


Recognition of pieces of arithmetic hyperplanes using the Stern-Brocot tree

Bastien Laboureix^{1*}, Alban Mattei², Jacques-Olivier Lachaud³, Isabelle Debled-Rennesson¹

¹Université de Lorraine, CNRS, LORIA, 54000 Nancy, France
bastien.laboureix@ens-paris-saclay.fr.

²Ecole Normale Supérieure Paris-Saclay, Université Paris-Saclay, France.

³Université Savoie Mont Blanc, CNRS, LAMA, F-73000 Chambéry, France.
 0000-0003-4236-2133.

Abstract

The classical problem of discrete structure recognition is revisited in this paper. We focus on pieces of naive lines and, more generally, naive arithmetic hyperplanes, and present a new approach to recognising these discrete structures based on the Stern-Brocot tree. The algorithm for pieces of lines in dimension 2 proposes an alternative method to the state of the art, [by proposing an efficient algorithm for any set, not necessarily connected. We also give an adaptation, in the connected case, which achieves linear complexity and keeps an incremental behaviour for the segments.](#) While most of the concepts can be generalised to planes in dimension 3 and hyperplanes in higher dimensions, certain points in the management of the descent in the Stern-Brocot tree merit further study. The proposed algorithm calculates separating chords characterising the membership of planes to cones generated by the branch of the Stern-Brocot tree. This generalisation shows the close link between arithmetic hyperplanes and the generalised Stern-Brocot tree, opens up interesting prospects for recognising pieces of arithmetic hyperplanes [but still suffers from certain limitations. However, beyond the recognition problem, which has already been well studied in the field, the main contribution of this article lies in the link it creates between generalisation of the Stern-Brocot tree and the combinatorial structure of arithmetic hyperplanes.](#) Finally, we propose a geometric interpretation of separating chords and an interpretation of plane probing algorithms in the Stern-Brocot tree, showing both the links and the differences with our approach.

Keywords: discrete lines, arithmetic hyperplanes, digital plane recognition, Stern-Brocot tree

1 Introduction

Discrete geometry is concerned with various structures of the digital space \mathbb{Z}^d . This article [is an extension of \[17\] and](#) focuses on discrete lines, introduced in [24] and their generalisation to any dimension d [as](#) arithmetic hyperplanes [1]. Our

problem is to decide, given a finite subset S of \mathbb{Z}^d , whether S is a piece of arithmetic hyperplane and, if so, to compute its minimal parameters.

In dimension 2, the problem of naive discrete segment recognition was first treated from the point of view of symbolic dynamics and the study

of Sturmian words: among others, see [14] for the initial article, [3] for a review of the links between discrete geometry and symbolic dynamics, [13] for a resolution of this problem via desubstitutions. A history of the discrete segment recognition problem can be found in [16]. In the nineties, automata were a popular way of recognising if a chaincode was a discrete line segment (e.g. [11] and the comprehensive historical notes in [6]). The arithmetic definition of a discrete line was introduced by J.-P. Réveillès in 1991 in [24]. This arithmetic definition and the geometric structure of the segments were used to obtain the incremental and linear discrete line recognition algorithm presented by I. Debled-Rennesson and J.-P. Réveillès in [10].

While the hyperplane recognition problem is largely solved for segments in dimension 2, the problem remains difficult in dimensions 3 and higher. In dimension 3, numerous studies of discrete planes have been carried out using different approaches (see the overview [4]). A generalisation of I. Debled-Rennesson's 2D algorithm was presented in 1994 in [12, 23], but the algorithm, restricted to pieces of rectangular planes, loses simplicity and the number of cases to process rapidly explodes. In 2002, L. Buzer presented in PhD thesis [7], in fixed dimension, a linear method for recognising a piece of arithmetic hyperplane, without however returning the minimal parameters. In 2005, the algorithm of Y. Gerard et al. [15] also solves the recognition problem for any finite set of points and in any dimension, using the properties of the convex hull of the chords space. However, the algorithm announces a high complexity for dimension 3 and does not guarantee to obtain the minimal parameters. In 2007, in PhD thesis [13], T. Fernique proposed an infinite hyperplane recognition algorithm using word combinatorics and the notion of desubstitution. In 2008, the COBA algorithm ([8]) proposed a linear optimisation approach to the problem. This method provides a quasi-linear algorithm, but does not provide the minimal parameters of the hyperplane. In a different way, a series of articles has been written on plane probing ([19] for the first version), which analyses the local linear geometry of a digital set known only through an oracle. While it returns the exact characteristics on an infinite plane, it does not decide if the set is a piece of hyperplane but rather outputs a

kind of best local linear approximation. Finally, in 2022, S. Barbieri and S. Labbé proposed in [2] a characterisation of Sturmian configurations in any dimension, an approach to arithmetic hyperplanes using word combinatorics.

In dimension 2, we propose another algorithm for recognising a piece of discrete line based on the Stern-Brocot tree (introduced in [25] and [5]). By taking into account the chords of the piece, we can then go down the tree until we find the slope corresponding to its minimal parameters. While I. Debled-Rennesson's segment recognition algorithm can be interpreted as a descent into the tree (see [9] and [26]), the algorithm proposed in this article differs. It works indeed for any pieces of line, not necessarily connected. For line segments (i.e. a connected piece of discrete line), it can be improved to reach the optimal linear complexity while retaining its incremental feature.

This approach to recognition by a descent in the Stern-Brocot tree can also be extended to higher dimensions, by noticing that H. Lennerstad generalises the Stern-Brocot tree to any finite dimension d [21]. We introduce here the notion of separating chord, which is used to determine the branch of descent in the tree, corresponding to a location of the normal vector of the plane to be recognised. Each step of the algorithm now requires $\binom{d}{2}$ tests to determine in which of the $d!$ branches to continue the search. The proposed method recognises discrete hyperplanes and opens up interesting prospects for recognising pieces of discrete planes. We then propose a geometric interpretation of the separating chords and a combinatorial study of them. [The separating chords approach unfortunately suffers from certain limitations, the main one being the detection of these chords, particularly in very small pieces of plane. We describe the problems encountered by this approach at the end of section 5 and present counter-examples to the good behaviour of the algorithm in pathological cases.](#) Finally, we show how to interpret plane probing algorithms, and in particular the H -algorithm [18], as a splitting of the Stern-Brocot tree.

2 Discrete lines

We are working in \mathbb{R}^d with its canonical scalar product $\langle \cdot, \cdot \rangle$, and in this section we take $d = 2$. In 1991, J.-P. Réveillès defined the concept of a discrete line in [24]. We are interested here in a version of the naive discrete line where the thickness parameter is fixed to guarantee good combinatorial and topological properties:

Definition 2.1 (Naive discrete line (Figure 1)). *Let $a, b, \mu \in \mathbb{R}$ with $(a, b) \neq (0, 0)$. The naive discrete line with slope $\frac{a}{b}$ (we agree that a slope $\frac{1}{0}$ is a vertical slope) and shift μ is the set $\mathcal{D}(a, b, \mu) \stackrel{\text{def}}{=} \{(x, y) \in \mathbb{Z}^2 \mid 0 \leq ax - by + \mu < \|(a, b)\|_\infty\}$.*

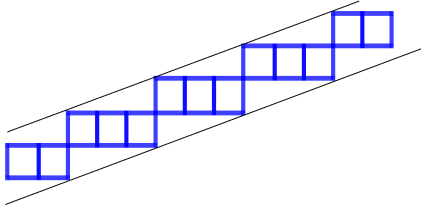


Figure 1 A naive line of parameters $(4, 11, 7)$ and the support lines surrounding it. The point at bottom left is $(0, 0)$.

Definition 2.2 (8-neighbourhood/Naive segments). *We say that 2 points $p, q \in \mathbb{Z}^2$ are 8-neighbours iff $\|p - q\|_\infty = 1$. A set $A \subset \mathbb{Z}^2$ is then 8-connected iff every pair of points of A is connected by a path for the 8-neighborhood. A naive segment is an 8-connected part of a naive line.*

Note that the definition of J.-P. Réveillès imposes a strict inequality on the right in the definition of a naive discrete line because the space is then naturally partitioned into naive lines. Naive lines are 8-connected and minimise thickness for this property, so they are widely used in discrete geometry. The terms « naive » and « 8-connected » will be omitted here for convenience. We can also be interested in any pieces of naive lines, not necessarily connected:

Definition 2.3 (Piece of line (Figure 2)). *Let $S \subset \mathbb{Z}^2$ finite. We say that S is a piece of naive*

line iff there exists $a, b, \mu \in \mathbb{R}$ and such that $S \subset \mathcal{D}(a, b, \mu)$.

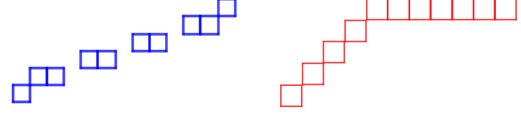


Figure 2 Points at bottom left are $(0, 0)$. **Left:** a piece of naive line. **Right:** a set that is not a piece of naive line.

In Figure 2, the set on the left is a piece of line because it admits $(4, 7, 11)$ as parameters. The set S on the right is not a right-hand piece because it is not H -convex, i.e. it does not satisfy $\text{Conv}(S) \cap \mathbb{Z}^d = S$. Given a finite piece of discrete line S , there are infinitely many parameters a, b, μ such that $S \subset \mathcal{D}(a, b, \mu)$. In fact, the set of parameters (a, b, μ) suitable for S is a non-empty interior \mathbb{R}^3 polyhedron because the property $S \subset \mathcal{D}(a, b, \mu)$ can be written as a set of linear constraints. For example, the piece at the left in Figure 2 (where the bottom left-hand end is $(0, 0)$) admits the parameters $(4, 11, 7)$, but also $(5, 14, 10)$ or $(\sqrt{13}, 10, \pi^2)$. In order to obtain canonical parameters, we define the notion of minimal parameters of a finite piece of discrete line.

Definition 2.4 (Minimal parameters). *Let S be a finite piece of line. We call the minimal parameters of S the integers a, b, μ that minimise $\|(a, b)\|_\infty$ such that $S \subset \mathcal{D}(a, b, \mu)$.*

Problem 2.5 (Piece of line recognition). *The problem of piece of line recognition is then to decide, given a set $S \subset \mathbb{Z}^2$, whether or not S is a piece of line and, if so, to calculate the minimal parameters of S .*

If S is connected (i.e. is a segment), the recognition algorithm of I. Debled-Rennesson and J.-P. Réveillès, presented in [9], is linear in $|S|$ and incremental, in the sense that adding a point updates the parameters of the segment in $O(1)$. In particular, the algorithm is based on the notion of leaning point:

Definition 2.6 (Leaning points (Figure 3)). *Let S be a piece of naive discrete line with minimal parameters (a, b, μ) and $p = (x, y)$ a point of S .*

We say that p is a lower (resp. upper) leaning point of S iff $ax - by + \mu = \|(a, b)\|_\infty - 1$ (resp. $ax - by + \mu = 0$).

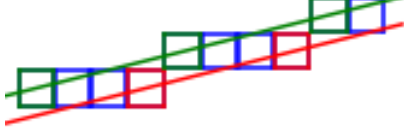


Figure 3 Representation of the lower (in red) and upper (in green) leaning points of a naive line with slope $1/4$.

