On the Exhaustivity of Simplicial Partitioning

Peter J.C. Dickinson*

August 17, 2012

Abstract

During the last 40 years, simplicial partitioning has shown itself to be highly useful, including in the field of Nonlinear Optimisation. In this article, we consider results on the exhaustivity of simplicial partitioning schemes. We consider conjectures on this exhaustivity which seem at first glance to be true (two of which have been stated as true in published articles). However, we will provide counter examples to these conjectures. We also provide a new simplicial partitioning scheme, which provides a lot of freedom, whilst guaranteeing exhaustivity.

Mathematics Subject Classification: 65K99; 90C26
Keywords: Nonconvex Programming; Simplices; Partitioning; Exhaustivity

1 Introduction

Simplicial partitioning has shown itself to be highly useful, including in the field of Nonlinear Optimisation [1, 2, 3, 4, 6, 8, 10].

When we refer to simplices, we mean sets of the form $\Delta = [\mathbf{v}_1, \dots, \mathbf{v}_p]$, where $\mathbf{v}_1, \dots, \mathbf{v}_p \in \mathbb{R}^n$ are affinely independent (equivalently $\dim \Delta = p - 1$) and $[\mathbf{v}_1, \dots, \mathbf{v}_p]$ denotes the convex hull of the set $\{\mathbf{v}_1, \dots, \mathbf{v}_p\}$. Note that if $p = 3$, then this is simply a triangle. We refer to $\{\mathbf{v}_1, \dots, \mathbf{v}_p\}$ as the set of *vertices* of Δ and we refer to $\{\{\mathbf{v}_i, \mathbf{v}_j\} \mid i, j = 1, \dots, p : \ i < j\}$ as the set of *edges* of Δ.

We define the *diameter* of this simplex as follows, where the vector norm used in this paper is the Euclidean norm,

$$\delta(\Delta) := \max\{\|\mathbf{x} - \mathbf{y}\| \mid \mathbf{x}, \mathbf{y} \in \Delta\} = \max\{\|\mathbf{v}_i - \mathbf{v}_j\| \mid i, j = 1, \dots, p\}.$$

If we have a simplex, Δ, and a finite set of simplices, $\mathcal{P} = \{\Delta_i \mid i \in \mathcal{I}\}$, such that

$$\begin{aligned} \Delta &= \bigcup_{i \in \mathcal{I}} \Delta_i, \\ \dim \Delta &= \dim \Delta_i \qquad \text{for all } i \in \mathcal{I}, \\ \emptyset &= \operatorname{reint} \Delta_i \cap \operatorname{reint} \Delta_j \qquad \text{for all } i, j \in \mathcal{I} : \ i \neq j, \end{aligned}$$

then we say that $\mathcal{P}$ is a *simplicial partition* of Δ. In this article, the only partitions that we consider are simplicial partitions and we shall thus simply refer to them as partitions.

*Johann Bernoulli Institute, University of Groningen, P.O. Box 407, 9700 AK Groningen, The Netherlands. Email:P.J.C.Dickinson@rug.nl

We define the diameter of the partition $\mathcal{P}$ given above as

$$\delta(\mathcal{P}) := \max\{\delta(\Delta_i) \mid i \in \mathcal{I}\}.$$

For two partitions $\mathcal{P}_1 = \{\Delta_i \mid i \in \mathcal{I}\}$ and $\mathcal{P}_2 = \{\Delta_j \mid j \in \mathcal{J}\}$, we say that $\mathcal{P}_1$ is nested in $\mathcal{P}_2$ if for all $i \in \mathcal{I}$, there exists $j \in \mathcal{J}$ such that $\Delta_i \subseteq \Delta_j$. Naturally, we then have that $\delta(\mathcal{P}_1) \leq \delta(\mathcal{P}_2)$.

In this article, we consider sequences of nested partitions $\{\mathcal{P}_i \mid i \in \mathbb{N}\}$, where $\mathbb{N} := \{0, 1, 2, \ldots\}$, such that $\mathcal{P}_0 = \{\Delta\}$ and $\Delta = \bigcup_{\Delta_j \in \mathcal{P}_i} \Delta_j$ for all $i \in \mathbb{N}$, and where by a sequence of nested partitions we mean that $\mathcal{P}_{i+1}$ is nested in $\mathcal{P}_i$ for all $i \in \mathbb{N}$. It is then desirable that $\lim_{i\to\infty} \delta(\mathcal{P}_i) = 0$, and when this holds we say that the sequence is *exhaustive*. In the following subsection we consider an application for this, then in Subsection 1.2, we consider three possible methods for going from one partition to the next one in the sequence, and in the remaining sections we look at when we can guarantee that the sequence of partitions is exhaustive, or give a counter example to this happening.

1.1 An application of partitioning

Suppose we wish to solve the following problem, where f is a continuous function.

$$\nu_f(\Delta) = \min_{\mathbf{x}}\{f(\mathbf{x}) \mid \mathbf{x} \in \Delta\}.$$

This is in general an $\mathcal{NP}$-hard problem [5, sections 1.7 and 2.2.3], so instead of solving exactly, we wish to find a hierarchy of approximate solutions to the problem. More specifically, we wish to be able to find a sequence $\{(\mathbf{x}_k, \nu_k) \mid k \in \mathbb{N}\} \subseteq \Delta \times \mathbb{R}$ such that $\nu_k \leq \nu_f(\Delta) \leq f(\mathbf{x}_k)$ for all k and $\lim_{k\to\infty}(f(\mathbf{x}_k) - \nu_k) = 0$.

We begin by defining $U_f(\Delta_i) := \min\{f(\mathbf{v}) \mid \mathbf{v} \text{ is a vertex of } \Delta_i\}$. We then have that $\nu_f(\Delta_i) \leq U_f(\Delta_i)$. Also, from f being continuous, for any sequence of simplices $\{\Delta_i \mid i \in \mathbb{N}\}$, contained within a compact set, such that $\lim_{i\to\infty} \delta(\Delta_i) = 0$, we have that $\lim_{i\to\infty}(U_f(\Delta_i) - \nu_f(\Delta_i)) = 0$.

Now suppose that we can also compute lower bounds to $\nu_f(\Delta_i)$, given by $L_f(\Delta_i)$, such that, for any sequence of simplices $\{\Delta_i \mid i \in \mathbb{N}\}$, contained within a compact set, with $\lim_{i\to\infty} \delta(\Delta_i) = 0$, we have that $\lim_{i\to\infty}(\nu_f(\Delta_i) - L_f(\Delta_i)) = 0$. In Example 1.1, at the end of this subsection, we shall see an example of such a lower bound.

For any partition $\mathcal{P}$, we let

$$U_f(\mathcal{P}) := \min\{U_f(\Delta_i) \mid \Delta_i \in \mathcal{P}\},$$
$$L_f(\mathcal{P}) := \min\{L_f(\Delta_i) \mid \Delta_i \in \mathcal{P}\}.$$

We now consider an arbitrary exhaustive sequence of nested partitions $\{\mathcal{P}_i \mid i \in \mathbb{N}\}$, such that $\mathcal{P}_0 = \{\Delta\}$ and $\Delta = \bigcup_{\Delta_j \in \mathcal{P}_i} \Delta_j$ for all i. We can then let $\mathbf{x}_k$ be one of the vertices minimising $U_f(\mathcal{P}_k)$ and let ν_k equal $L_f(\mathcal{P}_k)$, to get the desired properties for $\{(\mathbf{x}_k, \nu_k) \mid k \in \mathbb{N}\}$.

