INFORMATICA, 2016, Vol. 27, No. 2, 351–366

DOI: http://dx.doi.org/10.15388/Informatica.2016.89

On Bisecting the Unit Simplex Using Various Distance Norms

Jose M.G. SALMERÓN[1]* Leocadio G. CASADO[1]
Eligius M.T. HENDRIX[2]

[1]*Informatics Department, University of Almería (ceiA3), Spain*
[2]*Department of Computer Architecture, Universidad de Málaga, Spain*
e-mail: josemanuel@ual.es, leo@ual.es, eligius@uma.es

Received: January 2016; accepted: May 2016

Abstract. The iterative bisection of the longest edge of the unit simplex generates a binary tree, where the specific shape depends on the chosen longest edges to be bisected. In global optimization, the use of various distance norms may be advantageous for bounding purposes. The question dealt with in this paper is how the size of a binary tree generated by the refinement process depends on heuristics for longest edge selection when various distance norms are used. We focus on the minimum size of the tree that can be reached, how selection criteria may reduce the size of the tree compared to selecting the first edge, whether a predefined grid is covered and how unique are the selection criteria. The exact numerical values are provided for the unit simplex in 4 and 5-dimensional space.

Key words: simplex, Branch-and-Bound, longest edge bisection, norm, selection heuristics.

1. Introduction

During his scientific career, the interests of Antanas Žilinkas have shown a wide variety. Besides focus on practical problem solving such as Žilinskas and Žilinskas (2013), his handbooks written with Aimo Törn (Törn and Žilinskas, 1989) and Anatoly Zhigljavsky (Zhigljavsky and Žilinskas, 2008) aided the understanding of random function approaches, stochastic processes and branch and bound (B&B). With Jens Claussen (Claussen and Žilinskas, 2002) one of his topics focussed on the use of simplicial partition sets in B&B. The clear advantage of these partition sets is in the use of bounding when the function value of all vertices is taken into account.

This idea is of great interest to our team, as the use of simplicial partition sets is natural when the search region is the unit simplex. This is the case of mixture design problems. We studied these problems with quadratic constraints for practical design of mixtures in the lubricant industry (Hendrix and Pínter, 1991) and for the design of fat blends in food production (Casado *et al*., 2007).

*Corresponding author.

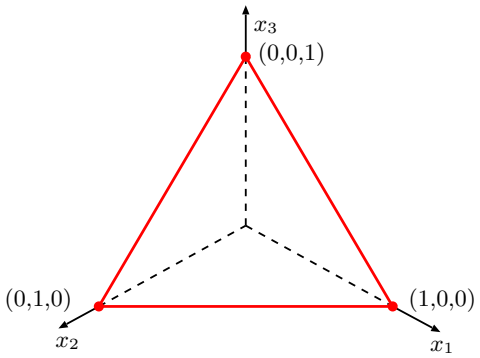

Fig. 1. The unit 2-simplex.

The search region can be described by the standard n-simplex defined in the $(n+1)$-dimensional space (see Fig. 1):

$$S_1 = \left\{ x \in \mathbb{R}^{n+1} \;\; \sum_{j=1}^{n+1} x_j = 1;\; x_j \geqslant 0 \right\}, \tag{1}$$

where x_i represents the contribution of ingredient i in the mixture. In the resulting B&B mostly Longest Edge Bisection (LEB) is applied as this provides relatively round simplices. As discussed by the decision makers at that time, this typically provides solutions with elements that are a power k of 0.5. Considering the worst case, where the complete tree is generated up to a given relative user accuracy $\epsilon = \frac{1}{2^k}$, one may expect the bisection method to generate points on a grid with mesh size ϵ, whereas more points than the grid are over-sampled. This paper shows that not necessarily all grid points are sampled and that the behaviour of this covering among others depends on the chosen norm.

More recently, our research interest goes to the fact that the size of the Binary Tree (BT) generated depends on the longest edges chosen to be bisected in the iterative bisection refinement of the unit simplex for dimension bigger than 3. First of all, heuristics were developed to choose the longest edge and the size of the tree was measured in Aparicio *et al*. (2014) taking as a reference the "first" longest edge as choice rule. Next, a specific algorithm was developed to discover the minimum size of a tree in dimension $n+1$ given an accuracy ϵ in Salmerón *et al*. (2015). Following the line of early research of Adler (1983), Horst (1997), Hendrix *et al*. (2012), we studied bisection of the longest edge in Euclidean norm.

Although not necessary for deriving bounds on quadratic constraints, the bounding in Hendrix and Pínter (1991) and Casado *et al*. (2007) was based on Lipschitzian considerations. Thinking in terms of Euclidean space, the Lipschitz constant links the accuracy δ in function space with a maximum accuracy ϵ in the decision space. This means that one does not have to subdivide partition parts further if their size is less than ϵ making the B&B algorithm a finite process given a certain accuracy. Recently some collaborators of Antanas Žilinskas have shown that one can extend the bounding with Lipschitz constants

to other norms such as the 1-norm and ∞-norm providing sharper bounds, see Paulavičius *et al*. (2011) and the further explanations in the book (Žilinskas and Paulavičius, 2014).