3 Recognition using the Stern-Brocot tree

3.1 Stern-Brocot tree

Our segment recognition algorithm is based on a descent into the Stern-Brocot tree, first presented in [25] and [5]. This tree lists all positive irreducible fractions and presents them in the form of a binary search tree (see Figure 4).

We define the Stern-Brocot tree B_n truncated at level n by recurrence on n :

- B_0 consists of 2 nodes labelled $\frac{0}{1}$ and $\frac{1}{0}$ called inputs.
- given a truncated tree B_n , we list all the fractions it contains using the prefix depth path. Between 2 fractions $\frac{a}{b}$ and $\frac{c}{d}$, we insert into the tree the fraction $\frac{a}{b} \oplus \frac{c}{d} \stackrel{\text{def}}{=} \frac{a+c}{b+d}$.

Let us build the tree in Figure 4. Initially, we start with the nodes $\frac{0}{1}$ and $\frac{1}{0}$. From these 2 fractions, we construct their sum $\frac{0}{1} \oplus \frac{1}{0} = \frac{1}{1}$ which we place in the middle of the 2. The list of fractions considered is then, in order $[\frac{0}{1}, \frac{1}{1}, \frac{1}{0}]$. Between $\frac{0}{1}$ and $\frac{1}{1}$ (so as the left-hand child of $\frac{1}{1}$), we insert the fraction $\frac{0}{1} \oplus \frac{1}{1} = \frac{1}{2}$. Similarly, the fraction $\frac{1}{1} \oplus \frac{1}{0} = \frac{2}{1}$ is inserted as the right-hand child of $\frac{1}{1}$. The new list of fractions is $[\frac{0}{1}, \frac{1}{2}, \frac{1}{1}, \frac{2}{1}, \frac{1}{0}]$. To build the next stage, we insert the fraction $\frac{1}{3}$ between $\frac{0}{1}$ and $\frac{1}{2}$ (i.e. as a left-hand child of $\frac{1}{2}$), and so on.

Remark 3.1. This addition of fractions is actually the addition of the pairs (a, b) and (c, d) . The irreducible fraction $\frac{a}{b}$ and the pair (a, b) will later be blithely confused.

Theorem 3.2 (Stern-Brocot [25, 5]). The Stern-Brocot tree, i.e. the union of $(B_n)_{n \in \mathbb{N}}$, contains exactly once each irreducible fraction of \mathbb{Q}_+ .

3.2 General algorithm for recognising a piece of line

Given a digital subset S of \mathbb{Z}^2 , the principle of the algorithm is then to find the slope a/b of the piece S in the Stern-Brocot tree. By symmetry and translation, we can always assume that S lies in the first octant (i.e. verifies $0 \leq a \leq b$) and has $(0, 0)$ as its minimum abscissa point. We first propose a general version of the algorithm for arbitrary pieces, before optimising it for discrete connected line segments. We search for an admissible slope $\frac{a}{b}$ in the Stern-Brocot tree.

Given a slope $\frac{a}{b}$, we can easily test whether it is suitable for a piece S . We start by calculating the set of remainders $R \stackrel{\text{def}}{=} \{ax - by \mid (x, y) \in S\}$. The slope $\frac{a}{b}$ is admissible iff there exists $\mu \in \mathbb{Z}$ such that $R + \mu \subset \llbracket 0, b - 1 \rrbracket$, i.e. iff $\max(R) - \min(R) < b$. In this case, we also obtain $\mu = -\min(R)$ as a solution. We can therefore test in linear time in $O(|S|)$ whether some slope $\frac{a}{b}$ is suitable and, if so, return a suitable μ shift.

When the slope $\frac{a}{b}$ considered is not suitable for S , we go down the Stern-Brocot tree and it is therefore necessary to decide whether to go left or right, i.e. to decrease or increase the slope. To do this, we look at a minimum remainder point p_{\min} and a maximum remainder point p_{\max} . Note that p_{\min} and p_{\max} are not necessarily unique, but we show in the invariant proof of the algorithm that the choice of p_{\min} and p_{\max} does not change the correction and the complexity of the algorithm. The proof of the invariant shows that the relative position of a minimum point and a maximum point depends only on the sign of the difference between the current slope of the algorithm and a suitable slope of the piece. The algorithm then

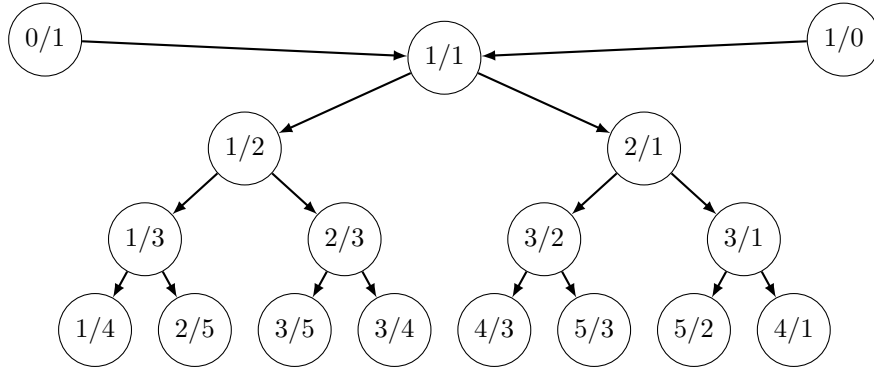


Figure 4 Illustration of Stern-Brocot tree, truncated at depth 4.

decreases the slope if p_{\min} is to the left of p_{\max} and increases it if p_{\min} is to the right of p_{\max} . Intuitively, decreasing the slope decreases the remainder of the points on the right and increases the remainder of the points on the left. So, when p_{\min} is to the left of p_{\max} , the difference in scalar product between these two points decreases. This method is illustrated on an example in Figure 5 and then formally proven.

Finally, it is necessary to have a stopping criterion in the case where the set S considered is not a piece of line. It is known that, if S is a piece of line, then the minimal parameters of S are smaller than $D \stackrel{\text{def}}{=} \max_{x,y \in S} \|y - x\|_{\infty}$, called the **diameter of S** . This can be proved, for example, using the segment recognition algorithm of [9], which characterises segments by their leaning points. We propose here an alternative proof, based on a study of remainders. The main interest of this proof is that it extends more easily into higher dimensions than the previous argument.

Theorem 3.3. *Let S be a piece of line with minimal parameters (a, b, μ) and diameter D . Then $D \geq \|(a, b)\|_{\infty}$.*

Proof. Without loss of generality, we can assume that S is in the first octant. Thus $0 \leq a \leq b$. Should we translate S , we can assume that $\mu = 0$ and that $(0, 0)$ is a lower leaning point of S . The set $\underline{S} \stackrel{\text{def}}{=} \{x \mid (x, y) \in S\}$ is called the support of S . Consider the parents $\frac{a_1}{b_1}$ and $\frac{a_2}{b_2}$ of $\frac{a}{b}$ in the Stern-Brocot tree. Let us call S_1 and S_2 the pieces of line with parameters $(a_1, b_1, 0)$ and $(a_2, b_2, 0)$ and

the same support as S . We will show the supports on which S and S_1 coincide, or on which S and S_2 coincide. Let us take $(x, y) \in \mathbb{Z}^2$. Let $r \stackrel{\text{def}}{=} \frac{a}{b}x - y$ and $r_1 \stackrel{\text{def}}{=} \frac{a_1}{b_1}x - y$. Thus,

$$\begin{aligned} |r - r_1| &= \left| \left(\frac{a}{b}x - y \right) - \left(\frac{a_1}{b_1}x - y \right) \right| \\ &= \left| \left(\frac{a}{b} - \frac{a_1}{b_1} \right) x \right| \\ &= \frac{|x|}{bb_1} \end{aligned}$$

If $(x, y) \in S \setminus S_1$ with $x > 0$, we have $r_1 < 0 \leq r$. Since r_1 is a rational with a denominator less than b_1 , we even have $r_1 \leq -\frac{1}{b_1}$. So $\frac{1}{b_1} \leq |r - r_1| = \frac{x}{bb_1}$. We can therefore deduce $x \geq b$.

If $(x, y) \in S \setminus S_1$ with $x < 0$, we have $r < 1 \leq r_1$. Since r is a rational with a denominator less than b , we even have $r \leq 1 - \frac{1}{b}$. So $\frac{1}{b} \leq |r - r_1| = \frac{-x}{bb_1}$. From this we deduce $x \leq -b_1$.

Hence S and S_1 coincide on the support $\llbracket -b_1 + 1, b - 1 \rrbracket$. Similarly, we show that S and S_2 coincide on the support $\llbracket -b + 1, b_2 - 1 \rrbracket$. For S to coincide with neither S_1 nor S_2 , it is therefore necessary for S to contain :

- either b (and 0) ;
- or $-b$ (and 0) ;
- or $-b_1$ and b_2 ;

In all cases, we have found 2 points on the support of S at a distance b . The diameter D of S therefore satisfies $D \geq b$. \square

For example, Let us look at the execution of the algorithm (see Figure 5) on the segment with parameters $(5, 8, 3)$ and length 11. The algorithm starts with the slope $\frac{1}{2}$ in the Stern-Brocot tree. The minimum remainder of -3 is obtained in $p_{\min} = (9, 6)$ and the maximum remainder of 0 is obtained in $p_{\max} = (0, 0)$. The difference in remainders is 3 which is greater than 2 so the slope is not suitable. As p_{\min} is to the right of p_{\max} , we increase the slope by going down to the right in the tree to arrive at a slope $\frac{2}{3}$. For the $\frac{2}{3}$ slope, the slope is still not suitable (the difference in remainders is $3 \geq 3$) and p_{\min} is to the left of p_{\max} so the slope is too steep: we move down the left-hand side of the tree towards $\frac{3}{5}$. For this new slope, the difference in remainders is $5 \geq 5$. As p_{\min} is to the right of p_{\max} , we increase the slope by going down to the right towards $\frac{5}{8}$. Finally, for $\frac{5}{8}$, the difference in remainders is 7, which is strictly smaller than 8, so the slope is appropriate. The shift is then the opposite of the minimum remainder -3 , so the segment parameters are $(5, 8, 3)$.

Theorem 3.4. *The recognition algorithm above decides whether a finite set S of the first octant is a piece of line, returns its minimal parameters if it is, and then terminates in $O(nh)$ where $n = |S|$ and h is the depth of the parameters of S in the Stern-Brocot tree.*

Proof. **The algorithm terminates** because $D - b$ is a strictly decreasing variable with positive values.

Correction of returned parameters: Let us first show that if the algorithm returns (a, b, μ) then a, b, μ are parameters of the segment S . Suppose the algorithm returns (a, b, μ) . If $b = 1$ then the algorithm has concluded on a horizontal or diagonal segment, hence the result. Otherwise, let us call p_{\min}, p_{\max} the output of `Extremal-remainders`(S, a, b). We then have $\langle p_{\max} - p_{\min}, (a, -b) \rangle < b$ and $\mu = -\langle p_{\min}, (a, -b) \rangle$. Let p be a point of S . By an obvious correction of the `Extremal-Remainders` algorithm,

Algorithm 1: General algorithm for recognising a piece of line

Input: $S \subset \mathbb{Z}^2$ in the first octant with $(0, 0)$ as its left extremity

Output: Decide whether S is a piece of line and, if so, return its minimal parameters (a, b, μ) .