Example 1.1. Consider $f(\mathbf{x}) = \mathbf{x}^\mathsf{T} A\mathbf{x}$, where A is a symmetric matrix and let

$$L_f(\Delta) = \min\left(\{\mathbf{u}^\mathsf{T} A\mathbf{v} \mid \{\mathbf{u}, \mathbf{v}\} \text{ is an edge of } \Delta\} \cup \{U_f(\Delta)\}\right).$$

First we shall show that this is indeed a lower bound.

Consider an arbitrary $\Delta = [\mathbf{v}_1, \ldots, \mathbf{v}_p]$ such that $\dim \Delta = p - 1$. For any $\mathbf{x} \in \Delta$, there exists a unique $\boldsymbol{\theta} \in \mathbb{R}^p$ such that $\mathbf{x} = \sum_i \theta_i \mathbf{v}_i$, with $\boldsymbol{\theta} \geq \mathbf{0}$ and $\sum_i \theta_i = 1$. We then have the following result, proving that we do indeed have a lower bound

$$f(\mathbf{x}) = \textstyle\sum_{i,j} \theta_i \theta_j \mathbf{v}_i^\mathsf{T} A \mathbf{v}_j \geq \sum_{i,j} \theta_i \theta_j L_f(\Delta) = L_f(\Delta)$$

We shall now prove the convergency result. If we consider arbitrary vertices $\mathbf{u}, \mathbf{v}$ from Δ, then we have the following, where the matrix norm used in this paper is the spectral norm.

$$\begin{aligned} 2\mathbf{u}^\mathsf{T} A \mathbf{v} &= f(\mathbf{u}) + f(\mathbf{v}) - (\mathbf{u} - \mathbf{v})^\mathsf{T} A (\mathbf{u} - \mathbf{v}) \\ &\geq 2U_f(\Delta) - \|A\| \|\mathbf{u} - \mathbf{v}\|^2 \\ &\geq 2U_f(\Delta) - \|A\| \delta(\Delta)^2 . \end{aligned}$$

This then implies that

$$0 \leq \nu_f(\Delta) - L_f(\Delta) \leq U_f(\Delta) - L_f(\Delta) \leq \tfrac{1}{2} \|A\| \delta(\Delta)^2 .$$

Now, considering this for partitions, we have that

$$\begin{aligned} U_f(\mathcal{P}) &= \min\{\mathbf{v}^\mathsf{T} A \mathbf{v} \mid \mathbf{v} \text{ is a vertex in } \mathcal{P}\}, \\ L_f(\mathcal{P}) &= \min\left(\{\mathbf{u}^\mathsf{T} A \mathbf{v} \mid \{\mathbf{u}, \mathbf{v}\} \text{ is an edge in } \mathcal{P}\} \cup \{U_f(\Delta)\}\right). \end{aligned}$$

1.2 Partitioning methods

In order to go from one partition to the next partition in the sequence, we consider three alternative methods, which are given in Algorithms 1 to 3. Good introductions to these are provided in [5]. In this paper, we wish to have as much freedom as possible in our choices. In Algorithms 1 and 2, the choice of what value of α to pick shall always be left open, and similarly for the choice of $\mathbf{w}$ in Algorithm 3. When in Algorithms 1 and 2 an edge $\{\mathbf{u}, \mathbf{v}\}$ is picked, we shall say that we are bisecting this edge, and when in Algorithms 1 and 3 a subsimplex Δ_i is picked then we say that we are partitioning this subsimplex. As previously mentioned, it is often desirable that the sequence of partitions is exhaustive, and we shall look at which algorithms and restrictions on choices ensure this, as well as which ones do not.

Algorithm 1 Classical partitioning of a single subsimplex.

Input: A parameter $\lambda \in (0, \frac{1}{2}]$, and a partition $\{\Delta_i \mid i \in \mathcal{I}\}$.
Output: A partition $\{\Delta_j \mid j \in \mathcal{J}\}$ which is nested in the original partition with $\bigcup_{j \in \mathcal{J}} \Delta_j = \bigcup_{i \in \mathcal{I}} \Delta_i$.

1: **Pick** an edge $\{\mathbf{u}, \mathbf{v}\}$, from a subsimplex Δ_i in the original partition.
2: **Pick** an $\alpha \in [\lambda, 1 - \lambda]$.
3: **Let** $\mathbf{w} = \alpha \mathbf{u} + (1 - \alpha)\mathbf{v}$.
4: **Let** $\{\mathbf{v}_1, \ldots, \mathbf{v}_{p-2}\}$ be the unique set such that $\Delta_i = [\mathbf{u}, \mathbf{v}, \mathbf{v}_1, \ldots, \mathbf{v}_{p-2}]$.
5: **Replace** Δ_i in the partition with two new simplices $\Delta_{i,1}, \Delta_{i,2}$ such that

$$\begin{aligned} \Delta_{i,1} &= [\mathbf{u}, \mathbf{w}, \mathbf{v}_1, \ldots, \mathbf{v}_{p-2}], \\ \Delta_{i,2} &= [\mathbf{w}, \mathbf{v}, \mathbf{v}_1, \ldots, \mathbf{v}_{p-2}]. \end{aligned}$$

6: **Output** resultant partition.

Algorithm 2 Simultaneous partitioning of all subsimplices with a common edge.

Input: A parameter $\lambda \in (0, \frac{1}{2}]$, and a partition $\{\Delta_i \mid i \in \mathcal{I}\}$.
Output: A partition $\{\Delta_j \mid j \in \mathcal{J}\}$ which is nested in the original partition with $\bigcup_{j\in\mathcal{J}} \Delta_j = \bigcup_{i\in\mathcal{I}} \Delta_i$.

1: **Pick** an edge $\{\mathbf{u}, \mathbf{v}\}$, from a subsimplex in the original partition.
2: **Pick** an $\alpha \in [\lambda, 1-\lambda]$.
3: **Let** $\mathbf{w} = \alpha\mathbf{u} + (1-\alpha)\mathbf{v}$.
4: **for** $i \in \mathcal{I}$ such that $\{\mathbf{u}, \mathbf{v}\}$ is an edge of Δ_i **do**
5: **Let** $\{\mathbf{v}_1, \ldots, \mathbf{v}_{p-2}\}$ be the unique set such that $\Delta_i = [\mathbf{u}, \mathbf{v}, \mathbf{v}_1, \ldots, \mathbf{v}_{p-2}]$.
6: **Replace** Δ_i in the partition with two new simplices $\Delta_{i,1}, \Delta_{i,2}$ such that

$$\Delta_{i,1} = [\mathbf{u}, \mathbf{w}, \mathbf{v}_1, \ldots, \mathbf{v}_{p-2}],$$
$$\Delta_{i,2} = [\mathbf{w}, \mathbf{v}, \mathbf{v}_1, \ldots, \mathbf{v}_{p-2}].$$

7: **end for**
8: **Output** resultant partition.

Algorithm 1 is a commonly used method for partitioning [2, 4, 6]. However, from the counter examples in the following section, we shall see that we do not get much freedom in the choice of $\{\mathbf{u}, \mathbf{v}\}$ if we wish to guarantee that the sequence of partitions is exhaustive. This is because with this method the same edge in two different subsimplices should really be considered as two separate edges due to the fact that they are bisected separately. For this reason we introduce Algorithm 2, which does not have this problem, and so, as we will see later, gives more freedom. In fact, in such cases as Example 1.1, this is a natural method to use, as the bounds are dependent on the edges and vertices, rather than the simplices directly.

Algorithm 3 Radial subdivision of a single subsimplex.

Input: A parameter $\rho \in (0, 1]$ and a partition $\mathcal{P} = \{\Delta_i \mid i \in \mathcal{I}\}$.
Output: A partition $\{\Delta_j \mid j \in \mathcal{J}\}$ which is nested in the original partition with $\bigcup_{j\in\mathcal{J}} \Delta_j = \bigcup_{i\in\mathcal{I}} \Delta_i$.

