Representation of Piecewise Linear Interval Functions

Rajat Kateja, Goran Frehse

Verimag Research Report n^o TR-2012-16

October 2, 2012

Representation of Piecewise Linear Interval Functions

Rajat Kateja, Goran Frehse

October 2, 2012

Abstract

In this report, we propose a breakpoint representation of Piecewise Linear Functions and extend it to Piecewise Linear Interval Functions. We also propose algorithms to perform basic operations like scalar multiplication, minimization, pointwise maximum, and intersection using this representation. Finally we consider floating point issues, which arise while using finite precision arithmetic, and suggest methods to get conservative results using interval arithmetic.

Keywords:

Reviewers:

How to cite this report:

```
@techreport {TR-2012-16,
   title = {Representation of Piecewise Linear Interval Functions},
   author = {Rajat Kateja, Goran Frehse},
   institution = {{Verimag} Research Report},
   number = {TR-2012-16},
   year = {}
}
```

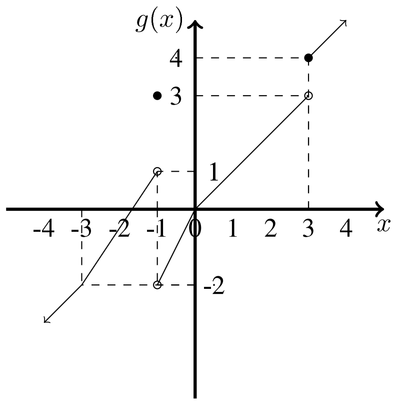

Figure 1: Typical Piecewise Linear Function

1 Piecewise Linear Functions

Let $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$. A *piecewise linear function*, over the domain $I \subseteq \overline{\mathbb{R}}$, $g : I \to \overline{\mathbb{R}}$, is a function which takes a linear form on a finite number of pieces of the domain. Here we define a *breakpoint representation* of such functions. A typical piecewise linear function $g(x)$ is shown in Fig. 1.

Any piecewise linear function can be given in breakpoint form as $g(x) := (I, P, g_l, g_r)$, where

- $I = [I^-, I^+] \subseteq \overline{\mathbb{R}}$ is the interval on which $g(x)$ is defined.
- P is a sequence $p_1, p_2, ...p_n$, with $p_i = (x_i, y_i^-, y_i, y_i^+)$, where

$$x_i \in \mathbb{R} \cap I, y_i = g(x_i), y_i^- = \lim_{x \to x_i; x < x_i} g(x) \; and \; y_i^+ = \lim_{x \to x_i; x > x_i} g(x).$$

  y_i^- is defined only for $x_i > I^-$, and y_i^+ is defined only for $x_i < I^+$.

- $g_l = \frac{dg}{dx} \; \forall \; x < x_1$ defined only if $I^- = -\infty$
- $g_r = \frac{dg}{dx} \; \forall \; x > x_1$ defined only if $I^+ = +\infty$

The meaning of the interval I is clear from its definition. The list P is a list of n points $x_i, i \in \{1, 2, ...n\}$ which divide the domain in pieces, along with the value of the function g and its left and right limits. In the later section on operations with the functions using this representation, it will be shown that this list representation allows easy computation of maximum and minimum of such functions. g_l and g_r store the slope of the left-most and right-most piece of the function (on an unbounded domain). These slopes differentiate between functions which have the same right (or left) end of the leftmost (or rightmost respectively) piece but have different sloping pieces. We require the list of the points to be in strictly increasing order i.e. $\forall k \in \{1, 2, ..., n\}$, $x_{k+1} > x_k$. One point to be kept in mind while using such a representation is that a function can have more than one representation, because we can always introduce a point in the list which does not create a new piece of the function, but is just a point in a particular piece itself. As we will see such ambiguity would not cause any problem in our operations.

