f_{\mathsf{R}'} \leftarrow f_{\rho^{(1)}}$
  Graft onto ρ as left child the node $\texttt{MRPOperate}(\mathsf{L}', \rho^{(2)}\mathsf{L}, \star)$
  Graft onto ρ as right child the node $\texttt{MRPOperate}(\mathsf{R}', \rho^{(2)}\mathsf{R}, \star)$
end

else if $\texttt{!IsLeaf}(\rho^{(1)})$ & $\texttt{IsLeaf}(\rho^{(2)})$ **then**
  Make temporary nodes L', R'
  $\boldsymbol{x}_{\mathsf{L}'} \leftarrow \boldsymbol{x}_{\rho^{(1)}\mathsf{L}}$; $\boldsymbol{x}_{\mathsf{R}'} \leftarrow \boldsymbol{x}_{\rho^{(1)}\mathsf{R}}$
  $g_{\mathsf{L}'} \leftarrow g_{\rho^{(2)}}$, $g_{\mathsf{R}'} \leftarrow g_{\rho^{(2)}}$
  Graft onto ρ as left child the node $\texttt{MRPOperate}(\rho^{(1)}\mathsf{L}, \mathsf{L}', \star)$
  Graft onto ρ as right child the node $\texttt{MRPOperate}(\rho^{(1)}\mathsf{R}, \mathsf{R}', \star)$
end

else if $\texttt{!IsLeaf}(\rho^{(1)})$ & $\texttt{!IsLeaf}(\rho^{(2)})$ **then**
  Graft onto ρ as left child the node $\texttt{MRPOperate}(\rho^{(1)}\mathsf{L}, \rho^{(2)}\mathsf{L}, \star)$
  Graft onto ρ as right child the node $\texttt{MRPOperate}(\rho^{(1)}\mathsf{R}, \rho^{(2)}\mathsf{R}, \star)$
end
return ρ



(a) A_1 (b) A_2 (c) $A_1 + A_2$



(d) $A_1 - A_2$ (e) $A_1 \times A_2$ (f) $A_1 \div A_2$

Figure 9: Two Boolean-mapped regular pavings A_1 and A_2 and Boolean arithmetic operations with $+$ for set union, $-$ for symmetric set difference, $\times$ for set intersection, and $\div$ for set difference.

We give some further examples of arithmetic on mapped regular pavings: the addition and subtraction of 3-d regular pavings (cubes) mapped to colours and the multiplication of 2-d regular pavings mapped to vectors.

Example 3 ($[0,1]^3$-MRPs as coloured cubes)**:** In this example the mapped regular pavings are on a cube $\boldsymbol{x}_\rho \in \mathbb{IR}^3$ and the mapping assigns a colour to each sub-box in a paving of the cube. A colour-mapped regular paving is a multi-coloured cube.

(a) f (b) g (c) $f + g$ (d) $f - g$

Figure 10: Two $[0,1]^3$-MRPs f and g as multicoloured cubes, showing $f + g$ and $f - g$.

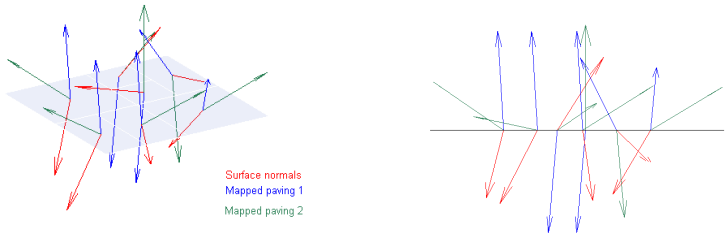
In this example, colours were implemented as RGB colour tuples (for example blue is $(0,0,1)$). Black is $(0,0,0)$ and is the additive identity, white is $(1,1,1)$ and is the multiplicative identity. In our simple implementation of colour arithmetic, adding many different colours will eventually result in white while subtracting a colour from itself will give black. These operations could have been implemented differently which would cause the equivalent operations on the mapped regular pavings to behave differently.

Example 4 (vector-MRPs)**:** In this example the mapped regular pavings are on a rectangle $\boldsymbol{x}_\rho \in \mathbb{IR}^2$ and the mapping assigns a normalised vector (i.e. a vector of length 1 or a directional vector) to each sub-box in a paving of the rectangle. A vector-mapped regular paving shows areas of the rectangle mapped to directional vectors.

(a) View 1 of f (b) View 2 of f (c) View 1 of g (d) View 2 of g

Figure 11: Two views of vector-MRP f and g.

We defined the $\times$ operation for the vectors used for implementing this example as a computer program to give the cross product of the two operand vectors (the cross product gives the direction of the surface normal of the unique plane defined by the two operand vectors). Equipped with this we can perform the $\times$ operation on vector-mapped regular pavings and get the cross product mapped onto each box in the mapped regular union of the pavings of the two operand vector-mapped regular pavings. In Figure 12 we show the operand vectors (in blue and green) as well as the surface normal vector, in red, resulting from the cross-product operation $\times$.



(a) View 1 (b) View 2

Figure 12: $f \times g$.

Arithmetic over vector-MRPs can be useful when working with empirical vector fields (coarse-grained observations of a dynamical system). For instance, adding several such vector-MRPs of the same system with multiple initial conditions can produce a vector-MRP of the global (or average) dynamics.

Let us consider arithmetic and algebra of $\mathbb{R}$-MRPs. Given any two $\mathbb{R}$-MRPs $f^{(1)}$ and $f^{(2)}$ with the same root box $\boldsymbol{x}_\rho$ and with root nodes $\rho^{(1)}$ and $\rho^{(2)}$, respectively, and a binary operation $\star \in \{+, -, \cdot, /\}$, we can obtain the resultant $\mathbb{R}$-MRP $f = f^{(1)} \star f^{(2)}$ via `MRPOperate`$(\rho^{(1)}, \rho^{(2)}, \star)$ of Algorithm 4. We can also transform an $\mathbb{R}$-MRP f

with root node ρ using any standard function $\tau \in \mathfrak{S} := \{\exp, \sin, \cos, \tan, \ldots\}$ and obtain the resultant $\mathbb{R}$-MRP $\tau(f)$ using `MRPTransform`(ρ, τ). Finally, we can obtain an $\mathbb{R}$-MRP arithmetical expression that is specified by finitely many sub-expressions involving constant $\mathbb{R}$-MRP, binary arithmetic operations over two $\mathbb{R}$-MRPs, standard transformations of $\mathbb{R}$-MRPs by elements of $\mathfrak{S}$ and their compositions. Thus we can obtain $\mathbb{R}$-MRPs as arithmetical expressions of other $\mathbb{R}$-MRPs.

Next we state a useful theorem about $\mathbb{R}$-MRPs and use it to obtain a simple recipe for approximating continuous functions.

Theorem 4.1. *Let $\mathcal{F}$ be the class of $\mathbb{R}$-MRPs with the same root box $\boldsymbol{x}_\rho$. Then $\mathcal{F}$ is dense in $C(\boldsymbol{x}_\rho, \mathbb{R})$, the algebra of real-valued continuous functions on $\boldsymbol{x}_\rho$.*

Proof. Since $\boldsymbol{x}_\rho \in \mathbb{IR}^d$ is a compact Hausdorff space, by the Stone-Weierstrass theorem we can establish that $\mathcal{F}$ is dense in $C(\boldsymbol{x}_\rho, \mathbb{R})$ with the topology of uniform convergence, provided that $\mathcal{F}$ is a sub-algebra of $C(\boldsymbol{x}_\rho, \mathbb{R})$ that separates points in $\boldsymbol{x}_\rho$ and which contains a non-zero constant function.