1: **Pick** a subsimplex $\Delta_i = [\mathbf{v}_1, \ldots, \mathbf{v}_p]$ in the original partition.
2: **Pick** a $\mathbf{w} \in \Delta_i$ such that $0 < \|\mathbf{w} - \mathbf{v}_j\| < \rho\delta(\mathcal{P})$ for all j.
3: **Remove** Δ_i from the partition.
4: **for** $j = 1, \ldots, p$ such that $\mathbf{w} \notin [\mathbf{v}_1, \ldots, \mathbf{v}_{j-1}, \mathbf{v}_{j+1}, \ldots, \mathbf{v}_p]$ **do**
5: **Add** the simplex $\Delta = [\mathbf{v}_1, \ldots, \mathbf{v}_{j-1}, \mathbf{w}, \mathbf{v}_{j+1}, \ldots, \mathbf{v}_p]$ to the partition.
6: **end for**

In Algorithm 3 we consider radial subdivisions, also referred to in the literature as ω-subdivisions, which provides another commonly used method for partitionnig [1, 3, 8, 10]. The conditions on $\mathbf{w}$ are in fact a slight adaptation of the ρ-eccentricity condition from [8]. In this we were restricted to $\rho \in (0, 1)$ and $\|\mathbf{w} - \mathbf{v}_j\| < \rho\delta(\Delta_i)$ for all j. We have relaxed the restriction on ρ (although we shall later show that $\rho = 1$ should not be chosen), and added the restriction that $\mathbf{w}$ is not one of the vertices of Δ_i. In general, if ρ is too small, then it is possible that no $\mathbf{w}$ will exist satisfying this condition. However for $\rho \geq \sqrt{3}/2$ there will exist a point obeying this condition. This is because, if we let $\mathbf{w}$ equal the midpoint of the longest edge of Δ_i, then it was shown in [9, 10] that $\|\mathbf{w} - \mathbf{v}\| \leq \frac{1}{2}\sqrt{3}\delta(\Delta_i)$ for all $\mathbf{v} \in \Delta_i$. This can also be seen using Lemma 2.2, introduced in the following section.

Often, when analysing partitions, instead of considering the partition as a whole we consider sequences of subsimplices $\Delta_i \in \mathcal{P}_i$ for all $i \in \mathbb{N}$ such that $\Delta_0 \supseteq \Delta_1 \supseteq \Delta_2 \supseteq \cdots$. Conditions are then included to ensure that for any such sequence we have that $\lim_{i\to\infty} \delta(\Delta_i) = 0$. This would then imply that the sequence of partitions is exhaustive. Although doing this can make the calculations for proving exhaustivity easier, in this paper will shall mainly be considering the partitions as a whole rather than sequences of subsimplices for two reasons. The first is that, in practice, keeping track of all the sequences and ensuring that the conditions hold for them can be computationally cumbersome. Secondly, Algorithm 2 acts in general on multiple subsimplices at once, and thus this type of analysis is insufficient.

2 Counter Examples

One simple choice for $\{\mathbf{u}, \mathbf{v}\}$ in Algorithms 1 and 2 is such that it is one of the longest edges in the partition. If we consider using Algorithm 1, then it was shown in [6] that by doing this for $\lambda = \frac{1}{2}$, we do indeed get that the sequence of partitions is exhaustive, and this was extended in [4] to show that this is also true for any fixed parameter $\lambda \in (0, \frac{1}{2}]$, independent of the choice of α. Now, due to the similarity of Algorithms 1 and 2, we see that this result must also hold for using Algorithm 2. This result is however of limited practical use as it does not give much freedom in the choice of $\{\mathbf{u}, \mathbf{v}\}$. Instead, in many practical applications, $\{\mathbf{u}, \mathbf{v}\}$ is picked in a way that works heuristically well for the application, which we shall refer to as a *free bisection*, then every so often a *controlling bisection* is performed which is meant to ensure the required limiting result [2]. Similarly, Algorithm 3 alone can not guarantee exhaustivity, as in general we are not bisecting edges, and so we perform Algorithm 3 in *free partitions* using a heuristic and every so often a controlling bisection is performed using Algorithm 1 or 2. This is formalised in Algorithm 4. (An alternative method to this for Algorithm 3, connected to a ρ-dominance condition, was given in [8], however this involved keeping tracking of sequences of subsimplices, which we wish to avoid doing.)

Algorithm 4 Partitioning algorithm.

Input: A simplex Δ and $q \in \mathbb{N} \setminus \{0\}$ and $\lambda \in (0, \frac{1}{2}]$.
Output: A nested sequence of partitions of Δ, denoted $\{\mathcal{P}_i \mid i \in \mathbb{N}\}$.

1: **Let** $\mathcal{P}_0 = \{\Delta\}$.
2: **for** $i \in \mathbb{N}$ **do**
3: **if** $i \not\equiv q - 1 \pmod{q}$ **then**
4: **Let** $\mathcal{P}_{i+1}$ be a nested partition of $\mathcal{P}_i$ produced by a free bisection/partition, using Algorithm 1, 2 or 3.
5: **else**
6: **Let** $\mathcal{P}_{i+1}$ be a nested partition of $\mathcal{P}_i$ produced by a controlling bisection, using Algorithm 1 or 2.
7: **end if**
8: **end for**

The controlling bisection is ordinarily in the form of the longest edge being bisected, and it is often stated by having this as the controlling bisection, whilst leaving the choice on the free bisection/partition unrestricted, we will have that the sequence of partitions is exhaustive, where by the free partition being unrestricted we mean that

ρ is set equal to one. However, although it seems obvious that this must be true, in the following example we shall see that this is actually not in general the case.

Example 2.1. This counter example will be for $p = q = 3$, works for Algorithms 1 to 3, and works for every possible controlling bisection scheme, not just that of bisecting the longest edge.

We start with the triangle in Fig. 1a, with all edge lengths equal to one. We shall describe free bisection/partition steps such that after every controlling bisection there must be at least one triangle in the partition which has $\mathbf{t}$ as one of its vertices and its opposite edge being contained in the edge $\{\mathbf{u}, \mathbf{v}\}$. The diameter of the partitions is then greater than or equal to the diameter of this triangle, which is in turn greater than or equal to $\frac{1}{2}\sqrt{3}$.

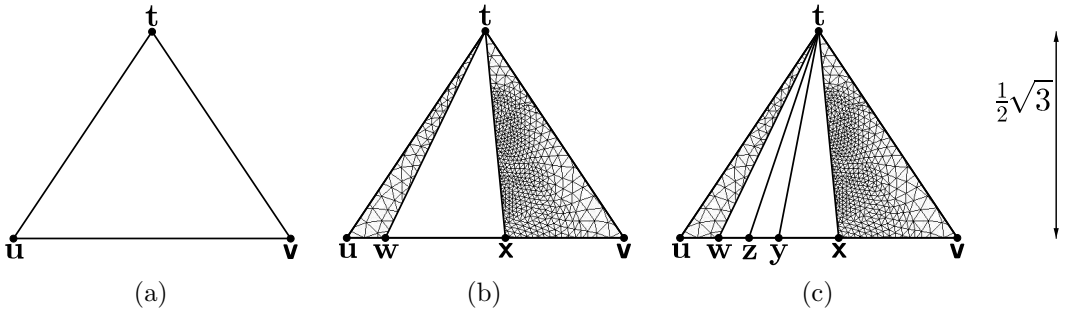


Figure 1: Illustration of Example 2.1.