Let us represent the function shown in Fig. 1 using the above representation:

- $I = (-\infty, +\infty)$
- P
  - $p_1 = (-3, -2, -2, -2)$
  - $p_2 = (-1, 1, 3, -2)$
  - $p_3 = (0, 0, 0, 0)$

  - $p_4 = (0, 0, 0, 0)$
  - $p_5 = (3, 3, 4, 4)$

- $g_l = 1$
- $g_r = 1$

1.1 Operation on Piecewise Linear Functions

Scalar Multiplication Given a piecewise linear function f and its breakpoint representation (I_f, P_f, f_l, f_r), and $\lambda \in \mathbb{R}$, the breakpoint representation of λf is given by $(I_f, \lambda P_f, \lambda f_l, \lambda f_r)$, with $(x_i, \lambda f^-(x_i), \lambda f(x_i), \lambda f^+(x_i))$ as the entries of the list λP_f.

Minimization Given a piecewise linear function f, $\inf_x f(x)$ is the greatest lower bound on the values that f assumes over all $x \in I_f$. For a piecewise linear function f with bounded domain,

$$\inf_x f(x) = \min_{x \in P_f} [\min(f^-(x), f(x), f^+(x))]$$

where P_f is the list of points in the breakpoint representation of f. The above equality follows from the fact that minimum of a linear function occurs at its end points, so checking for all the end points of the linear pieces, taking the infimum value at the end points i.e. the minimum of the actual value and the left and right limits, and then choosing the minimum among all those values, gives the infimum of f. For a piecewise linear function on bounded domain, we define $\min_x f(x)$, as

$$\min_x f(x) = \min_{x \in P_f} f(x).$$

Here we have considered only the values actually attained by the function. In case of f having an unbounded domain, if $f_l < 0$ or $f_r < 0$, then $\min_x f(x) = \inf_x f(x) = -\infty$. Else, the previous definitons give $\min_x f(x)$ and $\inf_x f(x)$

Pointwise Maximum Given breakpoint representations (I_f, P_f, f_l, f_r) and (I_g, P_g, g_l, g_r) of piecewise linear functions f and g respectively, the pointwise maximum of the two functions is the piecewise linear function

$$h(x) = max(f(x), g(x)) \; \forall \; x \in I_f \cap I_g.$$

Its piecewise representation (I_h, P_h, h_l, h_r) is given by $I_h = I_f \cap I_g$ and P_h, h_l, h_r are given by the following algorithm:

1. Extend the representation of f and g so that P_f and P_g contain the same points $x_i \in I_h \cap (P_f \cup P_g)$. This extension is easy using the given representation. Denote this list of $x_i,\ i \in 1, 2, ..., n$ by X^0.

2. For each $x_i \in X^0$, create the list $Q_h = q_1, q_2, ..., q_n$ with

$$q_i = (x_i, \max(y_i^{f-}, y_i^{g-}), \max(y_i^f, y_i^g), \max(y_i^{f+}, y_i^{g+}))$$

3. Create a list of intersection points $R_h = r_1, r_2, ...$ as follows. Let j=1. For each $x_i \in X^0$,$i < n$, with $(y_i^{f+} - y_i^{g+})(y_{i+1}^{f-} - y_{i+1}^{g-}) < 0$, let $x_i' \in (x_i, x_{i+1})$ be the point of intersection of f and g, and set

$$r_j = (x_i', f(x_i'), f(x_i'), f(x_i')).$$

   Increment j.

4. Merge the two sorted lists Q_h and R_h, to get the sorted list P_h.

5. If $I_h^- = -\infty$
   - Set $h_l = \min(f_l, g_l)$.

- If $(y_1^{f-} - y_1^{g-})(f_l - g_l) > 0$, let $x_0 \in (-\infty, x_1)$ be the intersection point of f and g, insert $p_0 = (x_0, f(x_0), f(x_0), f(x_0))$ as the first entry in the list P_h,

6. If $I_h^+ = +\infty$
   - Set $h_r = \max(f_r, g_r)$.
   - If $(y_n^{f-} - y_n^{g-})(f_l - g_l) < 0$, let $x_{n+1} \in (x_n, +\infty)$ be the intersection point of f and g, insert $p_{n+1} = (x_{n+1}, f(x_{n+1}), f(x_{n+1}), f(x_{n+1}))$ as the last entry in the list P_h.