`PIECE-RECOGNITION`(S):

```

if  $S$  is horizontal (resp. diagonal) then
  | Return  $(0, 1, 0)$  (resp.  $(1, 1, 0)$ ) ;
 $D \leftarrow$  diameter of  $S$  (difference between
  maximum and minimum abscissas) ;
 $\text{slope}_{\inf} \leftarrow \frac{0}{1}$  (lower limit) ;
 $\text{slope}_{\sup} \leftarrow \frac{1}{1}$  (upper limit) ;
 $\frac{a}{b} \leftarrow \text{slope}_{\inf} \oplus \text{slope}_{\sup}$  (current slope
  initially worth  $1/2$ ) ;
while  $b \leq D$  do
   $p_{\min}, p_{\max} \leftarrow$ 
    EXTREMAL-REMAINDERS( $S, a, b$ )
    (minimum and maximum remainder
    points for the  $a/b$  slope) ;
  if  $\langle p_{\max} - p_{\min}, (a, -b) \rangle < b$  (the  $a/b$ 
    slope is suitable) then
    |  $\mu \leftarrow -\langle p_{\min}, (a, -b) \rangle$  ;
    | Return  $(a, b, \mu)$ 
  if  $p_{\min}$  is to the left of  $p_{\max}$  (the
    current slope is too steep) then
    |  $\text{slope}_{\sup} \leftarrow \frac{a}{b}$  ;
  if  $p_{\min}$  is to the right of  $p_{\max}$  (the
    current slope is too small) then
    |  $\text{slope}_{\inf} \leftarrow \frac{a}{b}$  ;
   $\frac{a}{b} \leftarrow \text{slope}_{\inf} \oplus \text{slope}_{\sup}$  (new current
    slope) ;
Return "Not a piece of line" ;

```

Algorithm 2: Extremal remainders

Input: $S \subset \mathbb{Z}^2$, a, b parameters of the slope tested

Output: p_{\min}, p_{\max} points of S having the minimum and maximum remainders with slope $\frac{a}{b}$

`EXTREMAL-REMAINDERS`(S, a, b):

```

 $p_{\min}, p_{\max} \leftarrow S[0], S[0]$  ;
for  $i$  going from 1 to  $n - 1$  do
   $p \leftarrow S[i]$  ;
  if  $\langle p, (a, -b) \rangle < \langle p_{\min}, (a, -b) \rangle$  then
    |  $p_{\min} \leftarrow p$  ;
  if  $\langle p, (a, -b) \rangle > \langle p_{\max}, (a, -b) \rangle$  then
    |  $p_{\max} \leftarrow p$  ;
Return  $p_{\min}, p_{\max}$  ;

```

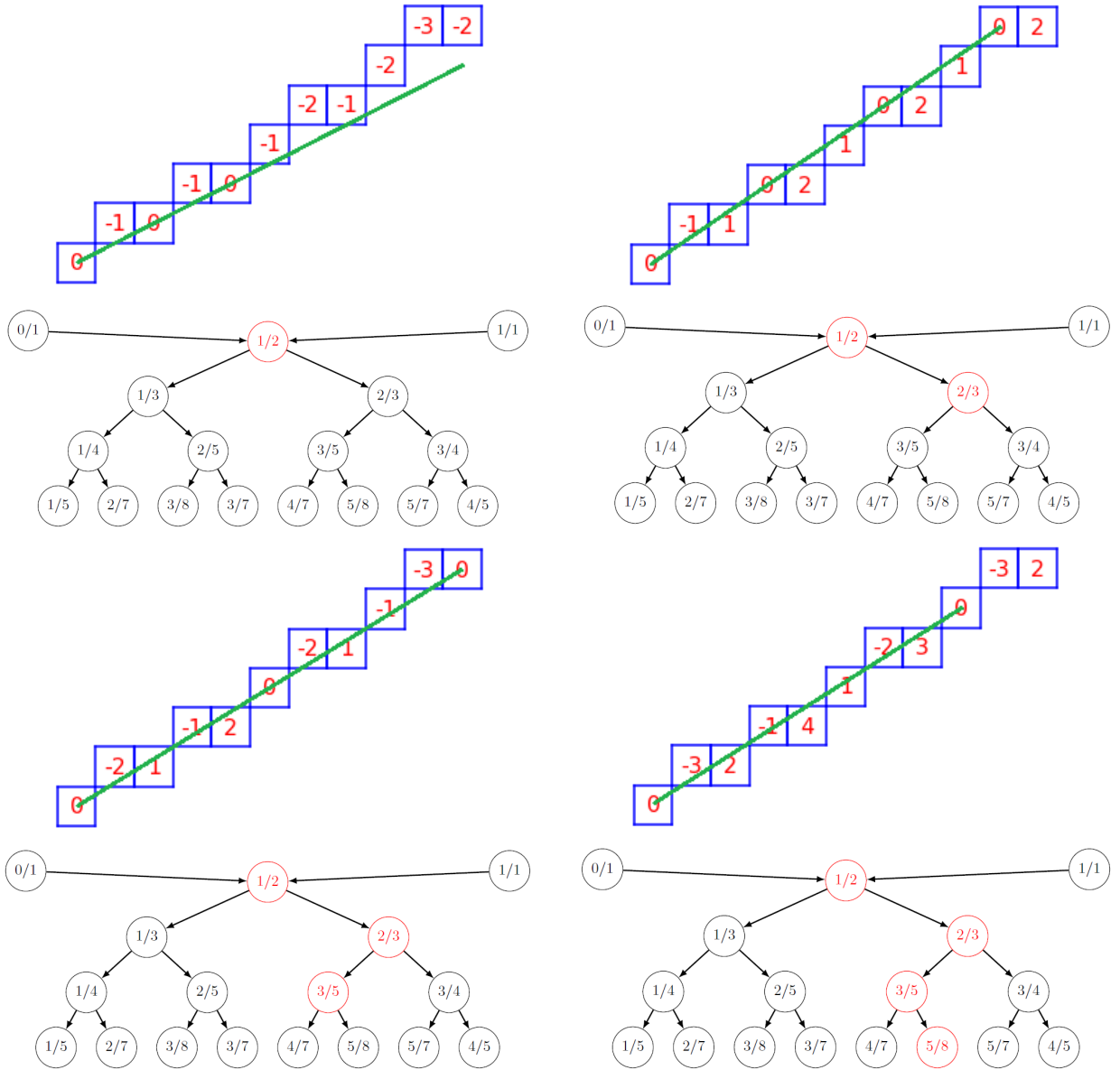


Figure 5 The algorithm runs on the segment with parameters $(5, 8, 3)$ and length 11. The green lines in the segment correspond to the successive slopes considered by the algorithm.

$\langle p_{\min}, (a, -b) \rangle \leq \langle p, (a, -b) \rangle \leq \langle p_{\max}, (a, -b) \rangle$.
Therefore $0 \langle p, (a, -b) \rangle - \langle p_{\max}, (a, -b) \rangle \leq \langle p_{\max}, (a, -b) \rangle - \langle p_{\min}, (a, -b) \rangle < b$. Then $0 \leq \langle p, (a, -b) \rangle + \mu < \|(a, b)\|_{\infty}$, which proves that S is a piece of line with parameters (a, b, μ) . In particular, if S is not a piece of line, the algorithm returns ‘Not a segment’.

It remains to show that if S is a piece of line then the algorithm returns its minimal

(a_S, b_S, μ_S) parameters. If S is horizontal or diagonal, the result is trivial.

Otherwise, we show the following **invariant**: $\text{slope}_{\inf} < \frac{a_S}{b_S} < \text{slope}_{\sup}$. The invariant is true initially because S is in the first octant and is neither horizontal nor diagonal. Assuming the invariant is true, let us perform an additional loop for the current slope $\frac{a}{b}$. Let p_{\min}, p_{\max} be the output of $\text{Extremal-remainders}(S, a, b)$ and let $\delta =$

$(x, y) \stackrel{\text{def}}{=} p_{\max} - p_{\min}$. Note first of all that, as S is in the first octant, the abscissa x of δ is non-zero. Since $p_{\min} = (x_{\min}, y_{\min})$ and $p_{\max} = (x_{\max}, y_{\max})$ are points of S , $0 \leq a_S x_{\min} - b_S y_{\min} + \mu_S < b_S$ and $0 \leq a_S x_{\max} - b_S y_{\max} + \mu_S < b_S$. By subtracting the 2 identities, we deduce $-b_S < a_S x - b_S y < b_S$. So $\frac{a_S}{b_S} x - y < 1$. Furthermore, $\langle \delta, (a, -b) \rangle \geq b$ so $ax - by \geq b$ hence $\frac{a}{b} x - y \geq 1 > \frac{a_S}{b_S} x - y$. So $\frac{a}{b} x > \frac{a_S}{b_S} x$. If $x > 0$ (p_{\min} is to the left of p_{\max}) then $\frac{a}{b} > \frac{a_S}{b_S}$ so $\frac{a_{\inf}}{b_{\inf}} < \frac{a_S}{b_S} < \frac{a}{b}$ which proves the invariant. If $x < 0$ (p_{\min} is to the right of p_{\max}) then $\frac{a}{b} < \frac{a_S}{b_S}$ so $\frac{a}{b} < \frac{a_S}{b_S} < \frac{a_{\sup}}{b_{\sup}}$ which proves the invariant. **This invariant proof also shows that the choice of p_{\min} and p_{\max} among the minimal or maximal points does not change the relative position of p_{\min} with respect to p_{\max} .**

End of proof of correction: Once the invariant has been proved, note that $b_S \leq D$ by the theorem 3.3. The fraction $\frac{a_S}{b_S}$ will therefore, by the invariant, necessarily be visited and the algorithm will then conclude.

Minimality: finally, let us show by convexity that the parameters (a, b, μ) returned are indeed the minimal parameters (a_S, b_S, μ_S) of the segment. Let $\frac{a'}{b'}$ be the common ancestor of $\frac{a}{b}$ and $\frac{a_S}{b_S}$ in the Stern-Brocot tree. $\frac{a'}{b'}$ is therefore a convex combination of $\frac{a}{b}$ and $\frac{a_S}{b_S}$, i.e. $\frac{a'}{b'} = \lambda_1 \frac{a}{b} + \lambda_2 \frac{a_S}{b_S}$. Let $(x, y) \in S$. $0 \leq \frac{a'}{b'} x + \frac{\mu'}{b'} - y < 1$ and $0 \leq \frac{a_S}{b_S} x + \frac{\mu_S}{b_S} - y < 1$. Taking the convex combination of the 2 equations, we obtain $0 \leq \left(\lambda_1 \frac{a_S}{b_S} + \lambda_2 \frac{a_S}{b_S} \right) x + \left(\lambda_1 \frac{a_S}{b_S} + \lambda_2 \frac{a_S}{b_S} \right) - y < 1$. Thus, by posing $\mu' \stackrel{\text{def}}{=} \lambda_1 \frac{\mu}{b} + \lambda_2 \frac{\mu_S}{b_S}$, we obtain $0 \leq \frac{a'}{b'} x + \frac{\mu'}{b'} - y < 1$. So (a', b', μ') are parameters for S . These parameters have been visited by the algorithm, so $(a, b, \mu) = (a', b', \mu')$. Furthermore, as (a', b', μ') is an ancestor of (a_S, b_S, μ_S) , $b' \leq b_S$ so, by minimality, $(a_S, b_S, \mu_S) = (a', b', \mu')$. So the parameters returned by the algorithm are indeed minimal.