Constructively by Algorithm 4, with $\star = +$, $\mathcal{F}$ is closed under function addition and using `MRPTransform` with $\tau(y) = r \times y$, for any given $r \in \mathbb{R}$, $\mathcal{F}$ is closed under scalar multiplication and therefore $\mathcal{F}$ is a sub-algebra of $C(\boldsymbol{x}_\rho, \mathbb{R})$. $\mathcal{F}$ contains non-zero constant functions, for e.g. $\left(\mathbb{1}_{\boldsymbol{x}_\rho}(x) r\right) \in \mathcal{F}$ for any given $r \in \mathbb{R}$. Finally, RPs can clearly separate distinct points $x, x' \in \boldsymbol{x}_\rho$ into distinct leaf boxes by splitting deeply enough. Therefore we have satisfied all the conditions of the Stone-Weierstrass theorem and have proved that $\mathcal{F}$ is dense in $C(\boldsymbol{x}_\rho, \mathbb{R})$. □

Let $\boldsymbol{x}_\rho \in \mathbb{IR}^d$ and $g : \boldsymbol{x}_\rho \to \mathbb{R}$ be a continuous function that is known pointwise for any $x \in \boldsymbol{x}_\rho$, i.e., we have a procedure that returns $g(x)$ for a given $x \in \boldsymbol{x}_\rho$. Our objective is to obtain an $\mathbb{R}$-MRP f with RP s in order to approximate a simple function $g : \boldsymbol{x}_\rho \to \mathbb{R}$, $\boldsymbol{x}_\rho \in \mathbb{IR}^d$. Theorem 4.1 guarantees that a simple function $f(x) = \sum_{\rho\mathsf{v} \in \mathbb{L}(s)} \mathbb{1}_{\boldsymbol{x}_{\rho\mathsf{v}}}(x) f_{\rho\mathsf{v}}$ obtained from an $\mathbb{R}$-MRP f with RP s based on $f_{\rho\mathsf{v}} = \operatorname{mid}(g(\boldsymbol{x}_{\rho\mathsf{v}}))$ over a partition formed by the leaf boxes $\{\boldsymbol{x}_{\rho\mathsf{v}} : \rho\mathsf{v} \in \mathbb{L}(s)\}$ can uniformly approximate g provided $\operatorname{mesh}(s) := \max_{\rho\mathsf{v} \in \mathbb{L}(s)} \max(\operatorname{wid}(\boldsymbol{x}_{\rho\mathsf{v}}))$, the mesh of the partition, goes to zero.

This means that we could try to approximate g using an algorithm that bisects each sub-box in the partition (splits each node ρ in the tree) until $\max(\operatorname{wid}(\boldsymbol{x}_{\rho\mathsf{v}})) \leq \varepsilon$ where ε is the specified mesh size. For each node, $f_\rho \leftarrow g(\operatorname{mid}(\boldsymbol{x}_\rho))$.

A serious disadvantage to using the ε-mesh approximation to g is that we do not know the uniform bound ϵ (note, not the same as the mesh ε) in the range:

$$\sup_{x \in \boldsymbol{x}_\rho} \operatorname{abs}(f(x) - g(x)) \leq \epsilon \ .$$

In practical terms, this means that we cannot guarantee how close the approximation f is to g. In order to be able to rigorously bound the range we need the notion of inclusion functions from interval analysis.

Let $\boldsymbol{x}_\rho$ and $\boldsymbol{y}$ be complete lattices. Typically, $\boldsymbol{x}_\rho \in \mathbb{IR}^d$ and $\boldsymbol{y} \in \mathbb{IR}^c$. Let $\mathcal{G}$ be the set of all functions from $\mathbb{I}\boldsymbol{x}_\rho$ to $\mathbb{I}\boldsymbol{y}$. Then, $(\mathcal{G}, \leq)$ is a complete lattice under the order relation $f \leq g \iff \forall x \in \boldsymbol{x}_\rho, f(x) \leq g(x)$ and an interval in $\mathcal{G}$ is $\boldsymbol{f} = [\underline{f}, \overline{f}]$ such that $\underline{f} \leq \overline{f}$. Now suppose we are given a function $g : \boldsymbol{x}_\rho \to \boldsymbol{y}$. Then an *inclusion function* of g is a function $\boldsymbol{g} : \mathbb{I}\boldsymbol{x} \to \mathbb{I}\boldsymbol{y}$ that satisfies:

$$\text{range enclosure:} \quad \forall \boldsymbol{x} \in \mathbb{I}\boldsymbol{x}_\rho, \quad g(\boldsymbol{x}) := \{g(x) : x \in \boldsymbol{x}\} \subseteq \boldsymbol{g}(\boldsymbol{x}) \ , \tag{2}$$

$$\text{inclusion monotone:} \quad \forall \boldsymbol{x}, \boldsymbol{x}' \in \mathbb{I}\boldsymbol{x}_\rho, \quad \boldsymbol{x} \subset \boldsymbol{x}' \implies g(\boldsymbol{x}) \subseteq \boldsymbol{g}(\boldsymbol{x}') \ . \tag{3}$$

We say that $g : \boldsymbol{x}_\rho \to \mathbb{R}$ has a *well-defined natural interval extension* $\boldsymbol{g} : \mathbb{I}\boldsymbol{x}_\rho \to \mathbb{IR}$ if $\boldsymbol{g}(\boldsymbol{x})$ that is obtained by replacing the sub-expressions in g with their interval counterparts satisfies $\boldsymbol{g}(\boldsymbol{x}) \in \mathbb{IR}^d$ for each $\boldsymbol{x} \in \mathbb{I}\boldsymbol{x}_\rho$. Note that by the fundamental theorem of interval analysis [6] a well-defined interval extension of g is indeed an inclusion function that satisfies Equations (2) and (3).

Theorem 4.2 (Enclosing range of functions)**.** *Let $\boldsymbol{x}_\rho \in \mathbb{IR}^d$ and $g : \boldsymbol{x}_\rho \to \mathbb{R}$ be a function with a well-defined inclusion function $\boldsymbol{g} : \mathbb{I}\boldsymbol{x}_\rho \to \mathbb{IR}$. Then, for a given $\epsilon > 0$: $\texttt{UniformApprox}(\rho, g, \boldsymbol{g}, \epsilon)$ of Algorithm 5 will produce an $\mathbb{IR}$-MRP $\boldsymbol{f}$ that uniformly encloses g using intervals of radius no larger than ϵ, i.e.,*

$$\forall x \in \boldsymbol{x}_\rho, \left(\boldsymbol{f}(x) := \sum_{\rho\mathsf{v} \in \mathbb{L}(s)} \mathbb{1}_{\boldsymbol{x}_{\rho\mathsf{v}}}(x) \boldsymbol{f}_{\rho\mathsf{v}} \supset g(x) \right) \text{ and } (\operatorname{wid}(\boldsymbol{f}(x)) \leq 2\epsilon) \quad .$$

Proof. The proof follows from the fundamental theorem of interval analysis by an induction argument on the sub-expressions of the arithmetical expression defining g [6]. □

Algorithm 5: $\texttt{UniformApprox}(\rho, g, \boldsymbol{g}, \epsilon)$

input : ρ, the root node of $\mathbb{IR}$-MRP f with root box $\boldsymbol{x}_\rho$, function g to be approximated, its inclusion function $\boldsymbol{g}$, and tolerance ϵ.
output : $\mathbb{IR}$-MRP $\boldsymbol{f}$ such that for any $x \in \boldsymbol{x}_\rho$, $g(x) \in f(x)$, $\operatorname{rad}(f(x)) \leq \epsilon$.

```
if !IsLeaf(ρ) then
    UniformApprox(ρL, g, g, ε)
    UniformApprox(ρR, g, g, ε)
end
else
    f_ρ ← g(□(x_ρ))
    if max{sup(g(x_ρ)) − g(mid(x_ρ)), g(mid(x_ρ)) − inf(g(x_ρ))} > ε
    then
        Split ρ: ▽(ρ) = {ρL, ρR}
        UniformApprox(ρL, g, g, ε)
        UniformApprox(ρR, g, g, ε)
    end
end
```

Using Algorithm 5 and Theorem 4.2 we can get an $\mathbb{IR}$-MRP $\boldsymbol{f}$ approximation of g such that for any leaf node ρ in the tree, for any $x \in \boldsymbol{x}_\rho$, $g(x) \in f(x)$ and $\operatorname{rad}(f(x)) \leq \epsilon$ for some specified tolerance ϵ. This means that we can, in theory, get an $\mathbb{IR}$-MRP $\boldsymbol{f}$ that encloses g as tightly as we want.

We can also turn this $\mathbb{IR}$-MRP $\boldsymbol{f}$ approximation of g into a $\mathbb{R}$-MRP by setting $f_\rho \leftarrow g(\operatorname{mid}(\boldsymbol{x}_\rho))$ for each node ρ in the tree. This will produce an $\mathbb{R}$-MRP f that is ϵ-close to g (in the sup norm), i.e., $\sup_{x \in \boldsymbol{x}_\rho} \operatorname{abs}(f(x) - g(x)) \leq \epsilon$, using the same argument as we use to prove Theorem 4.2.

A practical disadvantage of Algorithm 5 is that it is ultimately limited by available machine memory to store the MRP tree used to approximate $g : \boldsymbol{x}_\rho \to \mathbb{R}$, $\boldsymbol{x}_\rho \in \mathbb{IR}^d$, especially for large d. In some cases it may be impossible to run Algorithm 5 for a specified range tolerance ϵ because the required tree would be too large (have too many leaves). One could experimentally increase the tolerance to produce an MRP approximation that can be held in machine memory. A more sensible strategy is to produce an MRP approximation that tries to meet the range tolerance condition, or some other suitable criterion for the tightness of the enclosure, in an optimal manner for a given maximum number of leaves.