This motivates the question whether the use of other norms is also profitable in the generation of a complete binary tree from iterative refinement of the unit simplex. We pose and investigate the following research questions. What is the effectiveness of heuristics to choose the longest edge in the different distance norms? To investigate this question, we measure the reduction in the size of the tree with respect to the simple rule of selecting the first edge found. Second, is the minimum size tree of the 1-norm and ∞-norm smaller than that of using the Euclidean norm? For this we run an exhaustive algorithm and measure the corresponding tree size. Third, how well does bisection of the longest edge for the 1-norm and ∞-norm cover all points in a grid of a mesh size $\epsilon = \frac{1}{2^k}$? This question is investigated by generating the points on a regular grid and measuring the number that are used as vertices in the bisection process. Last, we will also investigate the uniqueness of the studied selection rules; how many of the longest edges have the same criterion value?

The reporting on the investigation of these research questions in this paper is organised as follows. Section 2 introduces the simplex refinement by LEB, the grid on the unit simplex and the different distance norms. Section 3 introduces the heuristics studied to select the edge to bisect. Section 4 describes the algorithm to obtain the minimum binary tree by simplex refinement using LEB. Section 5 describes the grid covering of the binary tree using LEB. Section 6 shows a numerical evaluation of the bisection process on the unit simplex in the various norms. Finally, conclusions and future research are discussed in Section 7.

2. Simplex Refinement Using Longest Edge Bisection

Consider the unit n-simplex S_1 of (1) to be iteratively bisected, where various distance norms can be used. In general, an n-simplex S is defined by the convex hull $S = conv(V)$ of its vertex set $V = \{v_1, \ldots, v_{n+1}\},\ v_j \in \mathbb{R}^{n+1},\ j = 1, \ldots, n+1$. Let $\omega(S)$ denote the size (width) of a simplex S given by the length of its longest edge. Figure 1 shows a 2-simplex of size $\sqrt{2}$ in Euclidean, size 2 in 1-norm and size 1 in ∞-norm distance.

Longest edge bisection (LEB) is a popular way of iteratively refining a simplex in the context of the finite element method, since it is very simple and can easily be applied in higher dimensions (Hannukainen *et al*., 2014). It is based on splitting a simplex using the hyperplane that connects the mid point of the longest edge of a simplex with the opposite vertices, as illustrated in Fig. 2. Longest edge bisection avoids the generation of needle shape simplices. In this way, the length of an edge in a simplex cannot be greater than two times the length of the other. Using LEB and $\omega(S) \leqslant \epsilon\omega(S_1)$ as termination criterion, the finiteness of the algorithm is assured.

Algorithm 1 describes the Simplex Refinement (SR) process which bisects an initial simplex iteratively. In principle, the refinement can continue infinitely. We study the process with a stopping criterion, i.e. the branching continues until the relative size of the simplex is smaller than or equal to a desired accuracy ϵ.

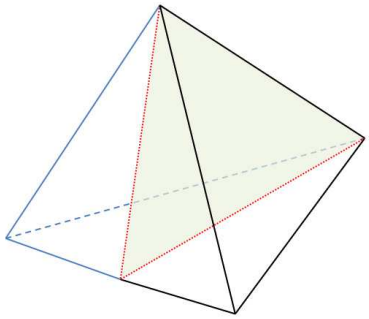

Fig. 2. Longest Edge Bisection (LEB) of the unit 3-simplex.

Algorithm 1 SR(S_1, ϵ)

Require: S_1: initial simplex, $\epsilon = \frac{1}{2^k}$: relative accuracy

$\Lambda := \{S_1\}$ {Set of leaf indices; simplices not yet split}
$ns := 1$ {Number of simplices}
while $\Lambda \neq \emptyset$ **do**
  Extract a simplex S_i from Λ
  if $\omega(S_i) > \epsilon\omega(S_1)$ **then** {Final accuracy not reached}
    $\{j, k\} :=$ SelectLE(S_i) {Select a longest edge}
    $\{S_{2i}, S_{2i+1}\} :=$ Bisect(S_i, j, k) {See Algorithm 2}
    Store simplices S_{2i} and S_{2i+1} in Λ.
    $ns := ns + 2$.
return ns

Figure 3 illustrates the result of the SR algorithm on a 2-simplex S_1 with Euclidean distance and accuracy $\epsilon = 0.5$. The number of levels in the binary tree is 4 and the number of generated simplices from S_1 is 10.

Bisecting a 2-simplex does not require any selection among the longest edges in SelectLE(), as the longest edge is either unique or the choice does not alter the size of the resulting BT (in the cases with regular simplices). Algorithm 2 follows a rule described in Mitchell (1989), where one can avoid edge length calculations in a 2-simplex by always bisecting the edge with vertices $\{1, 2\}$ and numbering the new vertex as the last one of the set of vertices in the generated new sub-simplices.

Figure 2 shows the bisection of a regular 3-simplex. It does not matter which edge is selected first, because all generated sub-simplices differ only in orientation. Notice that after the first sub-division, the generated sub-simplices are irregular and have three (out of six) edges with the longest length. Therefore, we need to make a decision on which longest edge should be bisected. The number of simplices in the finite BT generated by Algorithm 1 depends on how fast the simplex size decreases when we go deeper into the tree. A straightforward implementation selects the first longest edge $\{j, k\}$ found in SelectLE(). The resulting tree may differ if various norms are used. Moreover, applying

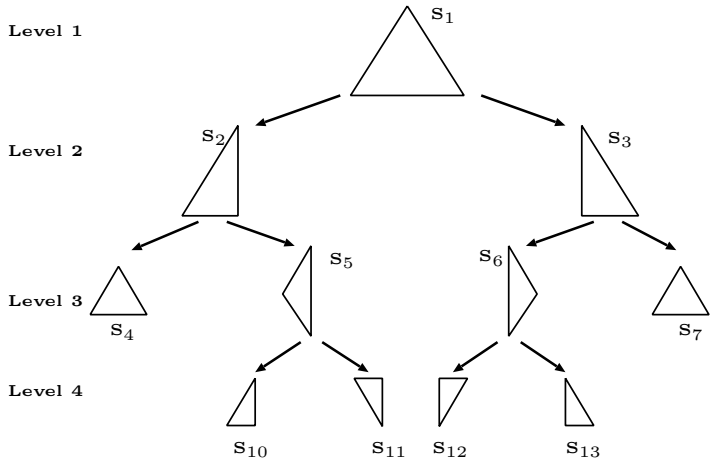

Fig. 3. Binary tree generated by the SR algorithm on unit 2-simplex, $\epsilon = 0.5$.