Complexity: the algorithm moves down the Stern-Brocot tree at each loop iteration. The number of loop iterations is therefore $O(h)$. In a loop, the only costly part is the call to the Remainder-extreme function, which is an array traversal, so the cost is $O(n)$. The total complexity is therefore $O(nh)$. \square

Algorithm 3: Extremal remainders

Input: a, b parameters of the slope being tested, P set of points to be tested

Output: p_{\min}, p_{\max} points of P having the minimum and maximum remainders with slope $\frac{a}{b}$.

```

EXTREMAL-REMAINDERS2( $P, a, b$ ):
   $p_{\min}, p_{\max} \leftarrow$  first element of  $P$  ;
  for  $p$  in  $P$  do
    if  $\langle p, (a, -b) \rangle < \langle p_{\min}, (a, -b) \rangle$  then
      |  $p_{\min} \leftarrow p$  ;
    if  $\langle p, (a, -b) \rangle > \langle p_{\max}, (a, -b) \rangle$  then
      |  $p_{\max} \leftarrow p$  ;
  Return  $p_{\min}, p_{\max}$  ;

```

The complexity announced in the correction theorem depends on the depth of a parameter in the output $\frac{a}{b}$. It is therefore interesting to evaluate the depth of a fraction in the Stern-Brocot tree. Unfortunately, the fraction $\frac{1}{D}$ is at depth D in the tree. Our algorithm therefore has a worst-case complexity of $O(nD)$ where n is the size of S and D its diameter. However, if we consider a uniform distribution over $\{\frac{p}{q} \mid 0 \leq p \leq q \leq D, p \wedge q = 1\}$, the average depth of the fraction $\frac{p}{q}$ is in $O(\log^2(D))$ as demonstrated in [27] by an analysis of the average complexity of Euclid's algorithm. We then obtain an average complexity in $O(n \log^2(D))$ for piece of line recognition algorithm. This new algorithm does not reach the complexity of the COBA algorithm presented in [8], which is in $O(n \log(D))$ but, on the other hand, the COBA algorithm does not return the minimal parameters of S .

3.3 An incremental algorithm for segments

In the case of discrete segments, we can significantly improve the complexity of our algorithm by using an incremental version and the notion of leaning point. First of all, we can add the points of the segment one by one and calculate the parameters of the new segment by starting again from the old parameters in the Stern-Brocot tree: the method then becomes incremental. Moreover, a large part of the complexity of our algorithm comes from the calculation of the remainders of

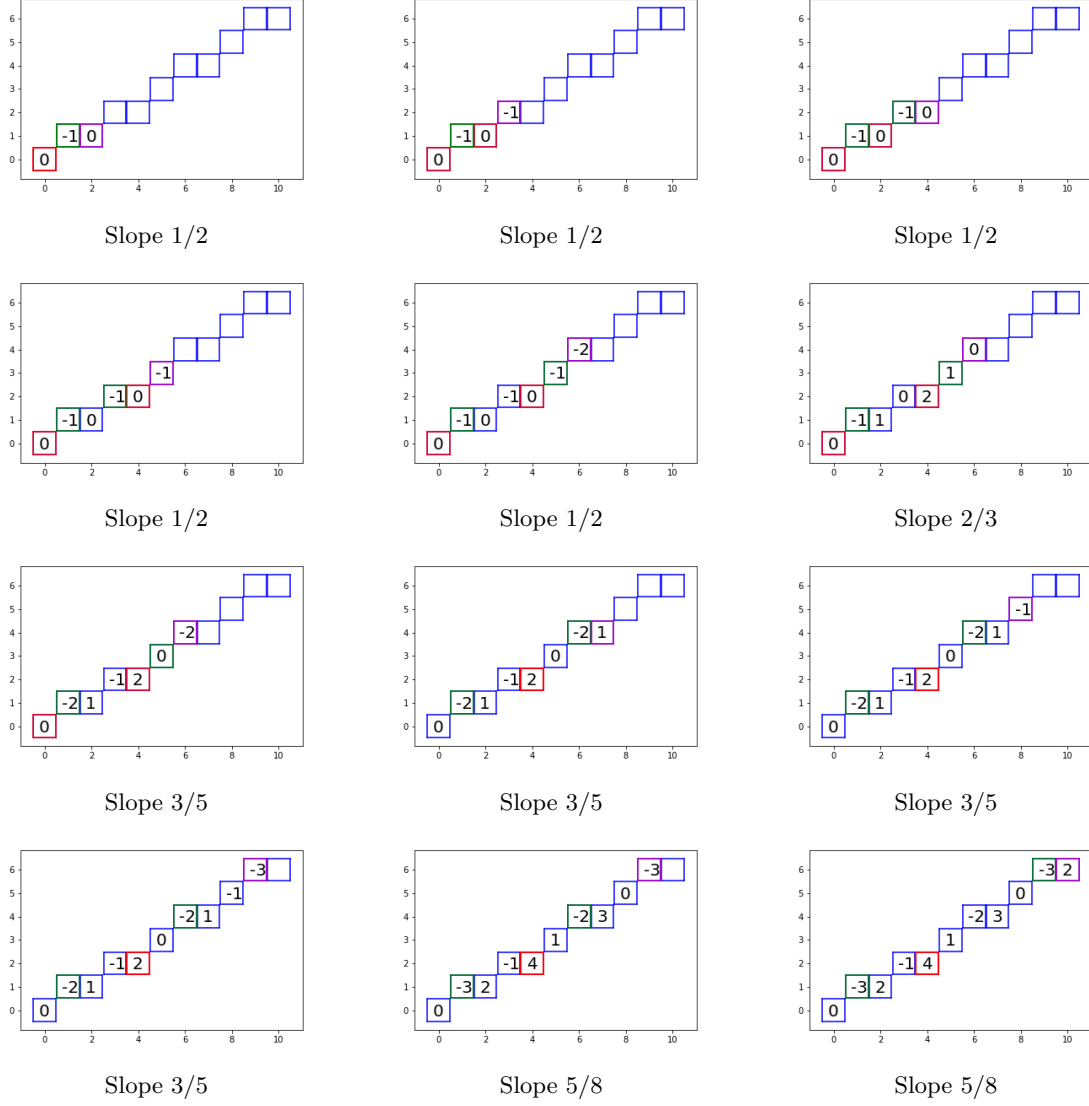


Figure 6 Execution of the incremental algorithm on the segment with parameters $(5, 8, 3)$ and length 11.

all the points in S . In a segment, however, there is no interest in calculating the remainders for all the points. The remainders associated with the extremal lower and upper leaning points (i.e. with minimum or maximum abscissa) are in fact sufficient to guarantee that an equation is satisfied for all the points of the segment (by correction of the algorithm in [9]). Therefore, at most a constant number of remainders must be calculated in the Extremal-remainder function at each step: the call is thus made in $O(1)$ and the total complexity therefore falls to $O(n + h) = O(n)$.

Let us look at the flow of the algorithm on an example in Figure 6. The numbers correspond to the scalar products with the slope, but only the remainders of the extremal leaning points (in red and green) are considered by the algorithm. The magenta point corresponds to the current point, which is added incrementally to the segment.

The incremental algorithm thus provides an efficient version of segment recognition using the Stern-Brocot tree. It has the same complexity as the segment recognition algorithm presented in

Algorithm 4: Incremental algorithm for segment recognition

Input: $S \subset \mathbb{Z}^2$, set of points in the first octant with abscissas indexed from 0 to $n - 1$.

Output: Decide whether S is a segment and, if so, return its minimal parameters (a, b, μ) .

```

INCREMENTAL-RECOGNITION( $S$ ):
   $i \leftarrow 0$  (maximum index of points considered) ;
  while the part of the segment under consideration is horizontal or diagonal and  $i < n$  do
     $i \leftarrow i + 1$  ;
  if  $i = n$  then
    Return  $(0, 1, 0)$  or  $(1, 1, 0)$  depending on whether the piece in question is horizontal or diagonal. ;
  else
     $test \leftarrow \{(0, 0), (i - 1, 0)\}$  or  $\{(0, 0), (i - 1, i - 1)\}$  either horizontal or diagonal ;
     $slope_{inf} \leftarrow \frac{0}{1}$  (lower limit) ;
     $slope_{sup} \leftarrow \frac{1}{1}$  (upper limit) ;
     $\frac{a}{b} \leftarrow slope_{inf} \oplus slope_{sup}$  (current slope initially worth  $1/2$ ) ;
     $\mu \leftarrow 0$  ;
    while  $i < n$  (the entire segment has not yet been considered) do
       $correct \leftarrow \text{FALSE}$  (indicates whether the current slope is suitable) ;
      while  $b < n$  and not  $correct$  do
         $p_{min}, p_{max} \leftarrow \text{EXTREMAL-REMAINDERS2}(a, b, test \cup \{S[i]\})$  (minimum and maximum remainder points for the  $a/b$  slope) ;
        if  $\langle p_{max} - p_{min}, (a, -b) \rangle < b$  (the slope  $a/b$  is suitable) then
           $\mu \leftarrow -\langle p_{min}, (a, -b) \rangle$  ;
           $i \leftarrow i + 1$  (if the slope is suitable, move on to the next point);
           $test \leftarrow$  all the end leaning points of the segment in question using  $a, b, \mu$  ;
           $correct \leftarrow \text{TRUE}$  ;
        if not  $correct$  then
          if  $p_{min}$  is on the left of  $p_{max}$  (the current slope is too steep) then
             $slope_{sup} \leftarrow \frac{a}{b}$  ;
          if  $p_{min}$  is on the right of  $p_{max}$  (the current slope is too small) then
             $slope_{inf} \leftarrow \frac{a}{b}$  ;
           $\frac{a}{b} \leftarrow slope_{inf} \oplus slope_{sup}$  (new current slope) ;
        if not  $correct$  and  $b \geq n$  then
          Return 'Not a segment'
    Return  $a, b, \mu$ 

```

[9], and is in fact only an interpretation of this algorithm in the Stern-Brocot tree.

4 Extension to higher dimensions

4.1 Hyperplane recognition

The notion of a discrete line naturally extends into higher dimensions, as presented in [1] :

Definition 4.1 (Naive arithmetic hyperplane). Let $v \in \mathbb{R}^d$ be non-zero and $\mu \in \mathbb{R}$. The naive arithmetic (or discrete) hyperplane with normal vector v and shift μ is the set

$$\mathbb{P}(v, \mu) \stackrel{\text{def}}{=} \{x \in \mathbb{Z}^d \mid 0 \leq \langle x, v \rangle + \mu < \|v\|_\infty\}$$

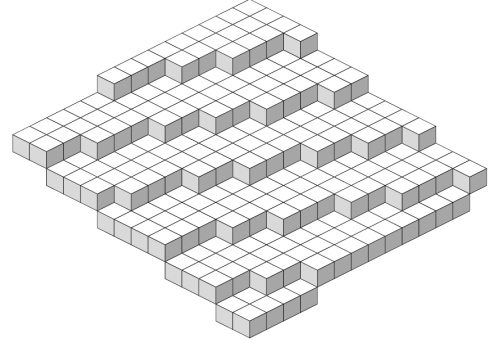


Figure 7 A naive arithmetic plane with normal vector $(7, 17, 57)$ and shift 0.