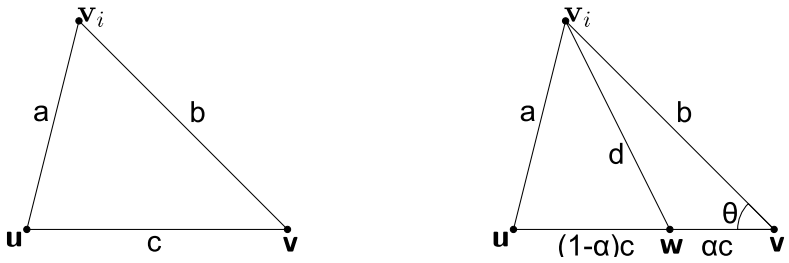
Suppose that after the ith controlling bisection there is such a triangle, for example the triangle $\{\mathbf{t}, \mathbf{w}, \mathbf{x}\}$ in Fig. 1b. The original triangle can be described as the partition after the 0th controlling bisection, so this is true for $i = 0$. Now, for our two free bisection/partition steps we bisect the edge $\{\mathbf{w}, \mathbf{x}\}$ to give the vertex $\mathbf{y}$ and then the edge $\{\mathbf{w}, \mathbf{y}\}$ to give the vertex $\mathbf{z}$ (Fig. 1c). All three of the triangles $\{\mathbf{t}, \mathbf{w}, \mathbf{z}\}$, $\{\mathbf{t}, \mathbf{z}, \mathbf{y}\}$ and $\{\mathbf{t}, \mathbf{y}, \mathbf{x}\}$ are such triangles as described in the previous paragraph. Now, whatever edge is bisected in the next controlling bisection, atleast one of these triangles will be left untouched, and so after the $(i + 1)$th controlling bisection there will be such a triangle as described in the previous paragraph. Thus, by induction, we have constructed a counter example.

From this example, we see that if we wish to guarantee that the sequence of partitions is exhaustive then we need to do something to restrict the choice of free bisections/partitions, and we shall first consider when Algorithm 1 or 2 are used in the free bisections. The reader most likely noticed that, in Example 2.1, the free bisections involved bisecting ever smaller edges. Thus one idea would be that perhaps we could pick a $\delta > 0$ and only bisect edges of length greater than or equal to δ. We would then wish for $\delta(\mathcal{P}_K) \leq \delta$ for some $K \in \mathbb{N}$, at which point we would need to either terminate the algorithm or reduce δ. It would seem that naturally such a thing would occur even without the controlling bisections, however we shall see in Example 2.3 that actually it is possible that without the controlling bisections we may not even get a reduction in $\delta(\mathcal{P}_i)$. Before presenting this example however, we shall first introduce the following lemma, which shall be useful in this example, along with being of use later in the paper.

Lemma 2.2. *Consider the triangle in Fig. 2a, with edge lengths given by a, b, c, as labelled. We partition this triangle as in Algorithm 2 for some $\alpha \in [0, 1]$, to produce*

the partitioned triangle in Fig. 2b. Then we have

$$d^2 = \alpha a^2 + (1-\alpha)b^2 - \alpha(1-\alpha)c^2.$$



(a) Unpartitioned triangle (b) Labelled Triangle

Figure 2: Triangle from Lemma 2.2 before and after partitioning, along with labelling.

Proof. From a standard trigonometric result, called the law of cosines, we get the following, which immediately implies the required result,

$$a^2 = b^2 + c^2 - 2bc\cos\theta \qquad \text{and} \qquad d^2 = b^2 + (\alpha c)^2 - 2b(\alpha c)\cos\theta. \qquad \square$$

We are now ready to present the example.

Example 2.3. This counter example is explained for using Algorithm 2, however it also works for using Algorithm 1, in which case we should pick the subsimplex $[\mathbf{u}, \mathbf{v}_i, \mathbf{w}]$ each time. This counter example is for $p = 3$, with all edge lengths in the initial simplex equal to one. We shall then describe how we can always bisect an edge of length greater than 0.65, but the diameter of the partitions shall remain equal to one.

We start with the triangle in Fig. 3a, with all edge lengths equal to one. For all $i \in \mathbb{N}$, we bisect with $\alpha = \frac{1}{2}$ to produce the new vector $\mathbf{v}_{i+1}$, such that if i is even then we bisect the edge $\{\mathbf{u}, \mathbf{v}_i\}$, and if i is odd then we bisect the edge $\{\mathbf{w}, \mathbf{v}_i\}$. The first few steps of this are demonstrated in Fig. 3.

If we let l_i be the length of the edge bisected to produce vertex $\mathbf{v}_{i+1}$, then using Lemma 2.2, it can be shown that

$$l_0^2 = 1, \qquad l_1^2 = \tfrac{3}{4}, \qquad \text{and} \qquad l_i^2 = \tfrac{1}{2} - \tfrac{1}{4}l_{i-1}^2 + \tfrac{1}{8}l_{i-2}^2 \quad \text{for all } i \geq 2.$$

We can then solve this recursive relation to give the following for all $i \in \mathbb{N}$:

$$l_i^2 = \tfrac{7}{9}\left(\left(-\tfrac{1}{2}\right)^i - \tfrac{1}{7}\right)^2 + \tfrac{3}{7} > \tfrac{3}{7} > (0.65)^2.$$

However, the diameter of the partition remains equal to one as neither of the edges $\{\mathbf{u}, \mathbf{w}\}$ or $\{\mathbf{v}_0, \mathbf{w}\}$ are ever bisected.

So what about combining these two approaches? For this we get partial success. We shall see in the next section that if we use Algorithm 2 in the controlling bisections then even with a slight relaxation to this, the diameter does indeed tend towards zero. However, if instead we use Algorithm 1 in the controlling bisections then the following example acts as a counter example to this approach.

Example 2.4. This counter example is an adaptation of Example 2.3. This is for $p = 4$ with all edge lengths in the initial simplex equal to one. We shall have $q = 2$, where in the controlling bisections we bisect a longest edge using Algorithm 1 and in the

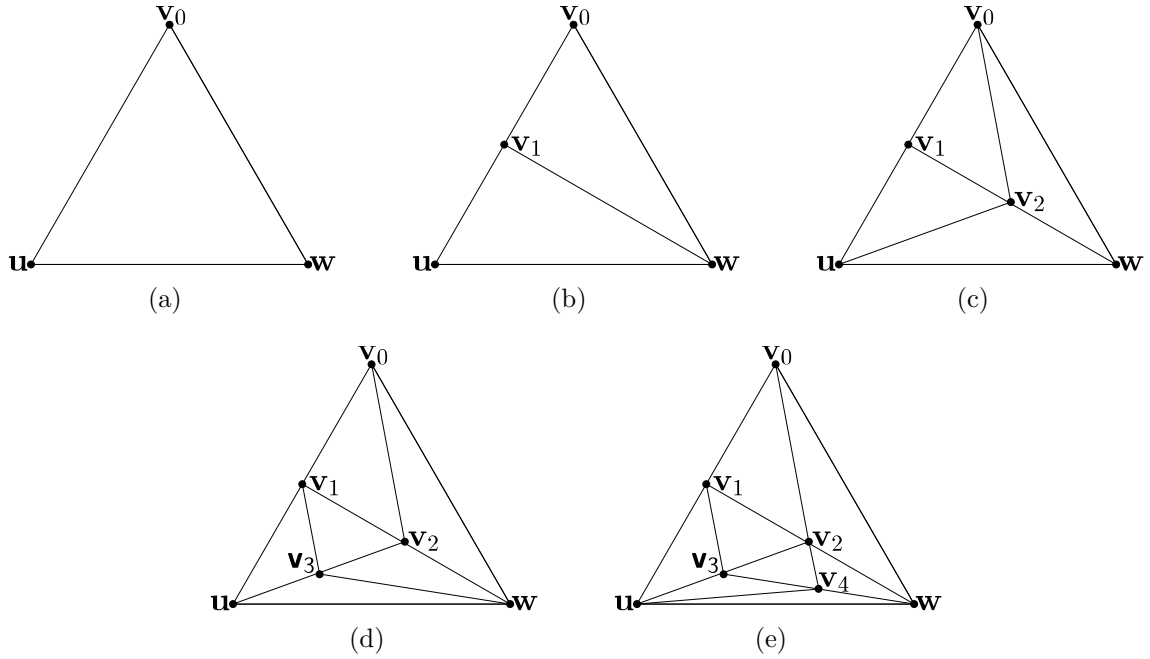


Figure 3: First few steps in Example 2.3.

free bisections we shall bisect an edge of length greater than or equal to 0.65 using Algorithm 1 or 2. However, the diameter of the partitions shall remain equal to one throughout.