Algorithm 2 Bisect(S, j, k)

Require: $S = conv(V)$; j, k: vertex indices, $v_j, v_k \in V$ determining an edge

1: Take the vertices v_j and v_k to generate $x := \frac{v_j + v_k}{2}$
2: $V_l := V_r := V$ {New vertex sets V_l and V_r inherit characteristic from the old}
3: Remove v_j from V_l and add x at the end $\{V_l = \{v_1, \ldots, v_{j-1}, v_{j+1}, \ldots, x\}\}$
4: Remove v_k from V_r and add x at the end $\{V_r = \{v_1, \ldots, v_{k-1}, v_{k+1}, \ldots, x\}\}$
5: **return** $S_l = conv(V_l)$, $S_r = conv(V_r)$

selection rules based on a criterion evaluation may reduce the size of the BT with respect to the straightforward implementation. We are interested in how the chosen norm affects the shape of the tree and the effectiveness of chosen selection rules, also called Heuristics for Longest Edge Bisection.

3. Heuristics for Longest Edge Selection

As described before, during the refinement of a regular n-simplex ($n \geqslant 3$) sub-simplices may have several longest edges. A heuristic that computes a criterion value and uses it to select the longest edge may reduce the number of generated sub-simplices and maintains the characteristics of the longest edge bisection. We investigate the effect of various heuristics with respect to the number of generated simplices.

3.1. *First Longest Edge, LEB*$_1$

The straightforward implementation to select a longest edge is to take the first one found. Which is the first one? It depends of course on the coding and storing of the vertices and edges, i.e. the index number assigned to each vertex of the simplex. The new vertex usually

has the same index as the one it substitutes. This rule, which we denote as LEB_1, is used as a benchmark in measuring performance.

3.2. *Midpoint Furthest from the Centroid, LEB_C*

The criterion is the distance of the midpoint of edge $\{v_i, v_j\}$ to the centroid C:

$$\left\| \frac{v_i + v_j}{2} - C \right\|. \tag{2}$$

The LEB_C rule selects the longest edge $\{v_i, v_j\}$ from

$$\arg\max_{i,j} \left\| \frac{v_i + v_j}{2} - C \right\|. \tag{3}$$

3.3. *Largest Distance from Edge Midpoint to the Vertices, LEB_M*

The criterion is the sum of distances of the midpoint of edge $\{v_i, v_j\}$ to the rest of vertices:

$$\sum_{k \neq i,j} \left\| \frac{v_i + v_j}{2} - v_k \right\|. \tag{4}$$

The LEB_M rule selects the longest edge $\{v_i, v_j\}$ from

$$\arg\max_{i,j} \sum_{k \neq i,j} \left\| \frac{v_i + v_j}{2} - v_k \right\|. \tag{5}$$

3.4. *Equal and Maximum Distance Sum to all Neighbours, LEB_N*

For each edge $\{v_i, v_j\}$ the sum of edge lengths from vertex v_i and v_j to the rest of vertices is given by

$$d_i = \sum_{k \neq i} \|v_i - v_k\|, \qquad d_j = \sum_{k \neq j} \|v_j - v_k\|. \tag{6}$$

The LEB_N rule selects the longest edge that gives the optimum of

$$\{v_i, v_j\} \in \arg\max_{i,j} \{d_i + d_j, d_i = d_j\}. \tag{7}$$

In the cases without edges with $d_i = d_j$, LEB_N takes the longest edge $\{v_i, v_j\}$ that maximises $d_i + d_j$.

The uniqueness of maximising a criterion is an interesting topic. In cases where the heuristics give more than one LE to bisect, i.e. the set arg max has more than one element, another criterion can be added, e.g. first select according to LEB_C and within the set

of candidates apply LEB_N. We will investigate possible combinations in future research and focus on the pure effects in this paper. Here, we choose to select the LE with its middle point being a grid point, as a second selection criterion. This can easily be tested by checking that each element is an integer multiple of ϵ.

4. Minimum Tree Size

This section describes an algorithm for determining the size of the smallest binary tree generated using Longest Edge Bisection as the rule for subdividing simplices as published in Salmerón *et al.* (2015). The method applies a full enumeration of simplices checking every division option, i.e. one for each longest edge, and counts the number of sub-simplices in each sub-tree generated from that option. Algorithm 3 performs this task recursively. It has the initial simplex and the required precision as input parameters and returns the number of simplices of the smallest trees.