We can also extend the notions of line pieces and minimal parameters to any finite dimension.

Definition 4.2 (Piece of hyperplane). Let $S \subset \mathbb{Z}^d$. We say that S is a piece of hyperplane iff there exists $v \in \mathbb{R}^d$ and $\mu \in \mathbb{R}$ such that $S \subset \mathbb{P}(v, \mu)$.

Definition 4.3 (Minimal parameters). Let S be a piece of finite hyperplane. We call the minimal parameters of S the pair (v, μ) where $v \in \mathbb{Z}^d$ and $\mu \in \mathbb{Z}$, such that $S \subset \mathbb{P}(v, \mu)$ and $\|v\|_\infty$ is minimized.

The hyperplane piece recognition problem is then to decide, given a finite set $S \subset \mathbb{Z}^d$, whether

or not S is a hyperplane piece and, if so, to compute its minimal parameters. We refer the reader to the relevant paragraph in the introduction for a brief presentation of some methods for solving this problem. In this section, we present a method for recognising arithmetic hyperplanes based on a general Stern-Brocot tree. The interest of this method is not so much its efficiency compared to other algorithms in the state of the art, but rather the combinatorial link between the Stern-Brocot tree and the arithmetic hyperplanes.

4.2 Stern-Brocot tree extension of Lennerstad

The Stern-Brocot tree naturally extends into 3D and higher dimensions, as shown by Lennerstad [21]. The tree presented here lists the d -tuples of natural integers that are prime to each other in their set. Instead of using 2 end points as before, we use d end points. These points are represented as a $(d-1)$ -simplex form as follows (illustrated in Figure 8):

- Initially, the d extremities are the points e_i of the canonical base of \mathbb{Z}^d .
- Given d points with extremities p_1, \dots, p_d (affinely independent in \mathbb{R}^{d-1} by induction on the tree), we construct the $(d-1)$ -simplex Γ with extremities p_1, \dots, p_d .
- Given a permutation σ of \mathfrak{S}_d , consider the sub-simplex $\Gamma[\sigma]$ formed by the points q_1, \dots, q_d where $q_j \stackrel{\text{def}}{=} \sum_{i=1}^j p_{\sigma(i)}$.
- The children of the simplex Γ in the tree are then the sub-simplexes $\Gamma[\sigma]$ for $\sigma \in \mathfrak{S}_d$.

The first level of the Stern-Brocot tree is shown in Figure 8 and subsequently in normalised form. Since two collinear normal vectors represent the same plane, we can project the vectors v by requiring $\|v\|_1 = 1$. Summing a group of vectors is then geometrically equivalent to considering their barycentre.

By symmetry, we can assume that the normal vector v of the piece of hyperplane under

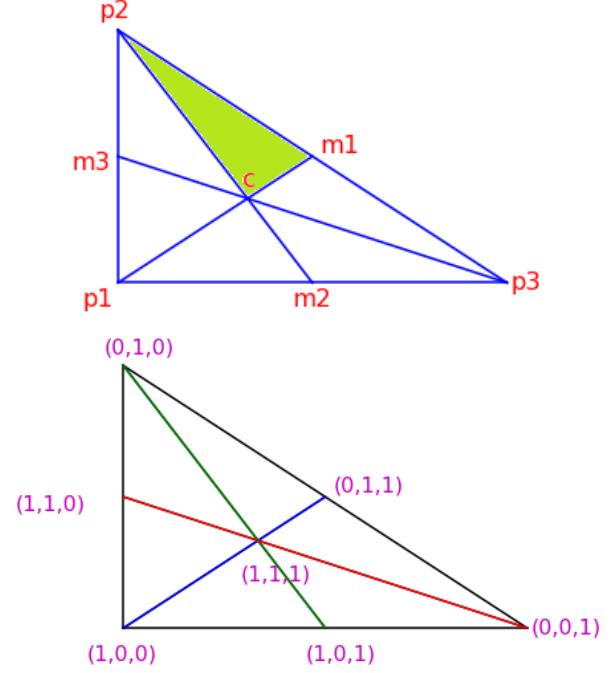


Figure 8 Top row: a step in the construction of the tree in dimension 3. The points p_1, p_2, p_3 are the extremities of the simplex. The points m_i represent the points $p_{i+1} + p_{i+2}$ (where the indices are taken modulo 3). Point c represents $p_1 + p_2 + p_3$. The permutation σ such that $\sigma(i) = (i+1) \bmod 3$ then gives the simplex with extremities $p_{\sigma(1)} = p_2$, $p_{\sigma(1)} + p_{\sigma(2)} = p_2 + p_3 = m_1$ and $p_{\sigma(1)} + p_{\sigma(2)} + p_{\sigma(3)} = c$, i.e. the coloured triangle p_2, m_1, c at the top right of the figure. Bottom row: an example with initialisation at $p_i = e_i$.

study has positive coordinates. The d initial extremities of the Stern-Brocot tree are then the e_i vectors of the canonical basis. In order to know whether a current vector v is suitable for the piece S , we calculate as before the set of remainders $R \stackrel{\text{def}}{=} \{\langle x, v \rangle \mid x \in S\}$ and check whether $\max(R) - \min(R) < \|v\|_\infty$. If this is the case, v is suitable for a shift $\mu \stackrel{\text{def}}{=} -\min(R)$. Otherwise, we need to go down one level in the sub-simplexes of the Stern-Brocot tree. The question is then, given a node (a d -simplex) of the tree, to determine into which of the $d!$ subtrees to descend. For the rest of this paper, we will only consider the case of dimension 3, as the results obtained can be generalised without effort to any finite dimension.

4.3 Separating chords

In order to choose the good sub-simplex, we define the notion of separating chord, using the definition of chord (see Figure 9 for an illustration):

Definition 4.4 (Chord). *Let $v \in \mathbb{Z}^d$ and $\mu \in \mathbb{Z}$. We say that the chord $\delta \in \mathbb{Z}^d$ appears in the plane $\mathbb{P}(v, \mu)$ if and only if there exist $x, y \in \mathbb{P}(v, \mu)$ such that $\delta = y - x$.*

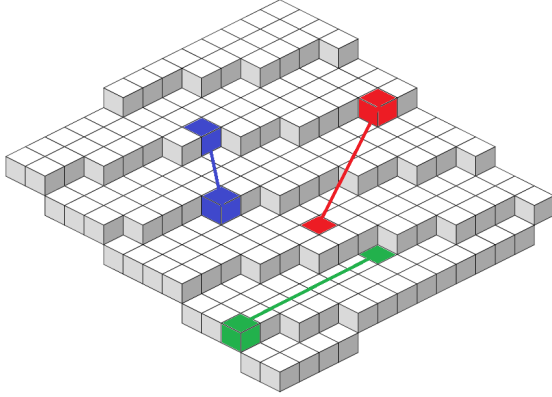


Figure 9 Chord in the plane with normal vector $(7, 17, 57)$. The plane contains, among others, the chords $(7, 0, 0)$ (green), $(2, 3, -1)$ (blue) and $(6, 4, -2)$ (red). However, the plane does not contain the chords $(0, 0, 1)$ or $(0, 4, 0)$.

Note that whether or not a chord belongs to a hyperplane does not depend on the considered shift μ , since the set of finite patterns in a hyperplane is shift invariant. This property can also be proved in a more pedestrian way, via Bézout's theorem or by density. Note also that a chord δ appears in a plane iff $-\delta$ appears in it. We can therefore confuse the chords δ and $-\delta$, which algebraically amounts to quotienting by the antipodal.

Remark 4.5. *A chord δ appears in a plane of normal vector v iff there exist $x, y \in \mathbb{P}(v, \mu)$ such that $\delta = y - x$ and $\begin{cases} 0 \leq \langle x, v \rangle + \mu < \|v\|_\infty \\ 0 \leq \langle y, v \rangle + \mu < \|v\|_\infty \end{cases}$. Thus, by subtracting, δ appears in a plane with normal vector v iff $|\langle \delta, v \rangle| < \|v\|_\infty$.*

The idea is to determine in which sub-simplex to go down in the Stern-Brocot tree by using some very special chords called separating chords.

These chords define boundaries along the medians of the triangle as shown in Figure 10 (left).

Definition 4.6 (Separating chords, Figure 10 (top)). *Let $p_1, p_2, p_3 \in \mathbb{N}^3$ be Stern-Brocot endpoints. We say that a pair $(\delta_-^{(i)}, \delta_+^{(i)})$ separates (p_1, p_2, p_3) along p_i iff for all $v \in \mathcal{C}(p_1, p_2, p_3)$ (cone generated by p_1, p_2, p_3):*

- *the chord $\delta_-^{(i)}$ appears in $\mathbb{P}(v, \mu)$ iff $v \in \mathcal{C}(p_i, m_i, p_{i-1}) \setminus \mathcal{C}(p_i, m_i)$ (triangle p_i, m_i, p_{i-1} deprived of the segment p_i, m_i).*
- *the chord $\delta_+^{(i)}$ appears in $\mathbb{P}(v, \mu)$ iff $v \in \mathcal{C}(p_i, m_i, p_{i+1}) \setminus \mathcal{C}(p_i, m_i)$ (triangle p_i, m_i, p_{i+1} deprived of the segment p_i, m_i).*

The $\delta_-^{(i)}$ and $\delta_+^{(i)}$ chords are then said to be separating.

If we have separating chords for each median of the triangle, it is then possible to locate the position of the normal vector v with respect to each of the medians, and thus to know the Stern-Brocot sub-simplex into which to descend (see Figure 10 on the bottom).