One way to create such a strategy is to specify an appropriate priority function $\psi : \mathbb{L}(s) \to \mathbb{R}$ on the set of leaves of the current MRP with RP s and split a leaf node that is uniformly chosen at random from $\operatorname{argmax}_{\rho\mathsf{v}\in\mathbb{L}(s)} \psi(\rho\mathsf{v})$, the set of leaf nodes of s which are equally 'large' when measured using ψ.

In Examples 5 and 6 we use $\psi(\rho\mathsf{v}) = \operatorname{vol}(\rho\mathsf{v})\operatorname{wid}(\boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{v}}))$. This priority function measures the *volume of interval enclosure* of the leaf node $\rho\mathsf{v}$ (the product of the volume of the box $\boldsymbol{x}_{\rho\mathsf{v}}$ and the width of its image interval under $\boldsymbol{g}$). This ψ prioritises the splitting of leaf nodes with the largest volume of interval enclosure.

Example 5 (prioritised splitting)**:** Let us illustrates this with a simple example in one dimension when the function to be enclosed is $g : [0,1] \to \mathbb{R}$ where $g(x) = x^2 + (x+1)\sin(10\pi x)^2\cos(3\pi x)^2$ with inclusion function $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{x}^2 + (\boldsymbol{x}+1)\sin(10\pi\boldsymbol{x})^2\cos(3\pi\boldsymbol{x})^2$. We use the priority function $\psi(\rho\mathsf{v}) = \operatorname{vol}(\rho\mathsf{v})\operatorname{wid}(\boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{v}}))$.

We start with a single (root) node and root box [0.0, 1.0]. Since this is the only node it is trivially the node with the largest $\psi(\rho\mathsf{v})$, and so will be split. The leaf nodes are now $\rho\mathsf{R}$ and $\rho\mathsf{L}$ with range enclosures $\boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{R}})$ and $\boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{L}})$ respectively.

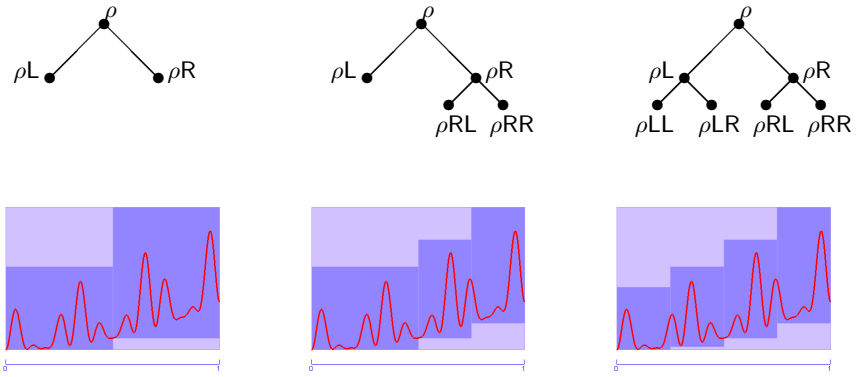
Figure 13 shows the next few steps:

1. Figure13(a) shows interval enclosures of the function on $\boldsymbol{x}_{\rho\mathsf{L}} = [0.0, 0.5)$ and $\boldsymbol{x}_{\rho\mathsf{R}} = [0.5, 1.0]$. The volume of the interval enclosure of $\boldsymbol{x}_{\rho\mathsf{R}}$ is the largest, i.e. $\psi(\rho\mathsf{R}) > \psi(\rho\mathsf{L})$, and so $\rho\mathsf{R}$ is the next node to be split.
2. Figure 13(b) shows interval enclosures of the function on $\boldsymbol{x}_{\rho\mathsf{L}} = [0.0, 0.5)$, $\boldsymbol{x}_{\rho\mathsf{RL}} = [0.5, 0.75)$ and $\boldsymbol{x}_{\rho\mathsf{RR}} = [0.75, 1.0]$. The volume of the interval enclosure of $\boldsymbol{x}_{\rho\mathsf{L}}$ is the largest ($\operatorname{argmax}_{\rho\mathsf{v}\in\mathbb{L}(s)} \psi(\rho\mathsf{v}) = \{\rho\mathsf{L}\}$) and $\rho\mathsf{L}$ is the next node to be split.
3. Figure 13(c) shows interval enclosures of the function on $\boldsymbol{x}_{\rho\mathsf{LL}} = [0.0, 0.25)$, $\boldsymbol{x}_{\rho\mathsf{LR}} = [0.25, 0.5)$, $\boldsymbol{x}_{\rho\mathsf{RL}} = [0.5, 0.75)$ and $\boldsymbol{x}_{\rho\mathsf{RR}} = [0.75, 1.0]$. $\operatorname{argmax}_{\rho\mathsf{v}\in\mathbb{L}(s)} \psi(\rho\mathsf{v}) = \{\rho\mathsf{RR}\}$ and $\rho\mathsf{RR}$ would be the next node to be split if we carried the example further.

In Example 5 we only have a single 'largest' node at each step, but it is possible that there may be more than one such largest node. Once the priority function is motivated we can implement such a sequential bisection procedure using a *randomised priority queue* over the current set of leaf nodes of the MRP. The randomised priority queue will choose the node to be split uniformly at random from $\operatorname{argmax}_{\rho\mathsf{v}\in\mathbb{L}(s)} \psi(\rho\mathsf{v})$.

We also need a rule that tells us when to stop splitting (a 'stopping rule'). A simple rule is to stop splitting when $|\mathbb{L}(s)|$, the total number of leaf nodes in the MRP, reaches some maximum limit $\bar{\ell}$ that is usually imposed by internal machine memory.

Finally, we are equipped with the necessary ingredients to produce a memory-efficient $\mathbb{IR}$-MRP $\boldsymbol{f}$ with RP s and root box $\boldsymbol{x}_\rho \in \mathbb{IR}^d$ that encloses the function $g : \boldsymbol{x}_\rho \to \mathbb{R}$ that has an inclusion function $\boldsymbol{g} : \mathbb{I}\boldsymbol{x}_\rho \to \mathbb{IR}$ using at most $\bar{\ell} = |\mathbb{L}(s)|$ many leaves, such that $\operatorname{vol}(\boldsymbol{x}_{\rho\mathsf{v}}) \cdot \operatorname{wid}(\boldsymbol{f}_{\rho\mathsf{v}})$ for each leaf node $\rho\mathsf{v} \in \mathbb{L}(s)$ is made as small as



(a) Two leaves (b) Three leaves (c) Four leaves

Figure 13: Priority split.

possible under a heuristic sequential splitting strategy based on a randomised priority queue of leaves.

Algorithm 6: $\texttt{RPQEnclose}^{\nabla}(\rho, \boldsymbol{g}, \psi, \bar{\ell})$

---

input : ρ, the root node of $\mathbb{IR}$-MRP $\boldsymbol{f}$ with RP s, root box $\boldsymbol{x}_\rho$ and $\boldsymbol{f}_\rho = \boldsymbol{g}(\boldsymbol{x}_\rho)$,
$\psi : \mathbb{L}(s) \to \mathbb{R}$ such that
$\psi(\rho\mathsf{v}) = \operatorname{vol}(\boldsymbol{x}_{\rho\mathsf{v}})\,(\boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{v}}) - 0.5\,(\boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{vL}}) + \boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{vR}})))$,
$\bar{\ell}$ the maximum number of leaves.

output : $\boldsymbol{f}$ with modified RP s such that $|\mathbb{L}(s)| = \bar{\ell}$.

if $|\mathbb{L}(s)| < \bar{\ell}$ **then**

$\rho\mathsf{v} \leftarrow \texttt{random\_sample}\left(\operatorname{argmax}_{\rho\mathsf{v}\in\mathbb{L}(s)} \psi(\rho\mathsf{v})\right)$

Split $\rho\mathsf{v}$: $\nabla(\rho\mathsf{v}) = \{\rho\mathsf{vL}, \rho\mathsf{vR}\}$; `// split the sampled node`

$\boldsymbol{f}_{\rho\mathsf{vL}} \leftarrow \boldsymbol{g}(\square(\boldsymbol{x}_{\rho\mathsf{vL}}))$

$\boldsymbol{f}_{\rho\mathsf{vR}} \leftarrow \boldsymbol{g}(\square(\boldsymbol{x}_{\rho\mathsf{vL}}))$

$\texttt{RPQEnclose}^{\nabla}(\rho, \psi, \bar{\ell})$

end

The result of this sequential splitting that is informed by the randomised priority queue with priority function ψ is an $\mathbb{IR}$-MRP $\boldsymbol{f}$ with RP $s \in \mathbb{S}_{0:\bar{\ell}-1}$. Note that we can supply any reasonable priority function ψ to Algorithm 6 in order to find an enclosure satisfying the desired priority criterion. For instance, $\psi(\rho\mathsf{v}) = \operatorname{wid}(\boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{v}}))$ will prioritise the splitting of leaf nodes with the widest range enclosure.