Algorithm 3 MinTree(S, ϵ)

Require: S: simplex, ϵ: accuracy.
1: **if** $\omega(S) \leqslant \epsilon\omega(S_1)$ **then**
2: **return** 1
3: **for** each longest edge $LE_h = \{j, k\}$ of S **do**
4: $\{S_l, S_r\} := \text{Bisect}(S, j, k)$
5: $r_l := \text{MinTree}(S_l, \epsilon)$ {size of a minimum left sub-tree}
6: $r_r := \text{MinTree}(S_r, \epsilon)$ {size of a minimum right sub-tree}
7: $R_h := r_l + r_r$
8: **return** $1 + \min_h\{R_h\}$

Algorithm 3 determines the size of the smallest sub-tree from a sub-simplex when it is bisected by one of its longest edges LE_h, with vertices $\{j, k\}$ (see line 4). It recursively calls itself to get the smallest sub-tree size for the two generated sub-simplices (see lines 5–6). When the recursive algorithm is back at the initial simplex, the size of the smallest trees is known and the algorithm ends. The algorithm does not provide information about which longest edges have been bisected to generate one of the smallest trees.

Figure 3 illustrates running Algorithm 3 on a 2-simplex for Euclidean distance and $\epsilon = 0.5$. The BT has 10 sub-simplices. Although the 2-simplex is not a very interesting case, because irregular simplices have just one longest edge, the illustration facilitates discussing several details not included in Algorithm 3 for the sake of simplicity:

- In Algorithm 3, if S_l and S_r are symmetric, only one of them is processed and the algorithm returns twice the size of the evaluated sub-tree. In the example of Fig. 3, siblings S_2 and S_3 and also S_{10}, S_{11} and S_{12}, S_{13} are symmetric. However, for our investigation we also would like to find the cover of the grid and therefore cannot apply this observation.

- Algorithm 3 only has to process one of the longest edges for a regular simplex, because any edge division will return the same sub-tree size. Simplices S_1, S_4 and S_7 are regular in Fig. 3.
- For an irregular simplex with several longest edges, it is of interest to determine those edges generating similar pairs of siblings. Therefore, Algorithm 3 only has to process one of the pairs. This will be studied in future work.

5. Covering of the Grid

In Törn and Žilinskas (1989), Antanas introduced the concept of everywhere dense sampling. An equidistant grid is not necessarily the sample with a minimum number of points where each point in the search space has a distance less than a predefined accuracy. However, it is easy to generate and understand. Choosing a certain accuracy $\epsilon = \frac{1}{2^k}$, one would expect iterative bisection of the unit simplex to generate all grid points and over-sample in the sense that it generates additional points. This paper shows that this is not necessarily true. The grid corresponds to $G = \frac{1}{2^k} + 1$ grid points per axis. It is known from Casado *et al.* (2007) that the number of grid points is

$$N_V(G, n) = \sum_{k=1}^{n} \binom{G}{j} \binom{n-1}{j-1}. \tag{8}$$

Two neighbour points x and y on a grid are characterised by the existence of two indices $i \neq j$ such that $(x_1, \ldots, x_{n+1}) = (y_1, \ldots, y_i + \epsilon, \ldots, y_j - \epsilon, \ldots, y_{n+1})$. The distance between two neighbours therefore is $d_1(x, y) = 2\epsilon$ in 1-norm, $d_2(x, y) = \epsilon\sqrt{2}$ in Euclidean norm and $d_\infty(x, y) = \epsilon$ in infinity norm.

Notice that in infinity norm, there are more points than the grid-neighbours defined before at a minimum distance to a grid point. For instance consider $\epsilon = .125$ and $n = 4$. Grid point $x = (0.5, 0, 0.25, 0, 0.25)$ has a neighbour $y = (0.375, 0.125, 0.25, 0, 0.25)$. However, the point $z = (0.375, 0.125, 0.125, 0.125, 0.25)$ is at the same minimum distance of x as the neighbour y. This observation motivates our question whether iterative bisection with the infinity norm will cover all grid points for all dimensions and accuracies.

A refinement of the unit simplex that would cover all $N_V(G, n)$ grid points consists of

$$N_S(G, n) = \binom{G + n - 3}{G - 2}, \tag{9}$$

overlapping regular simplices, see G.-Tóth *et al.* (2016). The number of generated simplices by bisection is much higher. Moreover, we have already questioned the cover of the grid points by bisection. In order to know which vertices from the binary tree are covering the grid points, we will compare the list of grid points with the list of generated vertices by the bisection process. Notice the following:

Remark 1. One grid point can be covered by a vertex of several simplices.

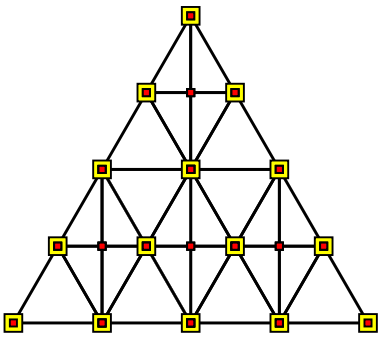

Fig. 4. Covering of grid points by iterative LE bisection, $\epsilon = \frac{1}{4}$.

REMARK 2. A full cover grid does not imply that all vertices are located on the grid. Off-grid vertices can exist. In Fig. 4, yellow boxes represent grid points, whereas red boxes represent generated vertices.

One of the questions is, what is the minimum size tree for the various norms? An observation is that in Fig. 3 after 1 bisection, the next iteration according to the infinite norm may choose from two longest edges, as two edges have a length of 1. Therefore, the hypothesis is that the corresponding minimum size tree will be smaller or equal in number of nodes (simplices) than the corresponding 2-norm tree. This motivates the research question on the size of the minimum tree. The third observation is that if the shape of the tree may differ due to the variety of uniqueness (ambiguity) of the longest edge among distance norms. The heuristics to choose from the longest edges may also differ in effectiveness in obtaining a minimum size tree. This observation motivates the last research question.