The separating chords can be found in a simple way, by expressing the set of constraints they must satisfy and solving this system of constraints. By symmetry, we take the case where the vector v has increasing positive coordinates. Thus, the chord δ appears in the plane of normal vector v iff $0 \leq |\langle v, \delta \rangle| < |\langle v, e_3 \rangle|$. Given extremities p_1, p_2, p_3 , in order to find a separation for p_i , it is sufficient to impose the following constraints on $(\delta_-^{(i)}, \delta_+^{(i)})$:

$$\begin{cases} \langle p_i, \delta_-^{(i)} \rangle = \langle p_i, e_3 \rangle \\ \langle m_i, \delta_-^{(i)} \rangle = \langle m_i, e_3 \rangle \\ \langle p_{i-1}, \delta_-^{(i)} \rangle = \langle p_{i-1}, e_3 \rangle - 1 \end{cases}$$

$$\begin{cases} \langle p_i, \delta_+^{(i)} \rangle = \langle p_i, e_3 \rangle \\ \langle m_i, \delta_+^{(i)} \rangle = \langle m_i, e_3 \rangle \\ \langle p_{i+1}, \delta_+^{(i)} \rangle = \langle p_{i+1}, e_3 \rangle - 1 \end{cases}$$

The first two conditions for each system ensure that the limit of appearance of the chords

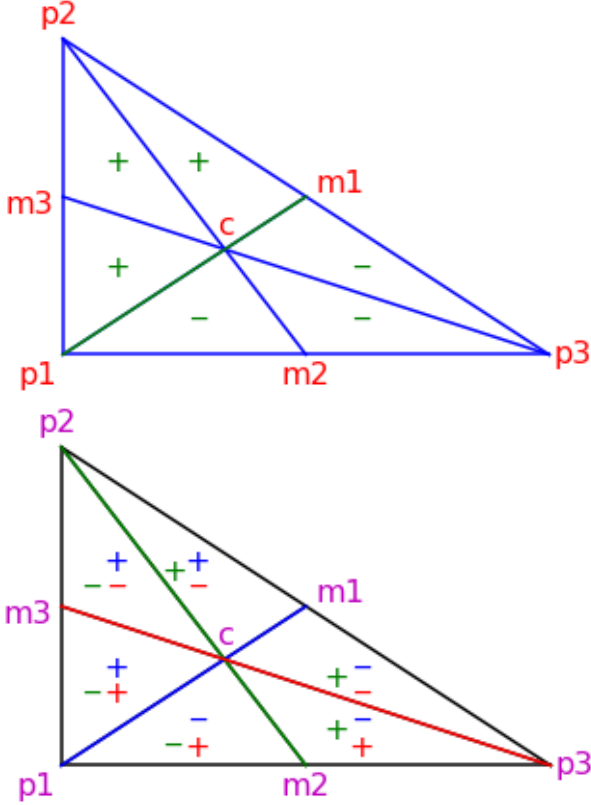


Figure 10 Top : illustration of a separation along p_1 . In the sub-triangles marked $-$, the plane contains the chord $\delta_-^{(1)}$ but not the chord $\delta_+^{(1)}$. In the triangles marked $+$, the plane contains the chord $\delta_+^{(1)}$ but not the chord $\delta_-^{(1)}$. On the green dividing line, neither of the two chords appears. Bottom : partition of the triangle according to the chords appearing in the planes. In green, blue and red, the boundary lines. The signs then correspond to the chords appearing in the planes of each sub-zone.

$\delta_-^{(i)}$ and $\delta_+^{(i)}$ is the green line passing through p_i and m_i (see Figure 10 (top)). The last condition ensures that $\delta_-^{(i)}$ appears in the plane of normal vector p_{i-1} (resp. $\delta_+^{(i)}$ in the plane of normal vector p_{i+1}). The characterisation by equivalence of the separation $(\delta_-^{(i)}, \delta_+^{(i)})$ then follows immediately from these observations by convexity. Finally, let us add that the 2 systems each admit a unique solution because they have determinant ± 1 (by induction on the tree). By separating the zones according to each p_i , we obtain a partition of the triangle p_1, p_2, p_3 into 6 sub-triangles (see Figure 10 on the bottom).

As for the last constraint $\langle p_{i-1}, \delta_-^{(i)} \rangle = \langle p_{i-1}, e_3 \rangle - 1$ which ensures that $\delta_-^{(i)}$ belongs to the plane with normal vector p_{i-1} , it can be made more flexible by requiring only $\langle p_{i-1}, \delta_-^{(i)} \rangle = \langle p_{i-1}, e_3 \rangle - \xi$ for $\xi \in [1, 2 \langle p_{i-1}, e_3 \rangle - 1]$. We later show that choosing $\xi = 1$ gives the smallest possible separating chords, which are therefore in general more relevant.

The separating chords can be used to characterise the membership of the normal vector of the plane to certain cones. For example, we can look at those for the Stern-Brocot tree in dimension 2, with extremities 0 and $1/2$ and centre $1/3$.

$$\begin{cases} \langle (1, 3), \delta_- \rangle = \|(1, 3)\|_\infty \\ \langle (0, 1), \delta_- \rangle = \|(0, 1)\|_\infty - 1 \end{cases}$$

$$\begin{cases} \langle (1, 3), \delta_+ \rangle = \|(1, 3)\|_\infty \\ \langle (1, 2), \delta_+ \rangle = \|(1, 2)\|_\infty - 1 \end{cases}$$

After solving the system, we obtain $\delta_- = (3, 0)$ and $\delta_+ = (3, 2)$. The lines whose slope is between 0 and $1/2$ are therefore :

- in the interval $[0, 1/3[$ iff they contain the chord $(3, 0)$, i.e. a step of size at least 4 (see Figure 11 on the top) ;
- in the interval $]1/3, 1/2]$ iff they contain the chord $(3, 2)$, i.e. a step of size exactly 2 (see Figure 11 on the bottom) ;
- the $1/3$ slope is the only one not to contain any of the 2 chords: the bearings are all exactly 3 in size (see Figure 11 in the middle).

The Stern-Brocot tree can then be repeated in dimension 2 with the various separations obtained for the interval $[0, 1]$, as in Figure 12. In general, as we show in their combinatorial study, the separating chords with respect to the slope $\frac{a}{b}$ are $\delta_- = (b, a - 1)$ and $\delta_+ = (b, a + 1)$.

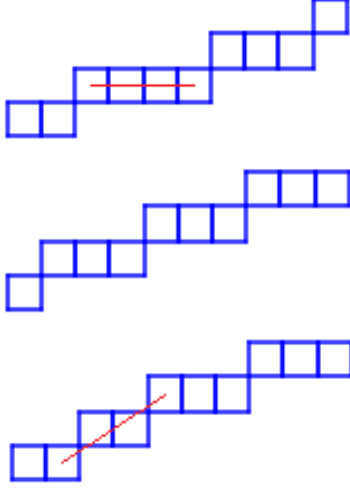


Figure 11 Three segments, with slopes strictly less than, equal to and greater than $\frac{1}{3}$ respectively, and the corresponding separating chords. The figure in the middle has no separating chord.

4.4 Combinatorial study of separating chords

The separating chords allow us to characterise geometrically in an arithmetic hyperplane the position of its normal vector in the Stern-Brocot tree. We propose here a geometric interpretation of these chords, by studying the system used to calculate them.

Using the previous notation (see Figure 10 (top)), consider a Stern-Brocot triangle with extremities p_1, p_2, p_3 and middles $m_i \stackrel{\text{def}}{=} p_{i-1} + p_{i+1}$ where $i \in \mathbb{Z}/3\mathbb{Z}$. Let us call δ_-, δ_+ the chords separating the triangle along the median $[p_i, m_i]$: the subscripts i on the chords are omitted here for clarity.

Let us take the system (S_-) characterising δ_- .

$$(S_-) : \begin{cases} \langle p_i, \delta_- \rangle &= \langle p_i, e_3 \rangle \\ \langle m_i, \delta_- \rangle &= \langle m_i, e_3 \rangle \\ \langle p_{i-1}, \delta_- \rangle &= \langle p_{i-1}, e_3 \rangle - \xi \end{cases}$$

which is rewritten, each time subtracting the scalar product on the right,

$$(S'_-) : \begin{cases} \langle p_i, \delta_- - e_3 \rangle &= 0 \\ \langle m_i, \delta_- - e_3 \rangle &= 0 \\ \langle p_{i-1}, \delta_- - e_3 \rangle &= -\xi \end{cases}$$

We can see that $\delta \stackrel{\text{def}}{=} \delta_- - e_3$ is orthogonal to p_i and m_i . So δ is proportional to the vector product $p_i \wedge m_i$. The final constraint $\langle p_{i-1}, \delta \rangle = -\xi$ is a normalisation constraint. Minimising the norm of δ is equivalent to minimising $|\langle p_{i-1}, \delta \rangle| = \xi$. The separating chords are therefore of minimal size for $\xi = 1$. Note also that the constraint $\langle p_{i-1}, \delta \rangle = -1$ ensures, by Bézout, that δ has coordinates which are prime between them. We therefore deduce that $\delta = \pm \frac{p_i \wedge m_i}{\text{pgcd}(p_i \wedge m_i)}$. Now

$$\begin{aligned} \langle p_i \wedge m_i, p_{i+1} \rangle &= \det(p_i, m_i, p_{i+1}) \\ &= \det(p_i, p_{i-1} + p_{i+1}, p_{i+1}) \\ &= \det(p_i, p_{i-1}, p_{i+1}) = \pm 1 \end{aligned}$$

Therefore, by Bézout, $p_i \wedge m_i$ has gcd 1 from which $\delta = \pm(p_i \wedge m_i)$. The vector δ thus obtained is therefore the common direction vector for all planes of normal vectors v on the median $[p_i, m_i]$.

Let us now look at the system characterising the δ_+ chord. Since the notion of a chord is antipodal invariant, we can look at the opposite of this system.

$$(S_+) : \begin{cases} \langle p_i, \delta_+ \rangle &= -\langle p_i, e_3 \rangle \\ \langle m_i, \delta_+ \rangle &= -\langle m_i, e_3 \rangle \\ \langle p_{i+1}, \delta_+ \rangle &= \xi - \langle p_{i+1}, e_3 \rangle \end{cases}$$

which is rewritten, each time adding the scalar product on the right,

$$(S'_+) : \begin{cases} \langle p_i, \delta_+ + e_3 \rangle &= 0 \\ \langle m_i, \delta_+ + e_3 \rangle &= 0 \\ \langle p_{i+1}, \delta_+ + e_3 \rangle &= \xi \end{cases}$$

Subtracting line 3 from line 2 gives us

$$\begin{cases} \langle p_i, \delta_+ + e_3 \rangle &= 0 \\ \langle m_i, \delta_+ + e_3 \rangle &= 0 \\ \langle p_{i-1}, \delta_+ + e_3 \rangle &= -\xi \end{cases}$$

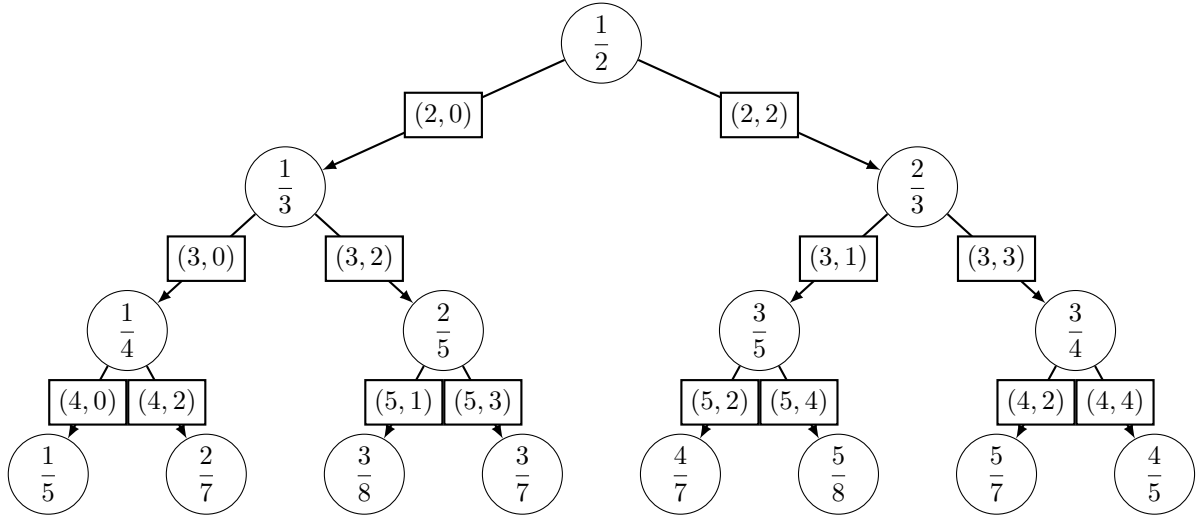


Figure 12 Stern-Brocot tree labelled by separating chords

Comparing with the (S_-) system, we obtain that $\delta_+ + e_3 = \delta_- - e_3 = \delta$. The separating chords therefore have a simple geometric interpretation in terms of the director vectors of the arithmetic hyperplane.