We begin with a simplex with four vertices, labelled $\mathbf{t}, \mathbf{u}, \mathbf{v}_0, \mathbf{w}$, and all edge lengths equal to one. We now have the following partitioning rules for $i \in \mathbb{N}$, where all bisections are performed with $\alpha = \frac{1}{2}$:

- $i \equiv 0 \pmod 4$: Pick the edge $\{\mathbf{u}, \mathbf{v}_i\}$, from the subsimplex $[\mathbf{t}, \mathbf{u}, \mathbf{v}_i, \mathbf{w}]$, to bisect using Algorithm 1 or 2 to produce the new vertex $\mathbf{v}_{i+2}$ and the new subsimplices $[\mathbf{t}, \mathbf{u}, \mathbf{v}_{i+2}, \mathbf{w}]$ and $[\mathbf{t}, \mathbf{v}_{i+2}, \mathbf{v}_i, \mathbf{w}]$.
- $i \equiv 1 \pmod 4$: Pick the edge $\{\mathbf{t}, \mathbf{w}\}$, from the subsimplex $[\mathbf{t}, \mathbf{v}_{i+1}, \mathbf{v}_{i-1}, \mathbf{w}]$, to bisect using Algorithm 1.
- $i \equiv 2 \pmod 4$: Pick the edge $\{\mathbf{w}, \mathbf{v}_i\}$, from the subsimplex $[\mathbf{t}, \mathbf{u}, \mathbf{v}_i, \mathbf{w}]$, to bisect using Algorithm 1 or 2 to produce the new vertex $\mathbf{v}_{i+2}$ and the new subsimplices $[\mathbf{t}, \mathbf{u}, \mathbf{v}_{i+2}, \mathbf{w}]$ and $[\mathbf{t}, \mathbf{u}, \mathbf{v}_i, \mathbf{v}_{i+2}]$.
- $i \equiv 3 \pmod 4$: Pick the edge $\{\mathbf{t}, \mathbf{u}\}$, from the subsimplex $[\mathbf{t}, \mathbf{u}, \mathbf{v}_{i-1}, \mathbf{v}_{i+1}]$, to bisect using Algorithm 1.

(Note that only some of the new subsimplices are described, however any excluded are unnecessary for understanding the example.)

For i odd, we are bisecting an edge of length equal to one, i.e. a longest edge, and for i even, if we let l_i be the length of the edge bisected, then similarly to Example 2.3, we have that $l_i^2 = \frac{7}{9}\left(\left(-\frac{1}{2}\right)^{i/2} - \frac{1}{7}\right)^2 + \frac{3}{7} > (0.65)^2$.

We now return to considering Algorithm 3 in the free partitions. From Example 2.1 we have seen that we must pick $\rho \in (0, 1)$. We next consider if simply doing this and using either Algorithm 1 or 2 in the controlling bisections to bisect the longest edge will guarantee that the sequence of partitions is exhaustive. Again we get partial success. In the next section we shall see that if we use Algorithm 2 in the controlling bisections then the sequence of partitions will be exhaustive. However, if instead we

use Algorithm 1 in the controlling bisections then we can not in general guarantee exhaustivity.

Example 2.5. This counter example is for $p = 4$, with all edge lengths in the initial simplex equal to one. We shall have $q = 2$, where in the controlling bisection we use Algorithm 1 to bisect a longest edge and in the free bisections we shall use Algorithm 3 with $\rho = \frac{3}{4}$. However, the diameter of the partitions shall remain equal to one throughout.

We begin with a simplex with four vertices, labelled $\mathbf{t}, \mathbf{u}, \mathbf{v}_0, \mathbf{w}$, and all edge lengths equal to one. We now have the following partitioning rules for $i \in \mathbb{N}$:

- $i \equiv 0 \pmod 2$: Pick the subsimplex $[\mathbf{t}, \mathbf{u}, \mathbf{v}_i, \mathbf{w}]$ to partition using Algorithm 3, producing the new vertex $\mathbf{v}_{i+2} = \frac{1}{4}(\mathbf{t} + \mathbf{u} + \mathbf{v}_i + \mathbf{w})$ and the new subsimplices $[\mathbf{t}, \mathbf{v}_{i+2}, \mathbf{v}_i, \mathbf{w}]$ and $[\mathbf{t}, \mathbf{u}, \mathbf{v}_{i+2}, \mathbf{w}]$, among others.
- $i \equiv 1 \pmod 2$: Pick the subsimplex $[\mathbf{t}, \mathbf{v}_{i+1}, \mathbf{v}_{i-1}, \mathbf{w}]$ and its edge $\{\mathbf{t}, \mathbf{w}\}$ to bisect using Algorithm 1.

For i odd, we are bisecting an edge of length equal to one, i.e. a longest edge. For i even, we are partitioning a subsimplex using Algorithm 3 and for $\mathbf{x} = \mathbf{t}, \mathbf{u}, \mathbf{v}_i, \mathbf{w}$ we have

$$\begin{aligned}\|\mathbf{x} - \mathbf{v}_{i+2}\| &= \|\tfrac{1}{4}(\mathbf{x} - \mathbf{t}) + \tfrac{1}{4}(\mathbf{x} - \mathbf{u}) + \tfrac{1}{4}(\mathbf{x} - \mathbf{v}_i) + \tfrac{1}{4}(\mathbf{x} - \mathbf{w})\| \\ &< \tfrac{1}{4}\|\mathbf{x} - \mathbf{t}\| + \tfrac{1}{4}\|\mathbf{x} - \mathbf{u}\| + \tfrac{1}{4}\|\mathbf{x} - \mathbf{v}_i\| + \tfrac{1}{4}\|\mathbf{x} - \mathbf{w}\| \\ &\leq \tfrac{3}{4}\delta\left([\mathbf{t}, \mathbf{u}, \mathbf{v}_i, \mathbf{w}]\right).\end{aligned}$$

3 A Convergent Partitioning Scheme

In this section we will consider the following two methods of simplex partitioning.

Method 3.1. Let Δ be a simplex, $q \in \mathbb{N} \setminus \{0\}$, $\lambda \in (0, \frac{1}{2}]$ and $\eta \in (0, 1]$. We consider Algorithm 4 for this, where

- In the free bisections, we always bisect an edge of length greater than or equal to $\eta\delta\left(\mathcal{P}_i\right)$ using Algorithm 1 or 2,
- In the controlling bisections, we bisect one of the longest edges using Algorithm 2.

Method 3.2. Let Δ be a simplex, $q \in \mathbb{N} \setminus \{0\}$, $\lambda \in (0, \frac{1}{2}]$ and $\rho \in (0, 1)$. We consider Algorithm 4, where

- In the free partitions, we partition using Algorithm 3,
- In the controlling bisections, we bisect one of the longest edges using Algorithm 2.

We shall prove the following result for these methods.

Theorem 3.3. *For Methods 3.1 and 3.2 we have that the sequence of partitions is exhaustive.*

For $q = 1$ this is already known to be true as in this case we are always bisecting the longest edge. From now on we shall consider $q \geq 2$ and shall define the following, which shall be used throughout this section and the next.