6. Numerical Evaluation

The heuristics and norms described in Sections 2 and 3 are evaluated on a regular n-simplex. Different values of $\epsilon = \frac{1}{2^k}$ are used for several dimensions of the problems. When there is more than one longest edge satisfying the criterion related to the heuristic and having its middle point as a grid point, the first longest edge is bisected. The question is what is the number of simplices in the binary tree and how close does it get to the minimum size of the binary tree.

Another issue that we evaluate, directly related to the edge selection and simplex evaluation, is the match of vertices of the simplices with the grid points, as described in Section 2. Moreover, we measure the ambiguity (uniqueness) of the criteria when selecting the edge to bisect in the simplex.

6.1. *3-Simplex*

We first focus on the refinement of the 3-simplex. Table 1 shows the number of simplices of a smallest binary tree (MTREE) and the number of nodes generated by each heuristic for

Table 1
Number of simplices in a 3-simplex refinement, $\epsilon = \frac{1}{2^k}$.

	$k=3$			$k=4$			$k=5$		
	L^2	L^1	L^∞	L^2	L^1	L^∞	L^2	L^1	L^∞
LEB_1	4103	8511	3443	32343	69775	28939	257455	558375	237931
LEB_C	2751	7183	5311	21887	63679	46607	174847	532751	382927
LEB_M	2751	7167	7287	21887	64591	58271	174847	552799	462999
LEB_N	2751	6847	2111	21887	57983	19071	174847	468223	164863
MTREE	2751	6099	1919	21887	52263	16255	174847	422407	127359

Table 2
Number of grid points covered in the 3-simplex refinement.

	$k=3$ (grid = 165)			$k=4$ (grid = 969)			$k=5$ (grid = 6,545)		
	L^2	L^1	L^∞	L^2	L^1	L^∞	L^2	L^1	L^∞
LEB_1	163	163	165	969	915	967	5983	5823	6477
LEB_C	165	157	157	969	913	829	6545	6049	5029
LEB_M	165	157	112	969	881	571	6545	5593	3455
LEB_N	165	157	165	969	873	969	6545	5737	6545
MTREE	165	153	165	969	855	969	6545	5517	6545

Table 3
Euclidean norm. $n=3$, $\epsilon = \frac{1}{32}$ ($k=5$).

NLE	LEB_1			LEB_C			LEB_M			LEB_N		
	Nle	Nle_1	$NleG_1$	*Nle*	Nle_C	$NleG_C$	*Nle*	Nle_M	$NleG_M$	*Nle*	Nle_N	$NleG_N$
1	217986	217986	75252	124996	124996	32768	124996	124996	32768	124996	124996	32768
2	23192	23192	17224	0	0	0	0	0	0	0	0	0
3	80430	80430	120	149550	99450	0	149550	99450	0	149550	99450	0
4	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0
6	6378	6378	0	6	6	0	6	6	0	6	6	0
Sum	543928	543928	110060	573682	423382	32768	573682	423382	32768	573682	423382	32768
Amb	0.60	0.60	0.32	0.78	0.70	0.00	0.78	0.70	0.00	0.78	0.70	0.00

a varying accuracy ϵ. It is interesting to observe that LEB_C, LEB_M and LEB_N achieve the smallest binary tree size in 2-norm bisection. For the other norms, LEB_N is the heuristic that gets closest to the minimum. However, none of the heuristics generates a minimum size tree. Considering the minimum size tree, the size for the 2-norm and ∞-norm contains less nodes (simplices) than the minimum size 1-norm tree.

Table 2 shows the number of covered grid points. LEB_C, LEB_M, LEB_N and the smallest binary tree cover the grid completely in the 2-norm cases. Interestingly, the heuristics of the 1-norm do not cover the grid. The LEB_N heuristic and minimum size tree for the ∞-norm is covering the grid.

The uniqueness of the selection criteria is measured in Tables 3, 4 and 5 for the 2-norm, 1-norm and ∞-norm, respectively.

They indicate the number of longest edges satisfying the criteria, i.e. the size of the arg max set. *Nle* indicates the number of longest edges in simplices with *NLE*

Table 4
1-norm. $n = 3, \epsilon = \frac{1}{32}$ $(k = 5)$.

NLE	LEB_1			LEB_C			LEB_M			LEB_N		
	Nle	Nle_1	$NleG_1$	*Nle*	Nle_C	$NleG_C$	*Nle*	Nle_M	$NleG_M$	*Nle*	Nle_N	$NleG_N$
1	310862	310862	229924	296304	296304	224072	330744	330744	260624	224040	224040	188848
2	278328	278328	158488	295104	177480	136504	277568	175328	154872	225200	138880	81008
3	237546	237546	127064	184968	115480	77552	172656	85512	72472	300960	188368	109184
4	83696	83696	33076	75152	32472	31288	68112	26148	24368	98576	32816	14592
5	29000	29000	13288	35930	16626	9352	32730	24890	23752	30850	6170	1744
6	14658	14658	4840	7590	5062	5056	12870	6014	5616	2694	2694	320
Sum	2147888	2147888	1155876	1967214	1241094	931984	1917166	1203062	1037712	2142038	1245182	747424
Amb	0.86	0.86	0.80	0.85	0.76	0.76	0.83	0.73	0.75	0.90	0.82	0.75

Table 5
∞-norm. $n = 3, \epsilon = \frac{1}{32}$ $(k = 5)$.