Explanation of Algorithm First we make a combined list of the points which divide the domain of either f or g, and are also in the domain of h. For each of these points, we compute the maximum function at that point, and its right and left limit. Next, we consider the possibility that the functions f and g might intersect in between the points considered above, dividing the domain further into pieces. So for each interval defined by the adjacent points, we check if one function is above another at one end, and the order is reversed at the other end. If the check returns true, it means the functions intersect in the interval, the intersection point is determined and added to the list of points. The left limit, right limit and the value at the intersection point are all the same because the functions are continuous between any two adjacent points. If the domain of the function is unbounded on the left, we set h_l as the minimum of f_l and g_l, because as $x \to -\infty$, the function with lower slope would assume higher values. Finally we test the possibility of intersection of f and g for some $x \in (-\infty, x_1)$, by checking if either function has higher left limit at x_1, and higher slope before x_1. If the check is true, it implies an intersection point which is determined and added to the list. Similar procedure is used if the domain is unbounded on the right.

2 Piecewise Linear Interval Functions

Let $\mathbb{I}\overline{\mathbb{R}}$ be the set of all intervals in $\overline{\mathbb{R}}$. A *piecewise linear interval function* over $I \subseteq \overline{\mathbb{R}}$ is a function $f : I \to \mathbb{I}\overline{\mathbb{R}}$, such that $\exists\ f^-(x)$ and $f^+(x)$, both piecewise linear functions defined on I, with

$$f(x) = [f^-(x), f^+(x)]\ \forall x \in I.$$

We interpret f as the set of functions bounded pointwise by the lower and upper bound function, i.e.,

$$[\![f]\!] = \{f' : f^-(x) \leqslant f'(x) \leqslant f^+(x)\ \forall\ x \in I\},$$

with all f' defined over the domain I. We denote the graph of f as

$$[\![f]\!]_2 = \{(x, y) : y \in f(x)\},$$

which forms a polygon. An interval function is *connected* if

$$\nexists\ x_1 < x_2 < x_3 \in I,\ such\ that\ f(x_1) \neq \emptyset,\ f(x_2) = \emptyset\ and\ f(x_3) \neq \emptyset.$$

We claim the following holds for piecewise linear interval functions:

1. If $[\![f]\!]_2$ is a connected polygon, then f is connected.
2. If $\exists\ f' \in [\![f]\!]$ such that f' is continuous and f is connected, then $[\![f]\!]_2$ is a connected polygon.

Note that the smallest enclosure of a set of continuous functions may be a disconinuous interval function.

2.1 Operation on Piecewise Linear Interval Functions

Scalar Multiplication Given a piecewise linear interval function $f(x) = [f^-(x), f^+(x)]$ and $\lambda \in \mathbb{R}$,

$$\lambda f(x) = \begin{cases} [\lambda f^-(x), \lambda f^+(x)], & \text{if } \lambda \geqslant 0, \\ [\lambda f^+(x), \lambda f^-(x)], & \text{if } \lambda < 0. \end{cases}$$

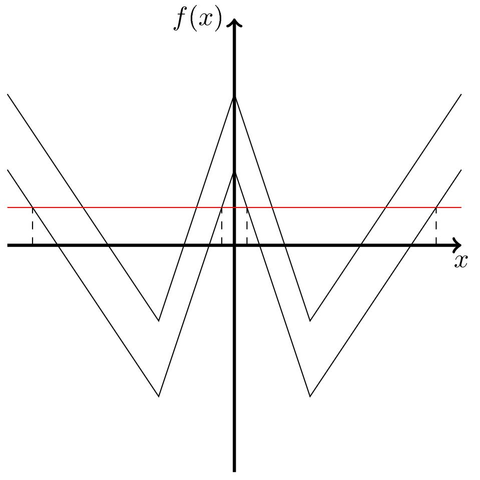

Figure 2: Intersection of f and g, resulting in a disconnected interval function.