5 All-dimensional recognition algorithm

Using the separating chords, we are now able to go down the Stern-Brocot tree. We thus derive an algorithm for recognising pieces of hyperplanes in any dimension. The algorithm below is presented in dimension 3 to simplify understanding, but can be extended without difficulty to any finite dimension. [By symmetry, we return to the study of the pieces included in the first 48th of space, i.e. those admitting a normal vector with increasing positive coordinates.](#)

Before looking at the tricky points of the algorithm (in red), Let us look at 2 executions of it on pieces of square planes with parameters $v = (4, 7, 11), \mu = 2, \text{size} = 12$ and $v = (7, 8, 10), \mu = 8, \text{size} = 10$ in Figure 13 and 14. Note that the algorithm can also be applied to any piece of plan, not necessarily rectangular or connected. The detected chords are indicated in the plane. The sub-triangle to descend to is indicated by a point in the plane. In the second example,

the chords $(7, -10, 4)$ and $(9, -9, 0)$ belong to the whole $\mathbb{P}(v, \mu)$ plane but not to the piece shown, which is too small for this. The algorithm therefore fails and returns ‘Inconsistency in the chords’, even though the piece is indeed that of a plane. Finally, the algorithm may only detect 2 chords and no third: this means that there is a dividing line and that there are 2 possible sub-triangles. In this case, the algorithm has found a first direction vector of the v^\perp network. We then have two options:

- you can ignore this information and choose one of the 2 sub-triangles. This option simplifies the writing of the algorithm but is less efficient;
- we can keep the first direction vector and perform a Stern-Brocot in dimension 2 on the median concerned. This option is more respectful of the geometry of the plane but can be more technical to deal with in proofs.

Our algorithm has a few tricky points that need to be explained in detail.

In its current version, the algorithm necessarily stops, either because it reaches the parameters of the plane under consideration, or because the separating chords it has to deal with lead to inconsistency. Unfortunately, it can happen that the separating chords are so large that they leave the part of the plane under consideration. The ‘While’

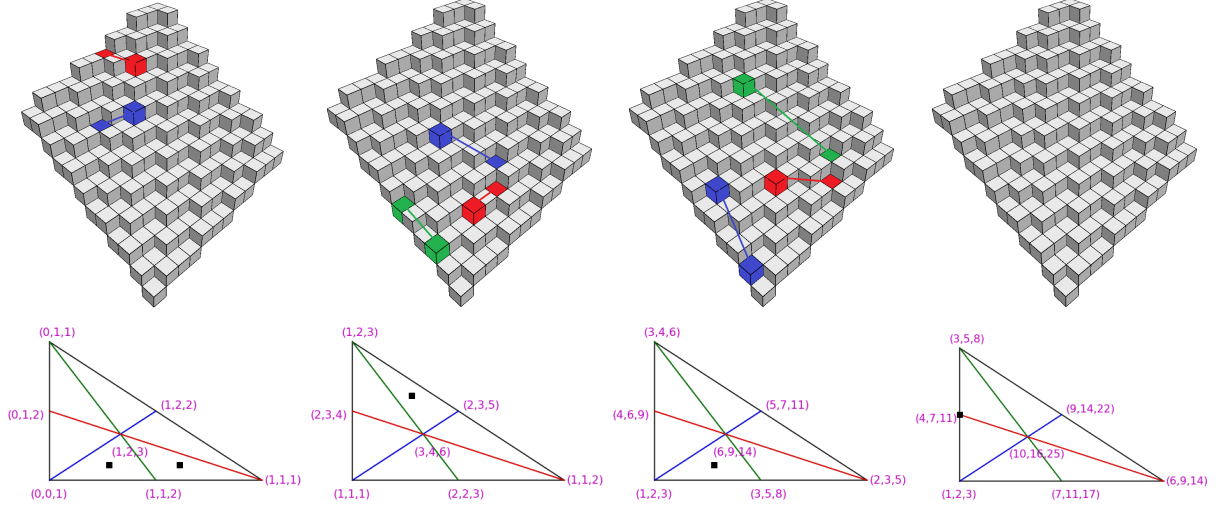


Figure 13 Run of the algorithm on the piece of square plane with normal vector $(4, 7, 11)$, shift 2 and size 12. The algorithm returns the correct result.

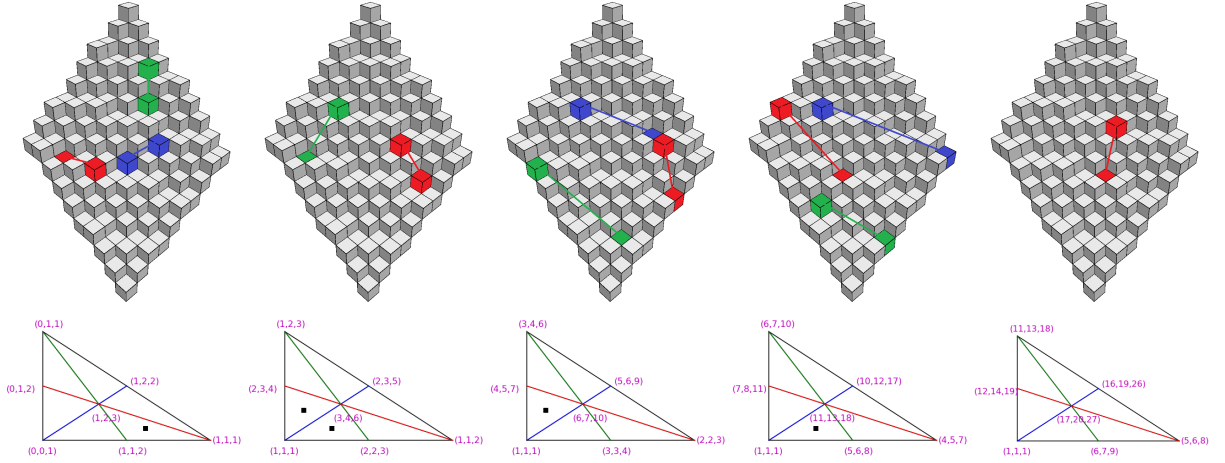


Figure 14 The algorithm runs on the piece of square plane with normal vector $(7, 8, 10)$, shift 2 and size 10. The algorithm detects an inconsistency in the chords because the piece is too small.

stopping condition deserves a simple bound on the coordinates of v . In dimension 2, the theorem 3.3 ensures that $\|v\|_\infty$ is bounded by the diameter of S . In dimension 3, the leaning points algorithm presented in [23] is used to obtain a bound for rectangular pieces of plane. Finally, the simplex algorithm used in [8] is used to obtain a bound in the general case, but much larger. This part of our algorithm is therefore easy to modify, by adding a bound on $\|v\|_\infty$.

Detecting chords in a finite set is much more problematic. First of all, the naive method is anything but optimal (all pairs of points are tested). However, this step can be improved by taking into account only the leaning points of each level (zone of constant z height), thus bringing us back to a constant number of points per level. However, after a certain stage, the chords under consideration force us to leave the piece S . We therefore observe the same problem as with plane probing algorithms (see [19] for the first version). Generally speaking, the deeper the algorithm dives into the Stern-Brocot tree, the larger the separating

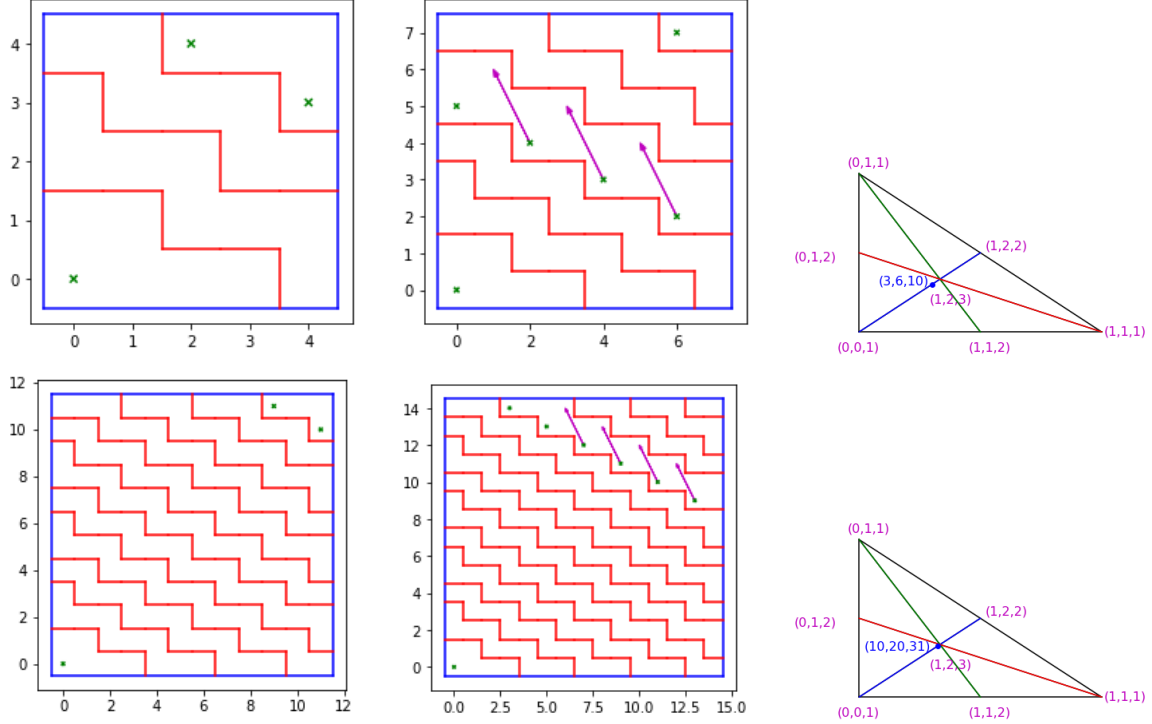


Figure 15 Illustration of pieces, for $n = 3$ and $n = 10$, of vector $v = (n, 2n, 3n + 1)$, shift $\mu = 0$ and sizes $n + 2$ (left) and $n + 5$ in the centre. The red lines correspond to the plane level limits. The lower leaning points of the plane are shown in green. On the left, the chord $\delta \stackrel{\text{def}}{=} (1, -2, 0)$ is not detected because the piece is too small. In the centre, this chord finally appears (in magenta). On the right, we represent the vector v (blue dot) at the first step of the Stern-Brocot tree. Its proximity to the green median, separated by δ , means that very large chunks have to be considered when descending.

chords become and the greater the risk of them leaving the study space. When the algorithm returns ‘Inconsistency in the chords’, there are two possible outcomes:

- the piece under consideration is not a piece of plane. Such an entry inevitably leads to the set of separating chords that appear in the plane not having an associated sub-triangle (see Figure 10 (bottom)), hence the inconsistency.
- the piece under consideration is indeed a piece of plane but is too small for certain separating chords to appear on the piece. The chord inconsistency detected simply means that the algorithm cannot conclude without considering more points.