NLE	LEB_1			LEB_C			LEB_M			LEB_N		
	Nle	Nle_1	$NleG_1$	*Nle*	Nle_C	$NleG_C$	*Nle*	Nle_M	$NleG_M$	*Nle*	Nle_N	$NleG_N$
1	68898	68898	12992	235624	235624	117168	344944	344944	209678	34712	34712	2560
2	83588	83588	13592	171616	104416	42888	174560	98922	56866	51680	49696	1920
3	144564	144564	29030	113484	53140	17760	74490	30558	13268	96348	65204	10496
4	163912	163912	36158	54368	21480	3248	16744	5484	2988	83712	47040	3840
5	146210	146210	27118	46570	21586	1512	7940	1690	196	226650	45330	12096
6	52986	52986	9192	4566	2870	0	1026	686	0	35622	35622	4352
Sum	2374380	2374380	462640	1397026	814946	276776	1030366	668964	376146	2108946	958258	139840
Amb	0.97	0.97	0.97	0.83	0.71	0.58	0.67	0.48	0.44	0.98	0.96	0.98

longest edges. Nle_a indicates how many of them satisfy the a criterion, i.e. $Nle_a = |\arg\max_{LE} LEB_a(LE)|$ in simplices with NLE longest edges. $NleG_a$ measures how many of them have their middle point on a grid point. The sum row indicates the total number of possible choices for a longest edge where the Amb row then measures the ambiguity as the number of cases where one can select more than one longest edge.

One can observe that the uniqueness of the heuristic selection rule is far larger for the 2-norm. Adding the side criterion to select that edge that has a midpoint on a grid point has a big effect. It reduces the ambiguity to zero, i.e. there is a unique longest edge to choose from. In the 1-norm and ∞-norm there is hardly any effect of adding this side selection criterion.

6.2. *4-Simplex*

Table 6 reports the number of generated simplices by the rules and the smallest size of a binary tree for a 4-Simplex for a varying accuracy ϵ. Interestingly enough, none of the heuristics in the 2-norm generates a minimum size tree. The best heuristic appears to be LEB_M for this norm. For the 1-norm bisection the LEB_C and LEB_N heuristics provide the smallest trees, but do not reach the smallest possible. In the ∞-norm, LEB_N is the heuristic seems to be the most effective. This shows more clearly that the best heuristic depends on the norm used in the bisection process.

Table 7 measures the covering of the grid points for a 4-simplex. For the first time in the experiments, we can observe that the iterative bisection process in the Euclidean

Table 6
Number of simplices in a 4-simplex refinement, $\epsilon = \frac{1}{2^k}$.

	$k=2$			$k=3$			$k=4$		
	L^2	L^1	L^∞	L^2	L^1	L^∞	L^2	L^1	L^∞
LEB_1	6391	18175	4419	100319	358479	80891	1557991	6097639	1415219
LEB_C	4319	14559	8327	71735	299271	170939	1192311	5523847	3024375
LEB_M	4311	20775	12343	71495	399455	217999	1188887	6864703	3583011
LEB_N	5743	15247	3663	90047	303599	64343	1402047	5341039	1121911
MTREE	3639	8311	1271	55623	[1]	[1]	883215	[1]	[1]

[1]The exact number is unknown up to 16 May 2016, where an algorithm has been running on an Intel® Xeon® E5 2650 for 124 days.

Table 7
Number of grid points covered in the 4-simplex refinement.

	$k=2$ (grid = 70)			$k=3$ (grid = 495)			$k=4$ (grid = 4 845)		
	L^2	L^1	L^∞	L^2	L^1	L^∞	L^2	L^1	L^∞
LEB_1	70	70	70	485	485	495	4525	4455	4829
LEB_C	70	58	70	491	381	480	4705	3717	4168
LEB_M	70	52	64	491	301	392	4705	2815	3172
LEB_N	70	70	70	487	463	483	4613	4021	4429
MTREE	70	66	70	491	[1]	[1]	4728	[1]	[1]

[1]The exact number is unknown up to 16 May 2016, where an algorithm has been running on an Intel® Xeon® E5 2650 for 124 days.

Table 8
Euclidean norm. $n=4$, $\epsilon = \frac{1}{8}$ $(k=3)$.

NLE	LEB_1			LEB_C			LEB_M			LEB_N		
	Nle	Nle_1	$NleG_1$	*Nle*	Nle_C	$NleG_C$	*Nle*	Nle_M	$NleG_M$	*Nle*	Nle_N	$NleG_N$
1	75542	75542	18150	45464	45464	7796	45212	45212	7704	64688	64688	15224
2	32528	32528	7764	29096	18756	4060	29120	17696	3420	32032	19248	2400
3	17322	17322	1680	20448	9100	1320	20400	8596	972	14196	5068	720
4	7544	7544	1840	14928	5688	376	14944	5688	372	13888	3576	136
5	1460	1460	556	3340	684	4	3360	672	4	3000	600	0
6	2124	2124	120	1788	894	8	1812	906	8	2508	1238	0
7	994	994	254	1456	560	12	1484	564	12	840	328	0
8	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0
10	650	650	0	10	10	0	10	10	0	10	10	0
Sum	256242	256242	51356	262432	145832	21532	262588	141988	19100	262920	145516	22728
Amb	0.71	0.71	0.65	0.83	0.69	0.64	0.83	0.68	0.60	0.75	0.56	0.33

norm does not necessarily generate all points on an equidistant grid where the accuracy ϵ with a power of .5. Like in the lower dimensional case, the final set of sample points in the 1-norm are of course 2ϵ apart, but do not completely coincide with a grid with the same mesh size. Surprisingly, for $k=2$, the ∞-norm is covering the grid with the LEB_N heuristic and the minimum size tree. This is an interesting result, because the size of the tree is much smaller than in the other norms.