Range of Minimum Given an interval function $f = [f^-, f^+]$, define the *range of the minimum* of f as

$$r_{min} := \{\min_x f'(x) : f' \in [\![f]\!]\},$$

where $\min_x f'(x)$ is the minimum of $f'(x)$ over the domain I_f. Now $\forall x \in I_f,\ f' \in [\![f]\!]$

$$\min_x f^-(x) \leqslant f^-(x) \leqslant f'(x)$$

$$\Rightarrow \min_x f^-(x) \leqslant \min_x f'(x).$$

Also $\min_x f^-(x) \in r_{min}$, hence $\min_x f^-(x)$ is the lower bound of r_{min}. Similarly, $\min_x f^+(x)$ is the upper bound of r_{min}. Hence

$$r_{min} = [\min_x f^-(x), \min_x f^+(x)].$$

Intersection Given two piecewise linear interval functions, f and g, their *intersection* is the function h such that

$$[\![h]\!] = [\![f]\!] \cap [\![g]\!] \Leftrightarrow [\![h]\!]_2 = [\![f]\!]_2 \cap [\![g]\!]_2$$
$$\Leftrightarrow h(x) = [max(f^-(x), g^-(x)), min(f^+(x), g^+(x))]$$

with $I_h = I_f \cap I_g$.

The intersection of two connected interval functions may result in a disconnected interval function as illustrated in Fig. 2. There, we have piecewise linear interval functions f and g, with g lowerbounded by $-\infty$ and upperbounded by $c\ \forall x$, and we want to find their intersection. This leaves us with a disconnected interval function with two connected parts.

Obtaining a Set of Connected Interval Functions from an arbitrary Interval Function Given an interval function f, we define

$$Split(f) := \{f_1, f_2, ...f_n\}$$

such that each f_i is a connected interval function on its domain. Note that a interval function f is disconnected iff in certain intervals of the domain $f^+ < f^-$. If f is a piecewise linear interval function on a bounded domain, then $Split(f)$ is computed by the follwing algorithm:

1. Extend the representation of f^- and f^+ so that P_{f^-} and P_{f^+} contain the same points $x_i \in (P_{f^-} \cup P_{f^+})$. This extension is easy using the given representation. Denote this list of $x_i,\ i \in 1, 2, ..., n$ by X^0.

2. Now we will look at consecutive intervals (x_k, x_{k+1}) and for each such interval check if the image of a subinterval is null, i.e. check for an suninterval where $f^+ < f^-$. For each $x_i \in X^0$ check the conditions

$$R_i : \quad If\ x_i < I^+,\ y_i^{f^- -} \leqslant y_i^{f^+ -} \qquad and \tag{1}$$

$$L_{i+1} : \quad If\ x_{i+1} > I^-,\ y_{i+1}^{f^- +} \leqslant y_{i+1}^{f^+ +}. \tag{2}$$

3. Now let a condition being true be representated as $condition$ and being false be represented as $\overline{condition}$. The following sub-routine is to find out the subintervals as explained above. For each i, the following four cases arise:

   - $R_i\ and\ L_{i+1}$: In the entire interval $f^+ \geqslant f^-$ and hence no action is to be performed.
   - $\overline{R_i}\ and\ L_{i+1}$: Create an interval (x_{start}, x_{end}) to be deleted from the domain and set $x_{start} = x_i$ and $x_{end} \in (x_i, x_{i+1})$ to be the intersection point of f^- and f^+.
   - $R_i\ and\ \overline{L_{i+1}}$: Create an interval (x_{start}, x_{end}) to be deleted from the domain and set $x_{start} \in (x_i, x_{i+1})$ to be the intersection point of f^- and f^+ and $x_{end} = x_{i+1}$.
   - $\overline{R_i}\ and\ \overline{L_{i+1}}$: Create an interval (x_{start}, x_{end}) to be deleted from the domain and set $x_{start} = x_i$ and $x_{end} = x_{i+1}$.

4. Remove all the intervals (x_{start}, x_{end}) from the domain I, to get a sequence $I_1, I_2, ...$ of disjoint domains.

5. For each $x_i \in X^0$, if $y_i^{f^-} > y_i^{f^+}$, delete x_i from the domain I_j, with $x_i \in I_j$ (there is exactly one such interval) breaking it into two disjoint domains.