Definition 3.4. Let $\delta \in (0, \delta(\Delta)]$, for Method 3.1 define $\rho := \sqrt{1-\lambda(1-\lambda)\eta^2}$, noting that in such case we have $\rho \in [\frac{1}{2}\sqrt{3}, 1)$, and further define

$$\begin{aligned} p &:= \text{ number of vertices in } \Delta, \\ L &:= \delta(\Delta), \\ \gamma &:= \left\lceil \log_\rho(\delta/L) \right\rceil, \\ K &:= \tfrac{2}{q}(\tfrac{1}{2}pq)^{2^\gamma}. \end{aligned}$$

In this section we shall prove the following lemma, which in turn proves Theorem 3.3. The bound in this lemma is purely there for the purpose of proving that the sequence of partitions is exhaustive, and is not tight.

Lemma 3.5. *For Methods 3.1 and 3.2 with $q \geq 2$, we have $\delta(\mathcal{P}_i) \leq \delta$ for all $i \geq K$.*

In order to prove this lemma, we begin by considering the following lemma on partitioning a triangle.

Lemma 3.6. *Consider the triangle $[\mathbf{u}, \mathbf{v}, \mathbf{v}_i]$, with all edge lengths being less than or equal to $l \in \mathbb{R}$. Also let $\|\mathbf{u}-\mathbf{v}\|$ be greater than or equal to ηl for some $\eta \in (0, 1]$. We partition this triangle as in Algorithm 1 for some $\alpha \in [\lambda, 1-\lambda]$, where $\lambda \in (0, \frac{1}{2}]$, giving the new vertex $\mathbf{w} = \alpha\mathbf{u} + (1-\alpha)\mathbf{v}$ (see Fig. 2b). Then all new edges produced (i.e. edges $\{\mathbf{u}, \mathbf{w}\}$, $\{\mathbf{v}, \mathbf{w}\}$ and $\{\mathbf{v}_i, \mathbf{w}\}$) will have lengths less than or equal to ρl.*

Proof. From Lemma 2.2 we have

$$\begin{aligned} \|\mathbf{v}_i - \mathbf{w}\|^2 &= \alpha\|\mathbf{u}-\mathbf{v}_i\|^2 + (1-\alpha)\|\mathbf{v}-\mathbf{v}_i\|^2 - \alpha(1-\alpha)\|\mathbf{u}-\mathbf{v}\|^2 \\ &\leq l^2 - \lambda(1-\lambda)(\eta l)^2 \\ &= \rho^2 l^2. \end{aligned}$$

We will now complete the proof by showing that $\|\mathbf{u}-\mathbf{w}\|, \|\mathbf{v}-\mathbf{w}\| \leq \rho l$:

$$\|\mathbf{u}-\mathbf{w}\| = (1-\alpha)\|\mathbf{v}-\mathbf{v}_i\| \leq (1-\alpha)l, \qquad \|\mathbf{v}-\mathbf{w}\| = \alpha\|\mathbf{v}-\mathbf{v}_i\| \leq \alpha l,$$
$$\alpha, (1-\alpha) \leq 1-\lambda < \sqrt{(1-\lambda)^2+\lambda} = \sqrt{1-\lambda(1-\lambda)} \leq \sqrt{1-\lambda(1-\lambda)\eta^2} = \rho. \ \square$$

From this we then get the following corollary.

Corollary 3.7. *Let $\mathcal{P}$ be a partition and consider performing Algorithm 1 or 2 on this for $\lambda \in (0, \frac{1}{2}]$, with $\|\mathbf{u}-\mathbf{v}\| \geq \eta\delta(\mathcal{P})$, where $\eta \in (0, 1]$, to produced a nested partition $\widehat{\mathcal{P}}$. Then any new edges produced, i.e. those in $\widehat{\mathcal{P}}$ but not in $\mathcal{P}$, have length less than or equal to $\rho\delta(\mathcal{P})$.*

If we now return to the methods discussed at the beginning of this section we get the following.

Corollary 3.8. *At any step in Methods 3.1 and 3.2, all new edges produces have length less than or equal to $\rho\delta(\mathcal{P}_i)$.*

In order to keep track of how a particular method is doing, we shall label phases in it. We say that at step i the method is in phase j if $\rho^{j+1}L < \delta(\mathcal{P}_i) \leq \rho^j L$. We then have the following lemma.

Lemma 3.9. *$\widehat{p}_j := \frac{2}{q}(\frac{1}{2}pq)^{2^j}$ is an upper bound on the number of vertices at the beginning of phase j (which is also a strict upper bound on the total number of steps from the beginning of the algorithm to the beginning of phase j).*

Proof. We have that $\widehat{p}_0 = p$, therefore the statement is true for $j = 0$. We shall now prove that if it is true for a given j, then it must be true for $j+1$, and so, by induction, we will have proven the required result.

Suppose for the sake of induction, that $\widehat{p}_j$ is an upper bound on the number of vertices at the beginning of phase j. Consider an arbitrary partition $\mathcal{P}$ in phase j. From Corollary 3.8 we get that every new edge produced when bisecting/partitioning this has length less than or equal to $\rho\delta(\mathcal{P}) \leq \rho^{j+1}L$. Therefore the only edges of length strictly greater than $\rho^{j+1}L$ present during this phase must have been there from the start of the phase, and will be the ones bisected during the controlling bisections. The total number of edges at the start of the phase is less than or equal to $\frac{1}{2}\widehat{p}_j(\widehat{p}_j - 1)$, therefore, after at most $\frac{1}{2}\widehat{p}_j(\widehat{p}_j - 1)q$ steps from the start of the phase, we must have bisected all edges of length strictly greater than $\rho^{j+1}L$, and thus have left phase j. Every bisection/partition produces exactly one new vertex, therefore the total number of vertices at the beginning of phase $j+1$ is less than or equal to

$$\widehat{p}_j + \tfrac{1}{2}\widehat{p}_j(\widehat{p}_j - 1)q = \tfrac{1}{2}\widehat{p}_j^2 q + \tfrac{1}{2}\widehat{p}_j(2-q) \leq \tfrac{1}{2}\widehat{p}_j^2 q = \widehat{p}_{j+1}. \qquad \square$$

Finally we note that if $\mathcal{P}$ is the partition at the beginning of phase γ, then $\delta(\mathcal{P}) \leq \rho^\gamma L \leq \delta$. This then completes the proof of Lemma 3.5, which in turn proves Theorem 3.3.

If we wish to consider the dependence on δ/L more explicitly, we can note that

$$\gamma \leq \log_\rho(\delta\rho/L), \qquad 2^\gamma \leq 2^{\log_\rho(\delta\rho/L)} = (\delta\rho/L)^{\log_\rho(2)}.$$

4 Reconsidering unrestricted free partitioning

In this section we shall reconsider the use of unrestricted free bisections. We start with the following corollary of Theorem 3.3.

Corollary 4.1. *Let Δ be a simplex, $q \in \mathbb{N}\setminus\{0\}$ and $\lambda \in (0, \frac{1}{2}]$. Consider Algorithm 4, for this where*

- *In the free bisections we use Algorithm 1 or 2, and we are unrestricted in our choice of edge to bisect,*
- *In the controlling bisection, we bisect one of the longest edges using Algorithm 2.*

Then for any $\delta \in (0, \delta(\Delta)]$, within a finite number of steps we must have bisected an edge of length less than or equal to δ.

We shall demonstrate an application of this theorem in the following example.