Tables 8, 9 and 10 measure the uniqueness of the selection criteria for the 4-simplex. First of all notice that the number of the longest edges can be much larger. In the 2-norm,

Table 9
1-norm. $n = 4$, $\epsilon = \frac{1}{8}$ $(k = 3)$.

NLE	LEB_1			LEB_C			LEB_M			LEB_N		
	Nle	Nle_1	$NleG_1$	*Nle*	Nle_C	$NleG_C$	*Nle*	Nle_M	$NleG_M$	*Nle*	Nle_N	$NleG_N$
1	204182	204182	143908	158780	158780	99164	268376	268376	186260	176192	176192	133200
2	155900	155900	89906	139248	77724	44304	173240	98536	69060	127216	75640	47112
3	107760	107760	52918	112656	43872	23544	91824	36488	23796	92520	36232	13992
4	87144	87144	32212	77920	23604	11352	36416	11412	7580	73120	25256	10648
5	49910	49910	16404	45260	11556	4096	16920	4524	2612	41120	9024	936
6	30672	30672	8998	20256	5188	1592	5784	1272	732	25536	7232	2472
7	18172	18172	4268	7336	1440	400	1904	352	140	12992	2080	192
8	3664	3664	802	2176	548	272	864	164	72	2656	344	0
9	3870	3870	652	774	230	60	162	34	0	90	30	0
10	630	630	0	10	10	0	10	10	0	10	10	0
Sum	1819066	1819066	789490	1510586	645802	339360	1177094	654994	445096	1472582	643386	332848
Amb	0.89	0.89	0.82	0.89	0.75	0.71	0.77	0.59	0.58	0.88	0.73	0.60

Table 10
∞-norm. $n = 4$, $\epsilon = \frac{1}{8}$ $(k = 3)$.

NLE	LEB_1			LEB_C			LEB_M			LEB_N		
	Nle	Nle_1	$NleG_1$	*Nle*	Nle_C	$NleG_C$	*Nle*	Nle_M	$NleG_M$	*Nle*	Nle_N	$NleG_N$
1	22604	22604	2932	77476	77476	32312	136986	136986	71552	21640	21640	5336
2	25572	25572	2496	83024	48820	16998	92944	51564	22108	23920	17272	3868
3	24360	24360	4148	68520	28596	8666	56286	21330	8590	23484	10536	1952
4	40464	40464	7786	57464	19914	5622	38456	11554	4788	33056	15060	2504
5	34430	34430	7294	37640	9508	1608	21280	4844	1196	24160	5252	984
6	58008	58008	8506	20064	4584	630	7848	1638	540	26928	9828	1480
7	29904	29904	4366	15750	3402	112	2968	620	68	16436	2596	384
8	26704	26704	3174	7696	1684	28	1264	240	6	10624	3580	732
9	20430	20430	1290	5670	1698	6	162	36	0	14274	4758	660
10	8310	8310	308	310	82	0	10	10	0	690	690	12
Sum	1518812	1518812	209662	1213472	468992	127676	831490	391052	170434	889934	329794	57348
Amb	0.99	0.99	0.99	0.94	0.83	0.75	0.84	0.65	0.58	0.98	0.93	0.91

now the additional criterion of being on a grid point reduces the ambiguity of the heuristic, but does not make the choice unique.

7. Conclusions

This paper studied how simplex evaluation and grid generation in iterative longest edge bisection of the unit simplex are influenced by the chosen norm. Several characteristics are investigated, namely the size of the smallest tree, the cover of the grid and the effectiveness of heuristics and rules to choose the LE to be bisected and the ambiguity of used criteria.

Three norms are compared, namely the Euclidean, 1-norm and ∞-norm. Focusing on the minimum size tree that can be generated given a certain accuracy ϵ, we can conclude that the minimum 1-norm tree is bigger than that of the other norms. The Euclidean and ∞-norm generate a smaller binary tree. Interesting is that for the 5-dimensional case, the infinity norm may provide a much smaller search tree. With respect to the ambiguity, in a higher dimension, the ambiguity of the criteria becomes higher, i.e. there are more longest edges to choose from. For the 3-simplex, the choice becomes unique in Euclidean

norm if we force the mid-point of the edge to be bisected to be a grid point. The 1-norm and ∞-norm have more longest edges than the Euclidean norm. This means that in the bisection process there are more longest edges to choose from.

This paper shows that the bisection process that is thought to sample all points on a grid with a mesh size that is an integer power of 0.5 and over-sample it, may not for all choices of the longest edge cover all grid points.

Acknowledgements. This work has been funded by grants from the Spanish Ministry (TIN2015-66680) and Junta de Andalucía (P11-TIC-7176), in part financed by the European Regional Development Fund (ERDF). J.M.G. Salmerón is a fellow of the Spanish FPU program.

References

Adler, A. (1983). On the bisection method for triangles. *Mathematics of Computation*, 40(162), 571–574. doi:10.1090/S0025-5718-1983-0689473-5.