6. Set $f_i = f$ over I_i.

7. Return the set $Split(f) = \{f_1, f_2, ...\}$.

Suppose the given function f has p pieces. From the above algorithm, it is clear that each of the p intervals can give rise to a maximum of one subinterval to be deleted. This implies that a maximum of p intervals would be deleted from the domain, giving rise to maximum of $p + 1$ disjoint domains after step 4. In step 5 each domain can leaad to a maximum of two domains. Hence the cardinality of the set $Split(f)$ cannot exceed $2(p + 1)$.

3 Floating Point Issues

To save memory and gain speed, we use floating point representations. So we have to consider that certain values are not representable. While operating on piecewise linear interval functions, appropriate rounding strategies can guarantee that we always get a conservative result.

For computations, we use interval arithmetic. In the following discussion, every value is to be considered as an interval, for example $x \equiv [\underline{x}, \overline{x}]$, where both $\underline{x}$ and $\overline{x}$ are floating point representable and they are the nearest floating point representable numbers satisfying $\underline{x} \leqslant x \leqslant \overline{x}$. Representable numbers give rise to degenerate intervals with the same lower and upper limit.

We assume that the breakpoint representations of functions f^- and f^+ are given such that they are representable. While computing with these functions, we might end up having arbitrary functions, whose breakpoint representation are not representable. For such situations, we introduce the following notation. For an arbitrary (not necessarily representable) function g with breakpoint representation (I_g, P_g, g_l, g_r), let the breakpoint representation of $\underline{g}$ be $(I_g, \underline{P_g}, \overline{g_l}, \underline{g_r})$, with the list $\underline{P_g} = (x_i, \underline{y_i^-}, \underline{y_i}, \underline{y_i^+})_i$ and the breakpoint representation of $\overline{g}$ be $(I_g, \overline{P_g}, \underline{g_l}, \overline{g_r})$, with the list $\overline{P_g} = (x_i, \overline{y_i^-}, \overline{y_i}, \overline{y_i^+})_i$. Then,

$$g \in [\![[\underline{g}, \overline{g}]]\!].$$

3.1 Scalar Multiplication

For conservative operations, define scalar multiplication of a function f, with a real λ as:

$$\lambda f(x) = \begin{cases} [\underline{\lambda f^-(x)}, \overline{\lambda f^+(x)}], & \text{if } \lambda \geqslant 0, \\ [\underline{\lambda f^+(x)}, \overline{\lambda f^-(x)}], & \text{if } \lambda < 0. \end{cases}$$

3.2 Range of Minimum

Based on the assumption that the given functions f^- and f^+ are representable in floating point, the range of minimum is also representable in floating point, because the range of minimum takes values from the representation itself. Hence there is no need to modify its definition to accommodate floating point issues.

3.3 Extending Breakpoint Representation

While taking the intersection of two functions and while finding the split of a function, we need to extend the representation of certain functions to include more points in their list. While extending lower bound functions, extension must always be done to lower values, and while extending upper bound functions, extension must be done to higher values. For example, if x_{new} is to be added to the list of lower bound function, it must be added as $(x_{new}, \underline{y^-_{x_{new}}}, \underline{y_{x_{new}}}, \underline{y^+_{x_{new}}})$. Where as if x_{new} is to be added to the list of an upper bound function, it must be added as $(x_{new}, \overline{y^-_{x_{new}}}, \overline{y_{x_{new}}}, \overline{y^+_{x_{new}}})$.

3.4 Crossing Test

To find pointwise maximum of two functions (which is used to find intersection), there are checks to see if the two functions cross each other in a particular interval. Once the extension of functions has been done correctly to incorporate floating point issues, these checks can be easily implemented by checking for inequalities. For example to check if f and g cross in the interval (x_1, x_2), we need to check the following condition

$$f(x_1) \geqslant g(x_1) \; and \; f(x_2) \leqslant g(x_2)$$

or vice-versa with f and g interchanged. Since all the four values are representable and the extension was done conservatively, checking for inequalities does not create any complications.