Recall that our aim in developing this procedure was to find a way to make an $\mathbb{IR}$-MRP f that encloses a target simple function g "in an optimal manner for a given maximum number of leaves". The priority function ψ reflects what is meant by

'optimal': $\psi(\rho\mathsf{v}) = \text{wid}\,(\boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{v}}))$ aims for the smallest maximum range enclosures on any sub-box of the partition; $\psi(\rho\mathsf{v}) = \text{vol}\,(\rho\mathsf{v})\text{wid}\,(\boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{v}}))$ aims to minimise the total volume of the interval enclosures of the sub-boxes $\sum_{\rho\mathsf{v}\in\mathbb{L}(s)} \left(\text{vol}\,(\boldsymbol{x}_{\rho\mathsf{v}})\text{wid}\,(\boldsymbol{f}_{\rho\mathsf{v}})\right)$.

Unfortunately, the process `RPQEnclose`$^{\nabla}$ of Algorithm 6 has not quite achieved this. We cannot guarantee that there is no other $\mathbb{IR}$-MRP that would give a better enclosure. For example, if we use $\psi(\rho\mathsf{v}) = \text{vol}\,(\rho\mathsf{v})\text{wid}\,(\boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{v}}))$ we cannot guarantee that the $\mathbb{IR}$-MRP $\boldsymbol{f}$ produced by Algorithm 6 has RP s that belongs to $\mathbb{S}^*_{\psi,\bar{\ell}} := \{\text{argmin}_{s\in\mathbb{S}_{0:\bar{\ell}-1}} \sum_{\rho\mathsf{v}\in\mathbb{L}(s)} \left(\text{vol}\,(\boldsymbol{x}_{\rho\mathsf{v}})\text{wid}\,(\boldsymbol{f}_{\rho\mathsf{v}})\right)\}$ (i.e. that s is one of the possible RPs giving the smallest possible total volume of the interval enclosures of the sub-boxes).

Algorithm 6 makes locally optimal choices based on a current set of leaves and ψ but these are not necessarily globally optimal decisions. It might seem that we could make some improvement by 'looking ahead' to the actual result of the split, for example using a priority function that measures the *reduction* in total volume of the interval enclosure that would result from splitting a node. This is still only a local decision: it only takes into account the results of the immediate next split. An algorithm making globally optimal decisions would have to be able to compare all possible $s \in \mathbb{S}_{0:\bar{\ell}-1}$. Recalling that there are $\sum_{k=0}^{\bar{\ell}-1} C_k$ such s, we can see that this is computationally infeasible.

In practice we have found that our heuristic splitting strategy for several target functions g, using $\psi(\rho\mathsf{v}) = \text{vol}\,(\rho\mathsf{v})\text{wid}\,(\boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{v}}))$, produces an s that is not too far from states in $\mathbb{S}^*_{\psi,\bar{\ell}}$ in terms of their total volume of the interval enclosures of the sub-boxes.

We now describe a refinement to our heuristic splitting strategy that can be guaranteed to give a range enclosure that is at least as good as that achieved by the use of `RPQEnclose`$^{\nabla}$ alone, and in many cases better.

This refinement is based on the observation that the more we split (the larger $\bar{\ell}$ is) the smaller the boxes associated with the leaves and (by Equation 3) the tighter the range enclosures of the inclusion function $\boldsymbol{g}$ over these boxes. This suggests that we could benefit by splitting to a larger number of leaves than we really want, using the information from the range enclosures at the leaves to adjust (tighten) the range enclosures of the internal nodes, and then perform a 'prioritised prune' to reduce the tree to some ultimate target number of leaves.

This is the idea behind Algorithms 7 and 8.

Let $\boldsymbol{x} = [\underline{x}, \overline{x}]$ and $\boldsymbol{y} = [\underline{y}, \overline{y}]$ be two intervals, then the smallest interval containing the union of $\boldsymbol{x}$ and $\boldsymbol{y}$ is called their *interval hull* and given by $\boldsymbol{x} \sqcup \boldsymbol{y} := [\min(\underline{x}, \underline{y}), \max(\overline{x}, \overline{y})]$.

Algorithm 7: `HullPropagate`(ρ)

input : ρ, the root node of $\mathbb{IR}$-MRP $\boldsymbol{f}$ with RP s.
output : Modify input MRP $\boldsymbol{f}$.

```
if !IsLeaf(ρ) then
    HullPropagate(ρL)
    HullPropagate(ρR)
    f_ρ ← f_ρL ⊔ f_ρR
end
```

`HullPropagate`(ρ) works from the leaves of $\boldsymbol{f}$ upwards to replace the intervals mapped onto the internal nodes with the interval hull of the intervals mapped onto

their children. Thus, $\text{wid}\,(\boldsymbol{f}_{\rho\mathsf{v}}) \leq \text{wid}\,(\boldsymbol{g}(x_{\rho\mathsf{v}}))$ for each internal node $\rho\mathsf{v} \in \mathbb{V}(s) \setminus \mathbb{L}(s)$ after an $\boldsymbol{f}$ produced by $\texttt{RPQEnclose}^{\nabla}$ has been modified by a call to $\texttt{HullPropagate}(\rho)$.

The 'prioritised prune' to fewer leaves is achieved by Algorithm 8. When we prune, we select a cherry node from $\mathbb{C}(s)$ the set of cherry nodes $\mathbb{IR}$-MRP $\boldsymbol{f}$ and prune off both children so that the selected node becomes a leaf node. Algorithm 8 uses some appropriate priority function $\psi : \mathbb{C}(s) \to \mathbb{R}$ and chooses the next cherry node for pruning uniformly at random from $\text{argmin}_{\rho\mathsf{v}\in\mathbb{C}(s)}\,\psi(\rho\mathsf{v})$, i.e. from the cherries that are the *smallest* measured using ψ.

In Example 6 we again use $\psi(\rho\mathsf{v}) = \text{vol}\,(\rho\mathsf{v})\text{wid}\,(\boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{v}}))$, the priority function that measures the volume of the interval enclosure of a node $\rho\mathsf{v}$. This ψ prioritises the pruning of cherry nodes with the smallest volume of interval enclosure.

A simple stopping-rule is to keep pruning cherries back into leaves until $|\mathbb{L}(s)|$, the total number of leaf nodes in the MRP, has decreased to some maximum limit $\bar{\ell}'$.

Algorithm 8: $\texttt{RPQEnclose}^{\Delta}(\rho, \psi, \bar{\ell}')$

---

input : ρ, the root node of $\mathbb{IR}$-MRP $\boldsymbol{f}$ with RP s, box $\boldsymbol{x}_\rho$,
$\psi : \mathbb{C}(s) \to \mathbb{R}$ as $\psi(\rho\mathsf{v}) = \text{vol}\,(\boldsymbol{x}_{\rho\mathsf{v}})\left(\boldsymbol{f}_{\rho\mathsf{v}} - 0.5\left(\boldsymbol{f}_{\rho\mathsf{vL}} + \boldsymbol{f}_{\rho\mathsf{vR}}\right)\right)$,
$\bar{\ell}'$ the maximum number of leaves.

output : modified $\boldsymbol{f}$ with RP s such that $|\mathbb{L}(s)| = \bar{\ell}'$ or $\mathbb{C}(s) = \emptyset$.

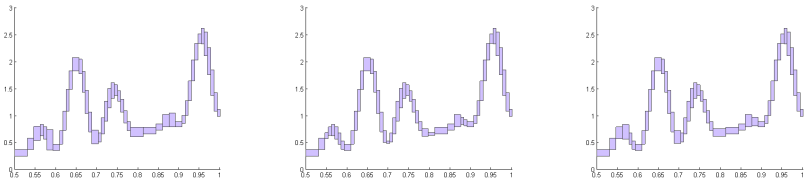
if $|\mathbb{L}(s)| \geq \bar{\ell}'$ & $\mathbb{C}(s) \neq \emptyset$ **then**
  $\rho\mathsf{v} \leftarrow \texttt{random\_sample}\left(\text{argmin}_{\rho\mathsf{v}\in\mathbb{C}(s)}\,\psi(\rho\mathsf{v})\right)$; `// choose a random node with smallest` ψ
  $\texttt{Prune}(\rho\mathsf{L})$
  $\texttt{Prune}(\rho\mathsf{R})$
  $\texttt{RPQEnclose}^{\Delta}(\rho, \psi, \bar{\ell}')$
end

Notice that the processes $\texttt{HullPropagate}$ and $\texttt{RPQEnclose}^{\Delta}$ can be nicely separated because of the tree structure. This allows a much clearer, simpler and more flexible implementation.