Example 4.2. In [2] the authors considered a symmetric nonzero matrix A and the simplex $\Delta^S = \{\mathbf{x} \in \mathbb{R}^n \mid \sum_i(\mathbf{x})_i = 1,\ \mathbf{x} \geq \mathbf{0}\}$, with the aim to check if $\mu \geq 0$, where $\mu := \min\{\mathbf{v}^\mathsf{T}A\mathbf{v} \mid \mathbf{v} \in \Delta^S\}$, and the desire to be able to guarantee completing this check in finite time when $\mu > 0$. We have that $\mu \geq 0$ if and only if $\mathbf{v}^\mathsf{T}A\mathbf{v} \geq 0$ for all nonnegative vectors $\mathbf{v}$, and such a matrix is described as *copositive*, so this check is equivalent to checking copositivity, which is an $\mathcal{NP}$-hard problem [7].

This is in fact closely related to Example 1.1, i.e. $f(\mathbf{x}) = \mathbf{x}^\mathsf{T}A\mathbf{x}$, and we will use the notation given in Subsection 1.1, including that in the example. We recall the following results from this subsection, where $\mathcal{P}$ is a partition of Δ^S:

- $L_f(\mathcal{P}) \leq \mu \leq U_f(\mathcal{P})$,

- For any edge $\{\mathbf{u}, \mathbf{v}\}$ in $\mathcal{P}$ we have that $\frac{1}{2}\|A\|\|\mathbf{u}-\mathbf{v}\|^2 \geq U_f(\mathcal{P}) - \mathbf{u}^\mathsf{T} A\mathbf{v}$,
- $L_f(\mathcal{P}) \geq U_f(\mathcal{P}) - \frac{1}{2}\|A\|\|\delta(\mathcal{P})^2$.

Therefore, if $\mu > 0$ and we have an exhaustive sequence of partitions, then after a finite number of steps we will have $\delta(\mathcal{P}) \leq \sqrt{2\mu/\|A\|}$, implying $L_f(\mathcal{P}) \geq U_f(\mathcal{P}) - \mu \geq 0$, and so the checking would be complete.

An edge $\{\mathbf{u}, \mathbf{v}\}$ in a partition is referred to as active if $\mathbf{u}^\mathsf{T} A\mathbf{v} < 0$ and the authors considered unrestricted free bisections along active edges. They then made the unfortunately common mistake of stating that if we apply Algorithm 4, using Algorithm 1 for all bisections, with the controlling bisections being bisecting a longest edge, then $\delta(\mathcal{P})$ will tend towards zero, which we see from Example 2.4 to not be true in general.

Luckily this can be easily remedied. If instead we apply Algorithm 4, using Algorithm 2 for the controlling bisections, then, for $\mu > 0$, the algorithm would complete in finite time. This is because for any active edge $\{\mathbf{u}, \mathbf{v}\}$ in a partition $\mathcal{P}$, we have

$$\tfrac{1}{2}\|A\|\|\mathbf{u}-\mathbf{v}\|^2 \geq U_f(\mathcal{P}) - \mathbf{u}^\mathsf{T} A\mathbf{v} > U_f(\mathcal{P}) \geq \mu > 0$$

which implies

$$\|\mathbf{u}-\mathbf{v}\| > \sqrt{2\mu/\|A\|}.$$

If we let $\delta = \sqrt{2\mu/\|A\|} > 0$, then, in the free bisections, we are always bisecting active edges of length strictly greater than δ. Now, from Corollary 4.1, we see that if we have a controlling bisection of bisecting one of the longest edges using Algorithm 2, then it is impossible to keep bisecting edges of length greater than δ indefinitely. Therefore, in a finite number of steps we must run out of active edges to bisect, and thus have $L_f(\mathcal{P}) \geq \min\{0, U_f(\mathcal{P})\} \geq 0$, and so the checking would be complete.

5 The importance of picking a fixed λ or ρ

We finish this article with a final example, this time to act as a reminder to the reader of the importance of picking a fixed λ in Algorithms 1 and 2, rather than simply limiting $\alpha \in (0, 1)$, as if this is not done then we can not guarantee exhaustivity. This example can equivalently be seen as emphasising the importance of picking $\rho < 1$ in Algorithm 3.

Rather than considering the partitions as a whole, we shall consider a sequence of nested simplices produced by bisections. Every other step will be a bisection at the midpoint of the longest edge, however the diameter of the simplices will not tend towards zero. Returning to the partitions as a whole, this means that even if ever other step we bisected all subsimplices along one of their longest edges, then we could still not guarantee exhaustivity. This error has previously occurred in published papers, see for example [1, Subsection 5.2].

Example 5.1. Let $\Delta_0 = [\mathbf{v}_{0,1}, \mathbf{v}_{0,2}, \mathbf{v}_3]$ be a simplex with 3 vertices and all edge lengths equal to one. We consider simplices $\Delta_i = [\mathbf{v}_{i,1}, \mathbf{v}_{i,2}, \mathbf{v}_3]$ such that $\Delta_0 \supset \Delta_1 \supset \Delta_2 \supset \cdots$, defined recursively as follows:

- $i \equiv 0 \pmod 2$: Let $\mathbf{v}_{i+1,2} = \mathbf{v}_{i,2}$ and $\mathbf{v}_{i+1,1} = \alpha_i \mathbf{v}_{i,1} + (1-\alpha_i)\mathbf{v}_{i,2}$, where $\alpha_i = \frac{1}{8}\left(\frac{1}{2}\right)^{i/2}$.
- $i \equiv 1 \pmod 2$: Let $\mathbf{v}_{i+1,2} = \mathbf{v}_{i,1}$ and $\mathbf{v}_{i+1,1} = \frac{1}{2}(\mathbf{v}_{i,2} + \mathbf{v}_3)$.

A diagram for two steps of this with i even is shown in Fig. 4.

We shall will show that the following hold for all $i \in \mathbb{N}$:

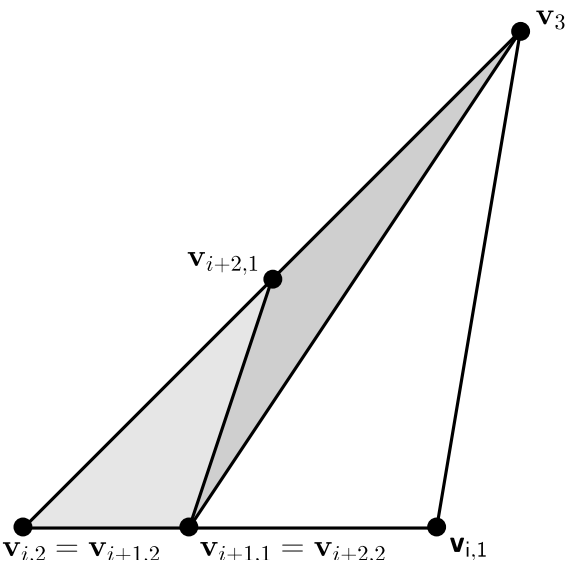


Figure 4: Diagram illustrating Example 5.1, with i even

1. $\{\mathbf{v}_{i,2}, \mathbf{v}_3\}$ is a longest edge of Δ_i, being the unique longest edge for $i > 0$,
2. for i even, all edges of Δ_i have length greater than $\frac{1}{4}\delta(\Delta_i)$,
3. for i even, we have $\|\mathbf{v}_{i,2}, \mathbf{v}_3\| \geq \left(\frac{1}{4}; \frac{1}{2}\right)_{i/2} \geq \left(\frac{1}{4}; \frac{1}{2}\right)_{\infty} > 0.57$, where we recall that $(a;q)_k := \prod_{j=0}^{k-1}(1 - aq^j)$ is the q-Pochhammer symbol.

The significance of these statements is that:

1. for i odd, we are bisecting the unique longest edge at the midpoint,
2. for i even, we are bisecting an edge of length greater than $\frac{1}{4}\delta(\Delta_i)$,
3. The diameters of the simplices are always greater than 0.57.