Aparicio, G., Casado, L.G., G-Tóth, B., Hendrix, E.M.T., García, I. (2014). Heuristics to reduce the number of simplices in longest edge bisection refinement of a regular n-simplex. In: *Computational Science and Its Applications, ICCSA 2014. Lecture Notes in Computer Science*, Vol. 8580. Springer International Publishing, pp. 115–125. doi: 10.1007/978-3-319-09129-7_9.

Casado, L.G., García, I., Hendrix, E.M.T. (2007). Infeasibility spheres for finding robust solutions of blending problems with quadratic constraints. *Journal of Global Optimization*, 38(4), 577–593. doi:10.1007/s10898-010-9524-x.

Claussen, J., Žilinskas, A. (2002). Subdivision, sampling and initialization strategies for simplicial branch and bound in global optimization. *Computers and Mathematics with Applications*, 44, 943–955. doi:10.1016/S0898-1221(02)00205-5.

G.-Tóth, B., Hendrix, E.M.T., Casado, L.G., García, I. (2016). On refinement of the unit simplex using regular simplices. *Journal of Global Optimization*, 64(2), 305–323. doi:10.1007/s10898-015-0363-7.

Hannukainen, A., Korotov, S., Křížek, M. (2014). On numerical regularity of the face-to-face longest-edge bisection algorithm for tetrahedral partitions. *Science of Computer Programming*, 90, 34–41. doi:10.1016/j.scico.2013.05.002.

Hendrix, E.M.T., Pínter, J. (1991). An application of Lipschitzian global optimization to product design. *Journal of Global Optimization*, 1, 389–401. doi:10.1007/BF00130833.

Hendrix, E.M.T., Casado, L.G., Amaral, P. (2012). Global optimization simplex bisection revisited based on considerations by Reiner Horst. In: *Computational Science and Its Applications. ICCSA 2012, Lecture Notes in Computer Science*, Vol. 7335. Springer, Berlin, pp. 159–173. doi:10.1007/978-3-642-31137-6_12.

Horst, R. (1997). On generalized bisection of n-simplices. *Mathematics of Computation*, 66(218), 691–698. doi:10.1090/S0025-5718-97-00809-0.

Mitchell, W.F. (1989). A comparison of adaptive refinement techniques for elliptic problems. *ACM Transactions on Mathematical Software*, 15(4), 326–347. doi:10.1145/76909.76912.

Paulavičius, R., Žilinskas, J., Grothey, A. (2011). Parallel branch and bound for global optimization with combination of Lipschitz bounds. *Optimization Methods and Software*, 26(3), 487–498. doi:10.1080/10556788.2010.551537.

Salmerón, J.M.G., Aparicio, G., Casado, L.G., García, I., Hendrix, E.M.T., Tóth, B.G. (2015). Generating a smallest binary tree by proper selection of the longest edges to bisect in a unit simplex refinement. *Journal of Combinatorial Optimization*, 1–14. doi:10.1007/s10878-015-9970-y.

Törn, A., Žilinskas, A. (1989). *Global Optimization. Lecture Notes in Computer Science*, Vol. 350. Springer, Berlin. doi:10.1007/3-540-50871-6.

Zhigljavsky, A., Žilinskas, A. (2008). *Stochastic Global Optimization*. Springer, New York. doi:10.1007/978-0-387-74740-8.

Žilinskas, A., Žilinskas, J. (2013). A hybrid global optimization algorithm for non-linear least squares regression. *Journal of Global Optimization*, 56(2), 265–277. doi:10.1007/s10898-011-9840-9.

Žilinskas, J., Paulavičius, R. (2014). *Simplicial Global Optimization*. Springer, New York. doi:10.1007/978-1-4614-9093-7.

J.M.G. Salmerón is a researcher of Informatics Department at University of Almería, Spain. He is doing his PhD thanks to the Spanish FPU program. His publications and more information can be found in www.hpca.ual.es/~josman. His research interests are high performance computing and global optimization.

L.G. Casado is an associate professor of Informatics Department at University of Almería, Spain from 2001. He obtained his PhD from the University of Málaga. His publications can be found in www.hpca.ual.es/~leo/curr.html. His research interests include global optimization, parallel computing and secure communications.

E.M.T. Hendrix is an European researcher in the field of optimization algorithms. He is affiliated to the universities of Wageningen and Málaga, but also teaches at other universities. His research interests are global and dynamic optimization and computational impacts.

Apie vienetinio simplekso dalijimą taikant įvairias atstumų normas

Jose M.G.SALMERÓN, Leocadio G. CASADO, Eligius M.T. HENDRIX

Iteratyvus ilgiausios kraštinės dalijimas pradedant vienetiniu simpleksu generuoja binarinį medį, kurio forma priklauso nuo dalijimui pasirinktos ilgiausios kraštinės. Globaliajame optimizavime įvairių atstumų normų taikymas gali būti naudingas rėžiams skaičiuoti. Šiame straipsnyje nagrinėjamas klausimas apie tai, kaip dalijimo procesu sugeneruoto binarinio medžio dydis priklauso nuo euristiškai parenkamos ilgiausios kraštinės, kai taikomos įvairios atstumų normos. Mūsų dėmesio centre – mažiausias medžio dydis, kaip parinkimo kriterijai gali sumažinti medžio dydį lyginant su pirmos kraštinės parinkimu, ar numatomas tinklas yra padengiamas, kiek parinkimo kriterijai yra unikalūs. Tikslūs skaičiai yra pateikti vienetiniams simpleksams keturmatėje ir penkiamatėje erdvėje.