We can use Algorithm 6, Algorithm 7 and Algorithm 8 in succession to obtain an interval-based approximation for the function $g : \boldsymbol{x}_\rho \to \mathbb{R}$ that has an inclusion function $\boldsymbol{g}$. We start with $\mathbb{IR}$-MRP $\boldsymbol{f}$ consisting of just the root node ρ with an appropriate root box $\boldsymbol{x}_\rho$ where $\boldsymbol{f}_\rho = \boldsymbol{g}(\boldsymbol{x}_\rho)$. First we use $\texttt{RPQEnclose}^{\nabla}$ of Algorithm 6, specifying some priority function ψ and some maximum number of leaves $\bar{\ell}$ that is larger than we want in our final approximation. The result is $\boldsymbol{f}$ with RP $s \in \mathbb{S}_{0:\bar{\ell}-1}$. Then we perform $\texttt{HullPropagate}(\rho)$ of Algorithm 7 on the root node ρ of $\boldsymbol{f}$ to tighten the range enclosures on the internal nodes of $\boldsymbol{f}$. Finally we reduce the number of leaves to $\bar{\ell}' < \bar{\ell}$ using $\texttt{RPQEnclose}^{\Delta}$ of Algorithm 8, again specifying some priority function for the pruning.

Example 6 (split, propagate hull & prune)**:** Let us reconsider the function $g : [0, 1] \to \mathbb{R}$ where $g(x) = x^2 + (x+1)\sin(10\pi x)^2\cos(3\pi x)^2$ with the well-defined inclusion function $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{x}^2 + (\boldsymbol{x}+1)\sin(10\pi\boldsymbol{x})^2\cos(3\pi\boldsymbol{x})^2$ of Example 5. Suppose we first try to approximate g on domain [0.5, 1.0] with $\mathbb{IR}$-MRP $\boldsymbol{f}$ made by $\texttt{RPQEnclose}^{\nabla}(\rho, \boldsymbol{g}, \psi, \bar{\ell})$ that encloses the range of g with intervals over an RP s with $|\mathbb{L}(s)| = \bar{\ell} = 45$ leaves and $\psi(\rho\mathsf{v}) = \text{vol}\,(\rho\mathsf{v})\text{wid}\,(\boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{v}}))$. Figure 14(a) displays such an $\boldsymbol{f}$. We can mea-

sure the tightness of the range enclosure of this $\mathbb{IR}$-MRP $\boldsymbol{f}$ by finding the volume of the enclosure over the leaf nodes $\Xi(\boldsymbol{f}) := \sum_{\rho\mathsf{v}\in\mathbb{L}(s)} \mathrm{wid}\,(\boldsymbol{f}_{\rho\mathsf{v}})\mathrm{vol}\,(\boldsymbol{x}_{\rho\mathsf{v}}) = 0.1215$. Figure 14(b) displays the $\mathbb{IR}$-MRP $\boldsymbol{f}'$ made by $\texttt{RPQEnclose}^{\nabla}(\rho, \boldsymbol{g}, \psi, \bar{\ell})$ with $\bar{\ell} = 50$ leaves. With 5 additional leaves the volume of the enclosure over the leaf nodes is smaller, $\Xi(\boldsymbol{f}') = 0.1102$. For 5 extra leaves we have reduced the total volume of the enclosure by $\Xi(\boldsymbol{f}) - \Xi(\boldsymbol{f}') = 0.1215 - 0.1102 = 0.0113$. However, by calling $\texttt{HullPropagate}(\rho')$ upon ρ', the root node of $\boldsymbol{f}'$ (50 leaves), we can propagate the hull of the range enclosures from the leaf nodes of $\boldsymbol{f}'$ up through its internal nodes. Finally, we can call $\texttt{RPQEnclose}^{\Delta}(\rho', \psi, \bar{\ell}')$ with $\bar{\ell}' = 45$ and $\psi(\rho\mathsf{v}) = \mathrm{vol}\,(\rho\mathsf{v})\mathrm{wid}\,(\boldsymbol{g}(\boldsymbol{x}_{\rho\mathsf{v}}))$. We obtain a modified $\mathbb{IR}$-MRP $\boldsymbol{f}'$ that has only 45 leaves as shown in Figure 14(c) with $\Xi(\boldsymbol{f}') = 0.1159$. Now $\boldsymbol{f}'$ has the same number of leaves as $\boldsymbol{f}$ and a tighter range enclosure: $\Xi(\boldsymbol{f}) - \Xi(\boldsymbol{f}') = 0.1215 - 0.1159 = 0.0056$.



(a) (b) (c)

Figure 14: $\mathbb{IR}$-MRPs to enclose the function g of Example 6 using $\texttt{RPQEnclose}^{\nabla}(\rho, \boldsymbol{g}, \psi, 45)$ in (a), $\texttt{RPQEnclose}^{\nabla}(\rho', \boldsymbol{g}, \psi, 50)$ in (b) and $\texttt{HullPropagate}(\rho')$ followed by $\texttt{RPQEnclose}^{\Delta}(\rho', \psi, 45)$ in (c).

As with an $\mathbb{IR}$-MRP $\boldsymbol{f}$ approximation of g produced by Algorithm 5, we can turn $\boldsymbol{f}$ produced by Algorithm 6 alone or by the sequence Algorithm 6, Algorithm 7, Algorithm 8, into a mid-point approximation $\mathbb{R}$-MRP by setting $f_\rho \leftarrow g(\mathrm{mid}\,(\boldsymbol{x}_\rho))$ for each node ρ in the tree.

Suppose we have an $\mathbb{R}$-MRP f with RP s containing $n = |\mathbb{L}(s)|$ leaf nodes with non-negative image values, i.e., $f_{\rho\mathsf{v}} > 0$, for each $\rho\mathsf{v} \in \mathbb{L}(s)$. In statistical interpretations of probability density functions we are often interested in the highest coverage region that gives the smallest possible subset of leaf nodes such that they constitute the highest image values and integrate at least to a specified fraction α of the integral of f over all of its leaf nodes. Algorithm 9 gives us a straightforward method to find the coverage region of a non-negative f for a specified α.

Example 7 (Levy Density)**:** The bivariate Levy density $\dot{l}(t_1, t_2)$ over $\mathbb{T} = [-10, 10]^2$ with normalising constant $N_l := \int_{[-10,10]^2} l(t_1, t_2) dt_1 dt_2$ has 700 modes and is given by:

$$\begin{aligned}
\dot{l}(t_1, t_2) =& \frac{1}{N_l} l(t_1, t_2), \quad l(t_1, t_2) = \exp\left(-\Upsilon(t_1, t_2)\right), \\
\Upsilon(t_1, t_2) =& \sum_{i=1}^{5} i \cos\left((i-1)t_1 + i\right) \sum_{j=1}^{5} j \cos\left((j+1)t_2 + j\right) \\
&+ (t_1 + 1.42513)^2 + (t_2 + 0.80032)^2.
\end{aligned}$$

Algorithm 9: `CoverageRegion`(ρ, α)

input : ρ, root node of $\mathbb{R}$-MRP f with RP s, n leaf nodes in $\mathbb{L}(s)$ and a real number α such that $0 \leq \alpha \leq 1$.

output : $\ell(s)$, smallest sub-set of leaf nodes of f that contain the highest α region, i.e., $\int_{\rho \mathsf{v} \in \ell(s)} f_{\rho \mathsf{v}} \mathrm{vol}\,(\boldsymbol{x}_{\rho \mathsf{v}}) \geq \alpha$ and $\min_{\rho \mathsf{v} \in \ell(s)} f_{\rho \mathsf{v}} > \max_{\rho \mathsf{v} \in \mathbb{L}(s) \setminus \ell(s)} f_{\rho \mathsf{v}}$.

initialize: $\ell(s) \leftarrow \emptyset$; $a \leftarrow 0$; $i \leftarrow 1$
; `// next sort leaves by height`
$\mathbb{L}^{\downarrow}(s) = [\rho \mathsf{v}_1, \rho \mathsf{v}_2, \ldots, \rho \mathsf{v}_n]$, $f_{\rho \mathsf{v}_1} \geq f_{\rho \mathsf{v}_2} \geq \cdots \geq f_{\rho \mathsf{v}_n}$
$N_f \leftarrow \sum_{i=1}^{n} f_{\rho \mathsf{v}_i} \mathrm{vol}\,(\boldsymbol{x}_{\rho \mathsf{v}_i})$; `// normalising constant`

while $i \leq n$ & $a < \alpha$ **do**
  $a \leftarrow a + (f_{\rho \mathsf{v}_i} \mathrm{vol}\,(\boldsymbol{x}_{\rho \mathsf{v}_i}))/N_f$
  $i \leftarrow i + 1$
  $\ell(s)$.`append`$(\boldsymbol{x}_{\rho \mathsf{v}_i})$; `// insert this node into the set` $\ell(s)$
end
return $\ell(s)$