3.5 Intersection Points

There are two procedures for which intersection of piecewise linear functions is required, and both require different types of representations of intersection points to keep the result conservative. Here we discuss them one by one:

Intersection To find the intersection of two pointwise linear interval functions f and g, we need to find the maximum of the two lowerbound functions, and the minimum of the two upperbound functions. Here we describe the procedure for maximum of the two lowerbound functions. Suppose we need to find $h = max(f^-, g^-)$. The only case which might create a problem is when we have to add an extra point i.e. when the lowerbounds intersect in some interval. Now assume that f^- and g^- intersect in the interval (a, b). For simplicity of notation, use $y_a^{f^- +} = m$, $y_a^{g^- +} = p$, $y_b^{f^- -} = n$ and $y_b^{g^- -} = q$, i.e. f^- goes from $A = (a, m)$ to $B = (b, n)$ and g^- goes from $C = (a, p)$ to $D = (b, q)$. Since the functions intersect, we have $(p - m)(q - n) \leqslant 0$. Consider the case with $m \geqslant p$ and $n \leqslant q$. Let the actual point of intersection be $E = (x', y')$. We now explain two methods to add representable point(s) to the representation of h:

- Adding two points: Using the notation as described above, we can represent the two lines as

$$y = m + \frac{n - m}{b - a}(x - a)$$

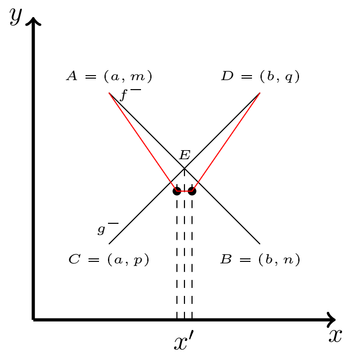

Figure 3: Illustration of conservative intersection to find maximum of two lower bounds using two points

$$y = p + \frac{q-p}{b-a}(x-a).$$

Equating the two y for intersection and solving for x gives

$$x = \frac{a(q-n) + b(m-p)}{m-p+q-n}.$$

Till this stage, all the values $a, b, m, n, p\ and\ q$ are known in their floating point representation. Now interval arithmetic can be used to calculate the above expression and obtain an inteval $[x]$ containing the actual intersection point x'. Next this $[x]$ can be susbtituted in the two line equations to obtain $[y_f]$ and $[y_g]$ from the f and g line equation respectively. Next, define $\underline{y} = max(\underline{y_f}, \underline{y_g})$. Then the points $(\underline{x}, \underline{y}, \underline{y}, \underline{y})$ and $(\overline{x}, \underline{y}, \underline{y}, \underline{y})$ can be added to the representation of h. This method is similar to the one described in [1] where first the lines are parametrised from their end points, then solved for parameters and finally the parameters are plugged back in the equations to get intersection point. Here on the other hand, we directly compute the intersection point. The method is illustrated in Fig. 3.

- Adding a single point: We look for a representable point $F = (\alpha, \beta)$ to add to the representation of h. To remain conservative, the point F must be such that $slope(AE) \geqslant slope(AF)$ and $slope(ED) \leqslant slope(FD)$. Using the above notation, these conditions can be written as

$$\frac{n-m}{b-a} \geqslant \frac{\beta-m}{\alpha-a}$$
$$\frac{q-p}{b-a} \leqslant \frac{q-\beta}{b-\alpha}.$$

Combining the two equations, we get

$$\frac{n-m}{q-p} \geqslant \frac{(\beta-m)(b-\alpha)}{(\alpha-a)(q-\beta)}$$

Using this inequality, for a given representable α, we will get an inequality of the form $\beta \leqslant \beta'$. Then we know that all such points $(\alpha, \underline{\beta'})$ can be added to the representation keeping it conservative. To decide upon a single point from all such available points, we look to minimize the cost function defined as the infinity norm betwen the curve obtained by choosing a point and the original intersection curve. Choice of infinity norm as the cost function is justified because the infinty norm relates to the maximum distance between the two functions. Choosing the maximum of all $\underline{\beta'}$ and adding that point to the representation minimizes the cost function. Next instead of considering all the possible representable numbers α between a and b, we consider only $\underline{x}$ and $\overline{x}$ from $[x]$ as found in above methods because those are the closest points to the actual intersection. Also, the farther we go from the actual solution, the lower would be β and hence the infinity norm would also be more. This method is illustrated in Fig. 4.