For $i = 0$, all edges are of length one, so the statements trivially hold. Now, for the sake of induction, assume it is true for all $i \leq 2k$, where $k \in \mathbb{N}$.

By construction, $\{\mathbf{v}_{i,2}, \mathbf{v}_3\}$ is the unique longest edge of Δ_{2k+1} and so the statements hold for all $i \leq 2k+1$. We will now consider the lengths of the edges in Δ_{2k+2} in order to prove each of the statements in turn by induction:

1. We shall first prove that $\{\mathbf{v}_{2k+2,2}, \mathbf{v}_3\}$ is the unique longest edge of Δ_{2k+2}.

$$\begin{aligned}
\|\mathbf{v}_{2k+2,1} - \mathbf{v}_{2k+2,2}\| &= \left\|(\tfrac{1}{2} - \alpha_{2k})(\mathbf{v}_3 - \mathbf{v}_{2k,2}) + \alpha_{2k}(\mathbf{v}_3 - \mathbf{v}_{2k,1})\right\| \\
&< (\tfrac{1}{2} - \alpha_{2k})\|\mathbf{v}_3 - \mathbf{v}_{2k,2}\| + \alpha_{2k}\|\mathbf{v}_3 - \mathbf{v}_{2k,1}\| \\
&\leq \tfrac{1}{2}\|\mathbf{v}_3 - \mathbf{v}_{2k,2}\| \\
&= \|\mathbf{v}_3 - \mathbf{v}_{2k+2,1}\|,
\end{aligned}$$

$$\begin{aligned}
\|\mathbf{v}_3 - \mathbf{v}_{2k+2,1}\| &= \tfrac{1}{2}\|\mathbf{v}_3 - \mathbf{v}_{2k,2}\| \\
&< (1 - 2\alpha_{2k})\|\mathbf{v}_3 - \mathbf{v}_{2k,2}\| \\
&\leq (1 - \alpha_{2k})\|\mathbf{v}_3 - \mathbf{v}_{2k,2}\| - \alpha_{2k}\|\mathbf{v}_3 - \mathbf{v}_{2k,1}\| \\
&< \|(1 - \alpha_{2k})(\mathbf{v}_3 - \mathbf{v}_{2k,2}) + \alpha_{2k}(\mathbf{v}_3 - \mathbf{v}_{2k,1})\| \\
&= \|\mathbf{v}_3 - \mathbf{v}_{2k+2,2}\|.
\end{aligned}$$

2. We now prove that the length of edge $\{\mathbf{v}_{2k+2,1}, \mathbf{v}_{2k+2,2}\}$ (i.e. the shortest edge), is greater than $\frac{1}{4}$ times the length of edge $\{\mathbf{v}_{2k+2,2}, \mathbf{v}_3\}$ (i.e. the longest edge).

$$\begin{aligned}
\|\mathbf{v}_{2k+2,1} - \mathbf{v}_{2k+2,2}\| &= \left\|(\tfrac{1}{2} - \alpha_{2k})(\mathbf{v}_3 - \mathbf{v}_{2k,2}) + \alpha_{2k}(\mathbf{v}_3 - \mathbf{v}_{2k,1})\right\| \\
&> (\tfrac{1}{2} - \alpha_{2k})\|\mathbf{v}_3 - \mathbf{v}_{2k,2}\| - \alpha_{2k}\|\mathbf{v}_3 - \mathbf{v}_{2k,1}\| \\
&\geq (\tfrac{1}{2} - 2\alpha_{2k})\|\mathbf{v}_3 - \mathbf{v}_{2k,2}\| \\
&\geq \tfrac{1}{4}\|\mathbf{v}_3 - \mathbf{v}_{2k,2}\| \\
&> \tfrac{1}{4}\left((1 - \alpha_{2k})\|\mathbf{v}_3 - \mathbf{v}_{2k,2}\| + \alpha_{2k}\|\mathbf{v}_3 - \mathbf{v}_{2k,1}\|\right) \\
&> \tfrac{1}{4}\left\|(1 - \alpha_{2k})(\mathbf{v}_3 - \mathbf{v}_{2k,2}) + \alpha_{2k}(\mathbf{v}_3 - \mathbf{v}_{2k,1})\right\| \\
&= \tfrac{1}{4}\|\mathbf{v}_3 - \mathbf{v}_{2k+2,2}\|.
\end{aligned}$$

3. Finally we prove that the length of edge $\{\mathbf{v}_{2k+2,2}, \mathbf{v}_3\}$ (i.e. the longest edge) is greater than $\left(\frac{1}{4}; \frac{1}{2}\right)_{k+1}$.

$$\begin{aligned}
\|\mathbf{v}_3 - \mathbf{v}_{2k+2,2}\| &= \|(1 - \alpha_{2k})(\mathbf{v}_3 - \mathbf{v}_{2k,2}) + \alpha_{2k}(\mathbf{v}_3 - \mathbf{v}_{2k,1})\| \\
&> (1 - \alpha_{2k})\|\mathbf{v}_3 - \mathbf{v}_{2k,2}\| - \alpha_{2k}\|\mathbf{v}_3 - \mathbf{v}_{2k,1}\| \\
&\geq (1 - 2\alpha_{2k})\|\mathbf{v}_3 - \mathbf{v}_{2k,2}\| \\
&\geq \left(1 - \tfrac{1}{4}\left(\tfrac{1}{2}\right)^k\right)\left(\tfrac{1}{4}; \tfrac{1}{2}\right)_k = \left(\tfrac{1}{4}; \tfrac{1}{2}\right)_{k+1}.
\end{aligned}$$

6 Conclusion

In this article we have seen that, for simplicial partitioning, whether a sequence of partitions is exhaustive or not can often defy our intuition. We have provided counter examples to methods which at first glance we may believe to fulfil this requirement. Motivated by these counter examples, a new method of partitioning was introduced which allows us to guarantee this requirement, whilst still giving us a lot of freedom.

Acknowledgment. The author wishes to thank Mirjam Dür for her helpful comments in the writing of this paper.

References

[1] Immanuel Bomze and Gabriele Eichfelder, *Copositivity detection by difference-of-convex decomposition and ω-subdivision*, To appear in: Math. Programming (2012).

[2] Stefan Bundfuss and Mirjam Dür, *Algorithmic copositivity detection by simplicial partition*, Linear Algebra and its Applications **428** (2008), no. 7, 1511–1523.

[3] Reiner Horst, *An algorithm for nonconvex programming problems*, Mathematical Programming **10** (1976), no. 1, 312321.

[4] _____, *On generalized bisection of n-simplices*, Mathematics of Computation **66** (1997), no. 218, 691–698.

[5] Reiner Horst, Panos M. Pardalos, and Nguyen Van Thoai, *Introduction to global optimization*, Springer, Dec 2000.

[6] Baker Kearfott, *A proof of convergence and an error bound for the method of bisection in r^n*, Mathematics of Computation **32** (1978), no. 144, 1147–1153.

[7] Katta G. Murty and Santosh N. Kabadi, *Some np-complete problems in quadratic and nonlinear programming*, Mathematical Programming **39** (1987), no. 2, 117129.

[8] Hoang Tuy, *Effect of the subdivision strategy on convergence and efficiency of some global optimization algorithms*, Journal of Global Optimization **1** (1991), no. 1, 23–36.

[9] ———, *Normal conical algorithm for concave minimization over polytopes*, Mathematical Programming **51** (1991), no. 1, 229245.

[10] Hoang Tuy and Reiner Horst, *Convergence and restart in branch-and-bound algorithms for global optimization. application to concave minimization and d.c. optimization problems*, Mathematical Programming **41** (1988), no. 1, 161–183.