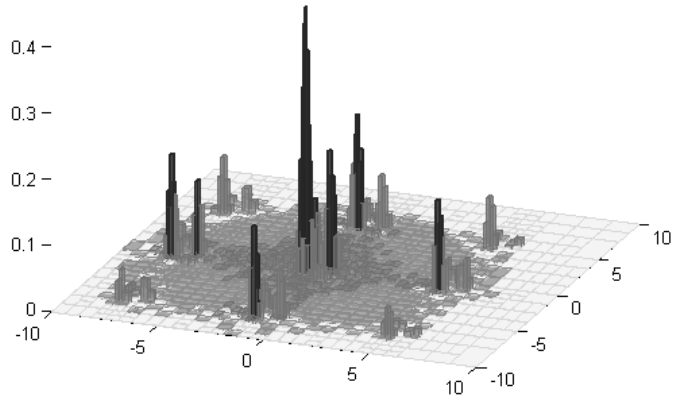


Figure 15: $\mathbb{R}$-MRP approximation to Levy density $\dot{l}$ with coverage regions for $\alpha = 0.9$ (light gray), $\alpha = 0.5$ (dark gray) and $\alpha = 0.1$ (black)

Figure 15 shows various coverage regions for a mid-point approximation of $\dot{l}$ with $\mathbb{R}$-MRP f that is obtained from the leaf nodes of an $\mathbb{IR}$-MRP f made by first splitting to 2000 leaves via $\texttt{RPQEnclose}^{\nabla}(\rho, \dot{l}, \psi, \bar{\ell} = 2000)$, then calling $\texttt{HullPropagate}(\rho)$ and finally pruning via $\texttt{RPQEnclose}^{\Delta}(\rho, \psi, \bar{\ell}' = 1400)$.

The coverage of f with $\alpha = 0.9$ is shown in light gray, $\alpha = 0.5$ in dark gray, and $\alpha = 0.1$ in black.

Note that the 1400-leaf $\mathbb{R}$-MRP f obtained as above (first splitting to 2000 leaves, then calling `HullPropagate` and finally pruning via back up) has $\Xi(\boldsymbol{f}) = 13.81$. If we directly split to 1400 leaves with $\texttt{RPQEnclose}^{\nabla}(\rho, \dot{l}, \psi, \bar{\ell} = 1400)$ then $\Xi(\boldsymbol{f}) = 18.21$. Thus, in this example there is a significant gain in the tightness of the range enclosure by splitting, propagating the hulls and then pruning back.

A particularly useful operation over $\mathbb{R}$-MRPs is to marginalise along a subset of its coordinates. Let $\Lambda \subset \Delta = \{1, 2, \ldots, d\}$ such that $\Lambda \neq \emptyset$. We introduce some notation for tuples and subtuples of a point or Δ-tuple:

$$x = (x_1, \ldots, x_d) =: (x_i)_{i \in \Delta} \in \boldsymbol{x}_\rho = (\boldsymbol{x}_{\rho,i})_{i \in \Delta} = \boxtimes_{i \in \Delta} [\underline{x}_{\rho,i}, \overline{x}_{\rho,i}] \in \mathbb{IR}^d \ .$$

The Λ-subtuple of a Δ-tuple is obtained by retaining the subset of coordinates Λ out of the d coordinates in Δ as follows:

$$(x_j)_{j \in \Lambda} \in (\boldsymbol{x}_{\rho,j})_{j \in \Lambda} = \boxtimes_{j \in \Lambda} [\underline{x}_{\rho,j}, \overline{x}_{\rho,j}] \in \mathbb{IR}^{|\Lambda|} \ .$$

Suppose we have an $\mathbb{R}$-MRP f with RP s, root box $\boldsymbol{x}_\rho = \boxtimes_{j \in \Delta} [\underline{x}_{\rho,j}, \overline{x}_{\rho,j}] \in \mathbb{IR}^d$ and we are interested in the marginal function f^Λ with RP s^Λ and root box $\boldsymbol{x}_\rho^\Lambda$ given by:

$$f^\Lambda\left((x_i)_{i \in \Lambda}\right) = \int_{\boxtimes_{j \in \Delta \setminus \Lambda} [\underline{x}_{\rho,j}, \overline{x}_{\rho,j}]} f\left((x_i)_{i \in \Delta}\right) d\left(x_j\right)_{j \in \Delta \setminus \Lambda} ,$$

$$\forall (x_i)_{i \in \Lambda} \in \boldsymbol{x}_\rho^\Lambda = \boxtimes_{i \in \Lambda} [\underline{x}_{\rho,i}, \overline{x}_{\rho,i}] \ . \quad (4)$$

Then we can use Algorithm 10 to obtain the marginal function of f as another $\mathbb{R}$-MRP. Moreover, if f is non-negative then we can renormalise f as well as the marginal function of f to obtain a marginal density that integrates to 1. Obtaining marginal densities over a subset of coordinates of a joint density is a fundamental operation with multivariate densities in statistical operations.

Example 8 (Levy Marginals)**:** Consider the $\mathbb{R}$-MRP f of Figure 15 that approximates the Levy density $\dot{l}(x_1, x_2)$ on $[-10, 10]^2$. Figure 16(a) and 16(b) show the $\mathbb{R}$-MRP marginal densities $f^{\{1\}}(x_1) = \int_{-10}^{10} f(x_1, x_2) dx_2$ and $f^{\{2\}}(x_2) = \int_{-10}^{10} f(x_1, x_2) dx_1$ over the first and the second coordinates respectively of the $\mathbb{R}$-MRP f. Thus, if $\dot{l}(x_1, x_2)$ is describing the density corresponding to inverse energy of a set of particles distributed over $[-10, 10]^2$ then $f^{\{1\}}(x_1)$ and $f^{\{2\}}(x_2)$ describe the densities of the same particles along each one of the two coordinates over $[-10, 10]$.

Suppose we have a $\mathbb{Y}$-MRP $f : \boldsymbol{x}_\rho \to \mathbb{Y}$ with subpaving s and root box $\boldsymbol{x}_\rho \in \mathbb{IR}^d$. Another useful operation is to be able to 'slice' f orthogonal to one or more of its coordinates specified by $\Lambda \subset \Delta = \{1, 2, \ldots, d\}$ such that $0 < |\Lambda| < |\Delta| = d$. A 'slice' of the $\mathbb{Y}$-MRP f at a fixed subtuple point $x' = (x'_{\rho,j})_{j \in \Lambda} \in (\boldsymbol{x}_{\rho,j})_{j \in \Lambda}$ is the $\mathbb{Y}$-MRP

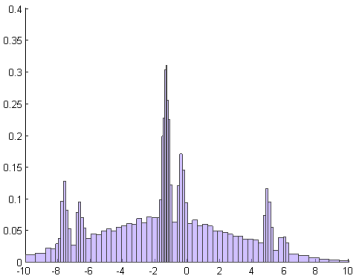
Algorithm 10: `Marginalise`(ρ, Λ)

input : ρ, the root node of $\mathbb{R}$-MRP f with RP s, root box $\boldsymbol{x}_\rho = \bigboxtimes_{j\in\Delta} [\underline{x}_{\rho,j}, \overline{x}_{\rho,j}] \in \mathbb{IR}^d$, $\Lambda \subset \Delta$ and $\Lambda \neq \emptyset$.

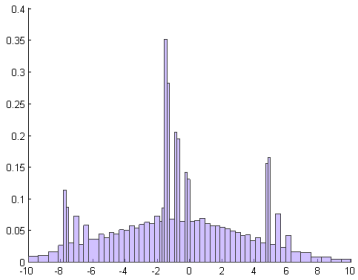
output : Change f into f^Λ of (4) with RP s^Λ and root box $\boldsymbol{x}_\rho^\Lambda = \bigboxtimes_{j\in\Lambda} [\underline{x}_{\rho,j}, \overline{x}_{\rho,j}]$.

if !`IsLeaf`(ρ) **then**
  `Marginalise`$(\rho\mathsf{L}, \Lambda)$
  `Marginalise`$(\rho\mathsf{R}, \Lambda)$
end

if !`IsLeaf`(ρ) & $\iota \notin \Lambda$ **then**
  $\rho \leftarrow$ `MRPOperate`$(\rho\mathsf{L}, \rho\mathsf{R}, +)$
end
else
  $\boldsymbol{x}_\rho \leftarrow \boldsymbol{x}_\rho^\Lambda$
  $f_\rho \leftarrow f_\rho^\Lambda = \frac{f_\rho \mathrm{vol}\,(\boldsymbol{x}_\rho)}{\mathrm{vol}\,(\boldsymbol{x}_\rho^\Lambda)}$
end



(a) $f^{\{1\}}(x_1) = \int_{-10}^{10} f(x_1, x_2)dx_2$



(b) $f^{\{2\}}(x_2) = \int_{-10}^{10} f(x_1, x_2)dx_1$

Figure 16: Marginal densities $f^{\{1\}}(x_1)$ and $f^{\{2\}}(x_2)$ along each coordinate of the $\mathbb{R}$-MRP f of Figure 15 that approximates the Levy density $\dot{l}(x_1, x_2)$ on $[-10, 10]^2$.