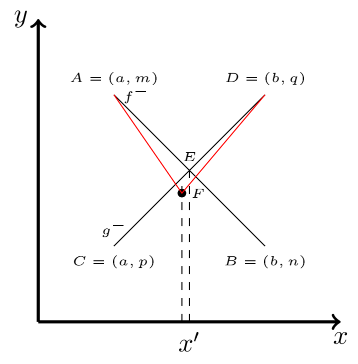

Figure 4: Illustration of conservative intersection to find maximum of two lower bounds using one point

Here are some observations regarding the above methods:

1. The difference in adding just one point or a two points is not very drastic when the space complexities to store the results are considered. Suppose we have two k piece functions, if we add a single point for intersection, we end up with a worst case $2k$ piece function, when each piece has an intersection point. Now suppose this $2k$ piece function is intersected with another $2k$ piece function, we end up with worst case $4k$ piece function, which on further intersection with another $4k$ piece function would result in $8k$ piece function. So after n intersections we end up with worst case $2^n k$ piece function. On the other hand, if we add two intersection points for each intersection, upon the first intersection of two k piece functions, we get a worst case $3k$ piece functions. Advancing as before after n intersections, we would end up with a worst case $3^n k$ piece function. So both the methods have exponential complexities with respect to the number of intersections. The only difference is in the base of the exponential functions.

2. From the discussion on intersections, it is clear that slopes are used a lot in such computations. To calculate slopes from the two given end points, we need interval arithmetic. So it might seem that we can make the computations easier and resolve floating points issues if we represent the functions by one end point and slope of a piece instead of two end points of the piece. Agreed that this approach will resolve some floating point issues in intersection computations, but the downside is as follows. The calculations for the other end point would now involve interval arithmetic, and the end points are required for each piece for crossing tests, whereas the intersection point computations are not always necessary for each piece. So it is better to represent functions by the end points, because they are used more frequently than slopes.

Splitting While splitting an arbitrary piecewise linear interval function f to connected piecewise linear interval functions, we need to find intersection points of f^- and f^+, to break the domain into intervals. We need to identify how we want to store the result as per our requirements. There can be two cases as explained below:

- Here we interpret a conservative result as one in which no part of the domain on which the function is disconnected must be present in the Split, even if some part of the domain where the function was connected is lost. Hence to keep the results conservative, we store the intervals to be deleted, i.e. (x_{start}, x_{end}) as $(\underline{x_{start}}, \overline{x_{end}})$. The reasoning is as follows: Suppose the function is connected on the intervals $(-\infty, x_{start})$ and (x_{end}, ∞), then the interval to be deleted would be (x_{start}, x_{end}). If the interval is made smaller by representing it as $(\overline{x_{start}}, \underline{x_{end}})$, we get that the function f is connected on the intervals $(-\infty, \overline{x_{start}})$ and $(\underline{x_{end}}, \infty)$, while clearly on the intervals $(x_{start}, \overline{x_{start}})$ and $(\underline{x_{end}}, x_{end})$, the functions is not connected. So to get conservative results, we make the intervals to be deleted larger by storing them as $(\underline{x_{start}}, \overline{x_{end}})$.

- In this case, we interpret a conservative result as the one which does not loose any part of the funciton which was connected, even if some part of disconnected function appears in the domain of some function of the Split. To achieve this result, we store the intervals to be deleted as $(\overline{x_{start}}, \underline{x_{end}})$. The explanation is similar to the previous one. Here we can tolerate some part of the domain $((x_{start}, \overline{x_{start}})$ and $((\underline{x_{end}}, x_{end}))$ on which the function is disconnected so that we do not loose the parts of the domain $((\underline{x_{start}}, x_{start})$ and $(x_{end}, \overline{x_{end}}))$ where the function is connected.

References

[1] Marina L. Gavrilova and Jon G. Rokne. Reliable line segment intersection testing. *Computer-Aided Design*, 32(12):737–745, 2000. 3.5