$f^{|x'}$ with RP $s^{|x'}$ and root box $(\boldsymbol{x}_{\rho,i})_{i\in\Delta\setminus\Lambda} := \boxtimes_{i\in\Delta\setminus\Lambda}[\underline{x}_{\rho,i}, \overline{x}_{\rho,i}]$ that satisfies:

$$f^{|x'}\left((x_i)_{i\in\Delta\setminus\Lambda}\right) = f\left((x_i)_{i\in\Delta}\right), \quad \forall\,(x_i)_{i\in\Delta\setminus\Lambda} \in (\boldsymbol{x}_{\rho,i})_{i\in\Delta\setminus\Lambda}\,,$$
$$\text{provided, } (x_{\rho,j})_{j\in\Lambda} = (x'_{\rho,j})_{j\in\Lambda}\ . \quad (5)$$

The box $(\boldsymbol{x}_{\rho\mathsf{v},i})_{i\in\Delta\setminus\Lambda}$ associated with each node $\rho\mathsf{v}$ of the RP $s^{|x'}$ is such that there is a unique node $\rho\mathsf{u}$ in RP s of the $\mathbb{Y}$-MRP f such that $(\boldsymbol{x}_{\rho\mathsf{u},i})_{i\in\Delta\setminus\Lambda} = (\boldsymbol{x}_{\rho\mathsf{v},i})_{i\in\Delta\setminus\Lambda}$ and $x' = (x'_{\rho,j})_{j\in\Lambda} \in (\boldsymbol{x}_{\rho\mathsf{u},j})_{j\in\Lambda}$ and thereby $f^{|x'}_{\rho\mathsf{v}} = f_{\rho\mathsf{u}}$.

Algorithm 11: `Slice`(ρ, Λ, x')

input : ρ, the root node of $\mathbb{Y}$-MRP f with RP s, root box $\boldsymbol{x}_\rho = (\boldsymbol{x}_{\rho,j})_{j\in\Delta} = \boxtimes_{j\in\Delta}[\underline{x}_j, \overline{x}_j] \in \mathbb{IR}^d$,
$\Lambda \subset \Delta$, such that $0 < |\Lambda| < |\Delta| = d$,
Point $x' = (x'_{\rho,j})_{j\in\Lambda} \in (\boldsymbol{x}_{\rho,j})_{j\in\Lambda}$.

output : Change f into $f^{|x'}$ that satisfies (5).

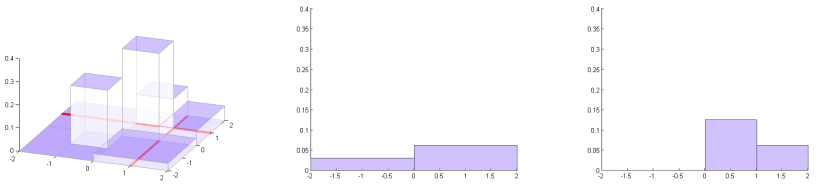
```
if IsLeaf(ρ) Or ι ∉ Λ then
    if !IsLeaf(ρ) then
        Slice(ρL, Λ, x')
        Slice(ρR, Λ, x')
    end
    x_ρ ← x'_ρ = ⊠_{j∈Δ\Λ} [x_j, x̄_j]
end
else
    if x'_ι < mid [x_ι, x̄_ι] then
        Slice(ρL, Λ, x')
        ρ ← Copy(ρL)
    end
    else
        Slice(ρR, Λ, x')
        ρ ← Copy(ρR)
    end
end
```

Algorithm 11 can be used to obtain a slice $\mathbb{Y}$-MRP of a given $\mathbb{Y}$-MRP at a specified subtuple x'. We refer to the notation developed in Section 3 where the index ι is the first coordinate of maximum width of a box $\boldsymbol{x}$ and $\text{mid}\,[\underline{x}_\iota, \overline{x}_\iota]$ is the mid-point of the interval $[\underline{x}_\iota, \overline{x}_\iota]$. Examples 9 and 10 illustrate the slice operation on an $\mathbb{R}$-MRP of $[-2, 2]^2$ and a $[0, 1]^3$-MRP of $[0, 1]^3$, respectively.

Like Algorithm 2, Algorithm 11 relies on our definition of the bisection of a box that gives the left child a box where the interval on coordinate ι is open at the top. (Section 3). Thus x'_ι can never be in both children (if $x'_\iota < \text{mid}\,[\underline{x}_\iota, \overline{x}_\iota]$, x'_ι is considered to be in the left child $\rho\mathsf{L}$, otherwise x'_ι is considered to be in the right child $\rho\mathsf{R}$).

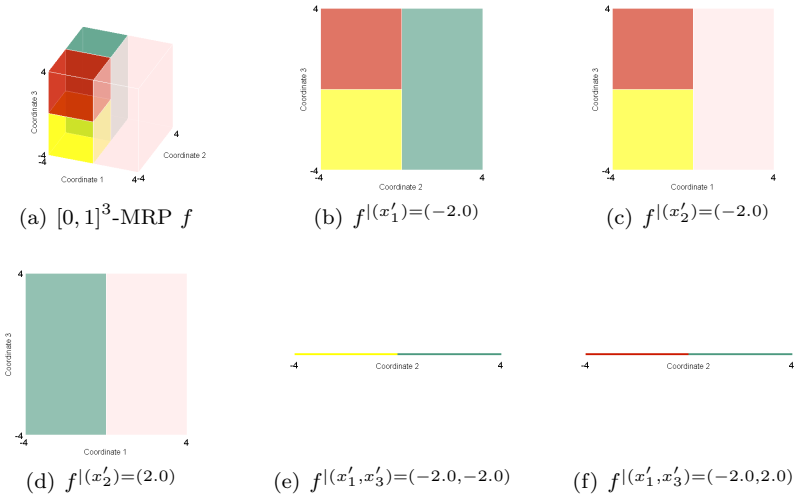
Example 9 (conditional densities)**:** The non-negative real-valued $\mathbb{R}$-MRP f shown in Figure 17(a) is defined over $[-2,2]^2$. Its $\mathbb{R}$-MRP slice orthogonal to the first coordinate at $(x'_1) = (1)$ is $f^{|(x'_1)}$ giving the conditional function $f^{|(x'_1)}(x_2) : [-2,2] \to \mathbb{R}$ shown in Figure 17(b). Similarly, the $\mathbb{R}$-MRP slice of f orthogonal to the second coordinate at $(x'_2) = (1)$ giving the conditional function $f^{|(x'_2)}(x_1) : [-2,2] \to \mathbb{R}$ is $f^{|(x'_2)}$ (Figure 17(c)).

Subsequent renormalisation of the conditional functions so that they integrate to 1 gives us the conditional $\mathbb{R}$-MRP densities from joint $\mathbb{R}$-MRP densities. Obtaining conditional densities and performing operations with them is fundamental in probability and statistics.



(a) $\mathbb{R}$-MRP f in 2-d (b) slice of f at $(x'_1) = (1)$ (c) slice of f at $(x'_2) = (1)$

Figure 17: The slices of a simple $\mathbb{R}$-MRP in 2-d



(a) $[0,1]^3$-MRP f (b) $f^{|(x'_1)=(-2.0)}$ (c) $f^{|(x'_2)=(-2.0)}$

(d) $f^{|(x'_2)=(2.0)}$ (e) $f^{|(x'_1,x'_3)=(-2.0,-2.0)}$ (f) $f^{|(x'_1,x'_3)=(-2.0,2.0)}$

Figure 18: Coloured cube f and its slices.

Example 10 (slices of a coloured cube)**:** The $[0,1]^3$-MRP and its slices in Figure 18 represents a three dimensional unit cube mapped with colour values for red, green and blue in $[0,1]^3$ and its slices at different points. Slices can thus be used to systematically visualize complex or high-dimensional data and subsequently perform visualizable operations with them.

Mapped regular pavings can also be used to extend existing methods of organising and analysing data, for example, for multi-dimensional nonparametric density estimation using histograms [9]. The histogram bins are represented by a data-driven regular paving of $\boldsymbol{x}_\rho \in \mathbb{IR}^d$ where $\boldsymbol{x}_\rho$ is the bounding box of some d-dimensional sample data. The histogram as a whole is represented as a $\mathbb{R}$-MRP where the real value mapped to each bin is the histogram height on that bin. The restrictive form of bisection used for regular pavings does have particular limitations in this context and is not an ideal data-driven binning rule, but for some purposes this disadvantage may be outweighed by the additional operations that can then be performed. For example, we can easily find the difference or error of the histogram against a parametric density approximated as a $\mathbb{R}$-MRP. Averaging of histograms (using addition and then a transformation equivalent to division by a scalar) can be used as a method of density estimation smoothing, as in [9]. Figure 1(c) shows a simple example of histogram averaging for 1-dimensional data. We can also obtain marginal density estimates and conditional density estimates very simply (as far as we know conditional density estimates cannot be as easily made using other available data structures for non-parametric density estimates) using `Marginalise` and `Slice`.

5 Discussion

This project has extended regular pavings that partition a box $\boldsymbol{x}_\rho$ in $\mathbb{IR}^d$ using a binary tree structure to mapped regular pavings which allow us to map the nodes of the regular paving tree to values in a set $\mathbb{Y}$.

By exploiting the algebraic structure formed by regular pavings under union operations we can extend arithmetical expressions over $\mathbb{Y}$ in a point-wise manner over elements of the boxes that partition $\boldsymbol{x}_\rho$. Operations on $\mathbb{Y}$-mapped regular pavings can be carried out very efficiently using recursive algorithms on the finite rooted binary trees that are closed under union operations.

Regular pavings use a restrictive bisection rule that bisects only the midpoint of the first widest coordinate to ensure that the paving union operation is closed. This means that a mapped regular paving may be unsuitable for applications requiring more flexibility in the partitioning of the domain. The compensation for this restriction is the ease with which arithmetic operations can be carried out on regular mapped pavings. The relatively memory-intensive tree structure is used because it also facilitates these operations. Regular mapped pavings have no benefits when the kind of arithmetical operations described in this article are not required. We believe, however, that for some applications the advantages will more than outweigh the disadvantages.

We discuss one such application in detail: the ability to perform arithmetical operations over piecewise constant functions with regular paving partitions of $\boldsymbol{x}_\rho$ using $\mathbb{R}$-mapped regular pavings and even turn it into an inclusion algebra for function ranges with $\mathbb{IR}$-mapped regular pavings. This inclusion algebra is a specialization of the $\mathbb{IR}^*$ algebra PC_k [7, 2.2.2, p. 43] to pieces formed by regular pavings. We have developed novel randomised algorithms that provide memory-efficient enclosures of a real-valued function with a well-defined inclusion function by forming $\mathbb{IR}$-mapped regular pavings using randomised priority queues of their leaf nodes. We show how the range enclosure for a number of leaf nodes can be tightened by making an initial priority queue split to some larger number of leaves and then using the tree structure to propagate the hull of the range enclosures from the leaves up to the root and then using a similar randomised priority queue to prune the tree to the required final size.

When we map Boolean values to regular pavings we obtain the regular subpavings of [2] that can be used to enclose subsets of interest. We also illustrated operations with vector-mapped regular pavings that have potential applications in vector-field arithmetic.

Finally, we provide elementary statistical operations that yield the marginal density, conditional density and highest coverage region of an $\mathbb{R}$-MRP that represents a probability density function. We illustrate how these operations — and others, such as averaging — can be applied in multi-dimensional nonparametric density estimation.

We hope that the elementary statistical operations and arithmetic with $\mathbb{R}$-MRPs will facilitate other statistical applications including nonparametric regression, classification, data exploration and visualisation. The structures and algorithms described in this paper are implemented in *MRS: a C++ class library for statistical set processing* and publicly available under the terms of the GNU General Public License from `http://www.math.canterbury.ac.nz/~r.sainudiin/codes/mrs/`.

Although we have focused here on enclosing functions from $\boldsymbol{x}_\rho \in \mathbb{IR}^d$ to $\mathbb{R}$ with well-defined inclusion functions, we can easily allow for differentiation arithmetic instead of range arithmetic by making further assumptions on the class of functions being enclosed. For instance, we can use $(\mathbb{IR}, \mathbb{IR}^d, \mathbb{IR}^{d\times d})$-MRP to obtain an inclusion algebra for range, gradient and Hessian of twice-differentiable functions from $\boldsymbol{x}_\rho \in \mathbb{IR}^d$ to $\boldsymbol{y} \in \mathbb{IR}$ by using interval extended Hessian differentiation arithmetic. Since the work for combining regular paving based partitions is already complete in this study, we can obtain general $\mathbb{IT}$ inclusion algebras when $\mathbb{Y} = \mathbb{IT}$ and $\mathbb{T}$ are complete lattices, for e.g., when we are interested in arithmetical expressions over enclosures of functions from $\mathbb{R}^m$ to $\mathbb{R}^n$.

The notion of regular paving need not be limited to intervals and boxes. The elementary operation needed to create a regular paving of a set $\boldsymbol{x}_\rho$ associated with node ρ is the bisection operation that produces a left child node $\rho\mathsf{L}$ and a right child node $\rho\mathsf{R}$ such that the sets associated with the child nodes given by $\boldsymbol{x}_{\rho\mathsf{L}}$ and $\boldsymbol{x}_{\rho\mathsf{R}}$ form a bipartition of $\boldsymbol{x}_\rho$, the set associated with the parent node, in a manner that only depends on $\boldsymbol{x}_\rho$. By calling these bisections recursively on the child nodes we can obtain a regular paving. For instance, one could use regular pavings to encode binary triangle trees [1] obtained by bisecting a triangle $\boldsymbol{x}_{\rho\mathsf{v}}$ associated with each node $\rho\mathsf{v}$ into two triangles along the first widest bisector, for instance, and breaking ties at random. In higher dimensions this would give binary d-simplex trees. It would also be interesting to define regular pavings of boxes in more exotic tree spaces [8] in order to obtain an $\mathbb{R}$-MPR histogram of trees over a root box in the tree space.

Acknowledgements

This research was partly supported by RS's external consulting revenues from the New Zealand Ministry of Tourism, WT's Swedish Research Council Grant 2008-7510 that enabled RS's visits to Uppsala in 2006 and 2009, Erskine grant from University of Canterbury that enabled WT's visit to Christchurch in 2011 and University of Canterbury MSc Scholarship to JH. RS is grateful for the discussions with Rick Beatson on Thm. 4.1 and binary triangle trees and with Rua Murray on arithmetic over coarse-grained empirical vector fields.

References

[1] M. Duchaineau, M. Wolinsky, D. E. Sigeti, M. C. Miller, C. Aldrich, and M. B. Mineev-Weinstein. Roaming terrain: Real-time optimally adapting meshes. In *Visualization '97., Proceedings*, pages 81 –88, oct. 1997.

[2] L. Jaulin, M. Kieffer, O. Didrit, and E. Walter. *Applied Interval Analysis with Examples in Parameter and State Estimation, Robust Control and Robotics*. Springer-Verlag, London, April 2001.

[3] M. Kieffer, L. Jaulin, I. Braems, and E. Walter. Guaranteed set computation with subpavings. In W. Kraemer and J. Wolff von Gudenberg, editors, *Scientific Computing, Validated Numerics, Interval Methods, Proceedings of SCAN 2000*, pages 167–178. Kluwer Academic Publishers, 2001.

[4] R. L. Kruse. *Data structures and program design*, chapter 8, pages 273–317. Prentice-Hall, Englewood Cliffs, New Jersey, second edition, 1987.

[5] J. Meier. *Groups, graphs and trees: an introduction to the geometry of infinite groups*. London Mathematical Society student texts. Cambridge University Press, Cambridge, 2008.

[6] R. E. Moore. *Interval analysis*. Prentice-Hall, Englewood Cliffs, New Jersey, 1967.

[7] A. Neumaier. *Interval methods for systems of equations*. Cambridge University Press, Cambridge, England, 1990.

[8] M. Owen and J. S. Provan. A fast algorithm for computing geodesic distances in tree space. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8:2–13, 2011.

[9] R. Sainudiin, G. Teng, J. Harlow, and D. S. Lee. Posterior expectation of regularly paved random histograms. *ACM Transactions on Modeling and Computer Simulation*, October 2012. (accepted for publication).

[10] H. Samet. *The design and analysis of spatial data structures*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1990.

[11] H. Samet. Multidimensional spatial data structures. In Dinesh P. Mehta and Sartaj Sahni, editors, *Handbook of Data Structures and Applications*, chapter 16. Chapman and Hall/CRC, Boca Raton, 2004.

[12] H. Samet. *Foundations of multidimensional and metric data structures*. Morgan Kaufman, San Francisco, 2006.

[13] R. P. Stanley. *Enumerative Combinatorics, Vol. 2*. Cambridge University Press, Cambridge, England, 1999.

[14] G. Teng, K. Kuhn, and R. Sainudiin. Statistical regular pavings to analyze massive data of aircraft trajectories. *Journal of Aerospace Computing, Information and Communication*, 9(1):14–25, 2012.