Note that, on a sufficiently large piece of plane (a fortiori on an infinite plane), the second case mentioned cannot occur. Therefore, the reference ‘Incoherence in the chords’ implies that the input

is not a plane. However, the ‘large enough’ character is difficult to quantify without knowing in advance the vector v of the minimal parameters of the hyperplane. In fact, there are arbitrarily large pieces of planes that lead the algorithm to fail in its first stage. For example, if we take $v = (n, 2n, 3n + 1)$ and $\mu = 0$ for $n \in \mathbb{N}^*$, and consider a square piece S of size $n + 2$ (see Figure 15), we reach a deadlock as early as the triangle of increasing Stern-Brocot coordinates. To move on to the next stage, the algorithm should detect the chord $(1, -2, 0)$ which is separating the green median linking $(0, 1, 1)$ and $(1, 1, 2)$. However, as the vector v is too close to this median, the δ chord (which amounts to being able to make a vertical displacement of a rider on the same landing) only appears for large chunk sizes.

The set of solution parameters of the problem forms a convex. In dimension 2, this property demonstrates that the algorithm will necessarily stop at the minimal parameters, and that there is

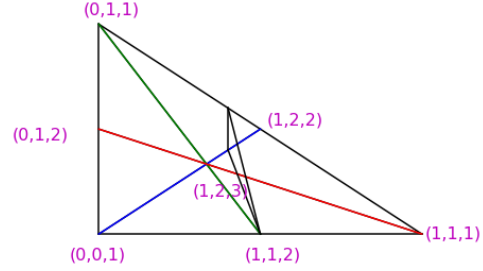
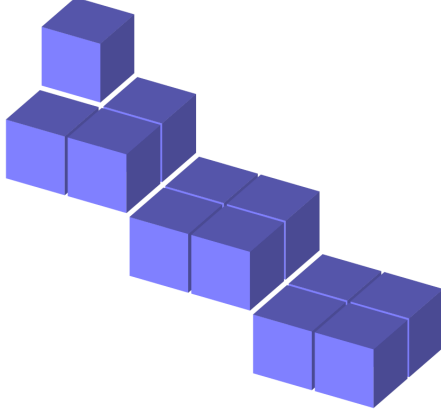


Figure 16 Example of a rectangular piece $\mathbb{P}((1, 4, -7), 0)$ where the set of solutions (black open triangle) intersects two medians of the Stern-Brocot triangle

uniqueness of the descent branch in the algorithm. In dimension 3, a convex may not intersect any remarkable point of the triangle and may straddle several zones. There are therefore configurations where the zone to be chosen for descent is not unique. This problem is illustrated in Figure 16. Consider the piece S of the plane $\mathbb{P}((1, 4, -7), 0)$ with support $\llbracket 0, 1 \rrbracket \times \llbracket 0, 5 \rrbracket$. The set of admissible vectors for S is indicated by a small black triangle straddling 3 zones. In particular, no separating chord can be detected for either the blue or the red median. However, this problem can be avoided by enlarging the area sufficiently.

6 Relation with plane probing algorithms

The algorithm proposed above for recognising pieces of digital plane shares some similarities with *plane probing* algorithms in 3D (see [19] for the first plane probing algorithm). Plane probing algorithms have proved to be useful for determining the exact characteristics of a given digital plane, but may also be used for analysing locally the linear geometry of digital shapes (e.g. see [20]). The input of plane probing algorithms is slightly different from our algorithm. Instead of considering a set of integer points, they assume to have a starting reference frame (an initial cube with one corner inside and one corner outside the shape) and an oracle answering the question "is some

point in the plane/shape?". The output is either the exact characteristics of the digital plane or an approximation of the local tangent plane of the shape.

More specifically we study first the relation between the H -algorithm [18] and our proposed algorithm, since we demonstrate here that the H -algorithm can also be interpreted as a descent in the 3D Stern-Brocot tree. Let us briefly sum up the H -algorithm.

Relation with H -algorithm. We assume here that the H -algorithm analyse a standard digital plane \mathbb{P} with normal vector N and null shift, i.e. $\mathbb{P} \stackrel{\text{def}}{=} \{x \in \mathbb{Z}^d \mid 0 \leq \langle x, N \rangle < \|N\|_1\}$. Notations are illustrated on Figure 17 made from a program by Tristan Roussillon. Point $p \stackrel{\text{def}}{=} (0, 0, 0)$ is then a lower leaning point to \mathbb{P} (and belongs to \mathbb{P}) while point $q \stackrel{\text{def}}{=} (1, 1, 1)$ is just exterior to \mathbb{P} (since $\langle q, N \rangle = \|N\|_1$). The algorithm evolves through iterations a triangle T with vertices t_1, t_2, t_3 , all belonging to \mathbb{P} and initialised with the canonical basis: $\forall i \in \mathbb{Z}/3\mathbb{Z}, v_i \stackrel{\text{def}}{=} e_i$. From now on, to simplify the exposition, we write \mathbb{Z}_3 for $\mathbb{Z}/3\mathbb{Z}$ (the 3 possible indices, taken modulo 3). We also set $\forall i \in \mathbb{Z}_3, u_i \stackrel{\text{def}}{=} q - t_i$ and $d_i \stackrel{\text{def}}{=} t_{i+1} - t_i = u_i - u_{i+1}$.

The normal vector to the plane \mathbb{P} is estimated from the normal vector to the triangle T . Therefore, at each iteration, the estimated normal is $\hat{N}(T) \stackrel{\text{def}}{=} d_i \wedge d_{i+1}$ (for an arbitrary $i \in \mathbb{Z}_3$).

Algorithm 5: Plane recognition

Input: $S \subset \mathbb{Z}^3$ finite in the first 48th of space

Output: Decides whether S is a discrete plane and, if so, returns parameters corresponding to a plane containing it

PLANE-RECOGNITION(S):

$p_1, p_2, p_3 \leftarrow (0, 0, 1), (0, 1, 1), (1, 1, 1)$
(extremities) ;

while *TRUE* **do**

$m_1, m_2, m_3, c \leftarrow$

$p_2 + p_3, p_1 + p_3, p_1 + p_2, p_1 + p_2 + p_3$
(middle and central points) ;

for $v \in \{m_1, m_2, m_3, c\}$ (test of the different points of the triangle) **do**

$(p_{\min}, p_{\max}) \leftarrow$

EXTREMAL-REMAINDERS(S, v) ;

if $\langle p_{\max} - p_{\min}, v \rangle < \|v\|_{\infty}$ **then**

$\mu \leftarrow -\langle p_{\min}, v \rangle$;

Return v, μ

Calculate the separations $(\delta_{-}^{(i)}, \delta_{+}^{(i)})$ for each p_i ;

See which chords appear in S ;

if the chords observed are incoherent (according to Figure 10 (bottom))

then

Return ‘Inconsistency in the chords’

Choose the sub-triangle corresponding to the observed chords (according to Figure 10 (bottom)) and update

p_1, p_2, p_3 ;

Algorithm 6: Extremal-remainders

Input: $S \subset \mathbb{Z}^3$ finite, $v \in \mathbb{Z}^3$

Output: Calculate the minimum and maximum remainders of the points of S according to v .

EXTREMAL-REMAINDERS(S, v) :

$p_{\min}, p_{\max} \leftarrow$ first element of S ;

for p in S **do**

if $\langle p, v \rangle < \langle p_{\min}, v \rangle$ **then**

$p_{\min} \leftarrow p$;

if $\langle p, v \rangle > \langle p_{\max}, v \rangle$ **then**

$p_{\max} \leftarrow p$;

Return p_{\min}, p_{\max} ;

Last, we define the three normal vectors ν_i to the triangles (q, t_{i+1}, t_{i+2}) , so $\nu_i \stackrel{\text{def}}{=} u_{i+1} \wedge u_{i+2}$. An invariant of the algorithm is that the normal vector N to the plane \mathbb{P} belongs to the cone generated by the three vectors ν_i .

The main idea of the H -algorithm is to update the triangle T one vertex at a time, i.e. some extremity t_k of T is moved to a new location t' . A constraint is that the new location t' must still belong to \mathbb{P} , while the way this location is chosen ensures that the triangle is moved closer to the upper leaning plane of \mathbb{P} . Progressively the triangle T advances to the upper limit of \mathbb{P} and hence becomes parallel to it.

Concretely, the point t' is chosen within the set $q + \mathcal{V}_H \stackrel{\text{def}}{=} \{q \pm d_j\}_{j \in \mathbb{Z}_3}$ (whose shape forms an hexagon, hence the name of H -algorithm), and must also belong to \mathbb{P} . The algorithm tests which points of $q + \mathcal{V}_H$ are also in \mathbb{P} , using the oracle. When the set $(q + \mathcal{V}_H) \cap \mathbb{P}$ is empty, the triangle T form an affine network of upper leaning points to \mathbb{P} and $\hat{N}(T)$ is the sought normal N (in case of recognising an infinite rational plane). Otherwise we consider some point of $(q + \mathcal{V}_H) \cap \mathbb{P}$. This new point t' (say) has either the form $q + d_j = q + t_j - t_{j+1} = t_j + u_{j+1}$ (so we move t_j to t') or the form $q - d_j = q + t_{j+1} - t_j = t_{j+1} + u_j$ (so we move t_{j+1} to t'). It follows there exists $j \in \mathbb{Z}_3, k \in \mathbb{Z}_3, j \neq k$ with $t'_k = t_k + u_j$.

Without loss of generality, we perform computations when $k = 1$ and $j = 2$ (computations are similar for other values). The algorithm replaces v_1 with v'_1 while keeping the other extremities. The new triangle T' has thus the vertices $(t'_1, t'_2, t'_3) \stackrel{\text{def}}{=} (t_1 + u_2, t_2, t_3)$. Other vectors are modified as follows:

$$u'_1 \stackrel{\text{def}}{=} q - t'_1 = q - t_1 - u_2 = u_1 - u_2$$

$$u'_2 \stackrel{\text{def}}{=} q - t'_2 = q - t_2 = u_2$$

$$u'_3 \stackrel{\text{def}}{=} q - t'_3 = q - t_3 = u_3$$

$$\nu'_1 \stackrel{\text{def}}{=} u'_2 \wedge u'_3 = u_2 \wedge u_3 = \nu_1$$

$$\nu'_2 \stackrel{\text{def}}{=} u'_3 \wedge u'_1 = u_3 \wedge (u_1 - u_2) = \nu_2 + \nu_1$$

$$\nu'_3 \stackrel{\text{def}}{=} u'_1 \wedge u'_2 = (u_1 - u_2) \wedge u_2 = \nu_3$$

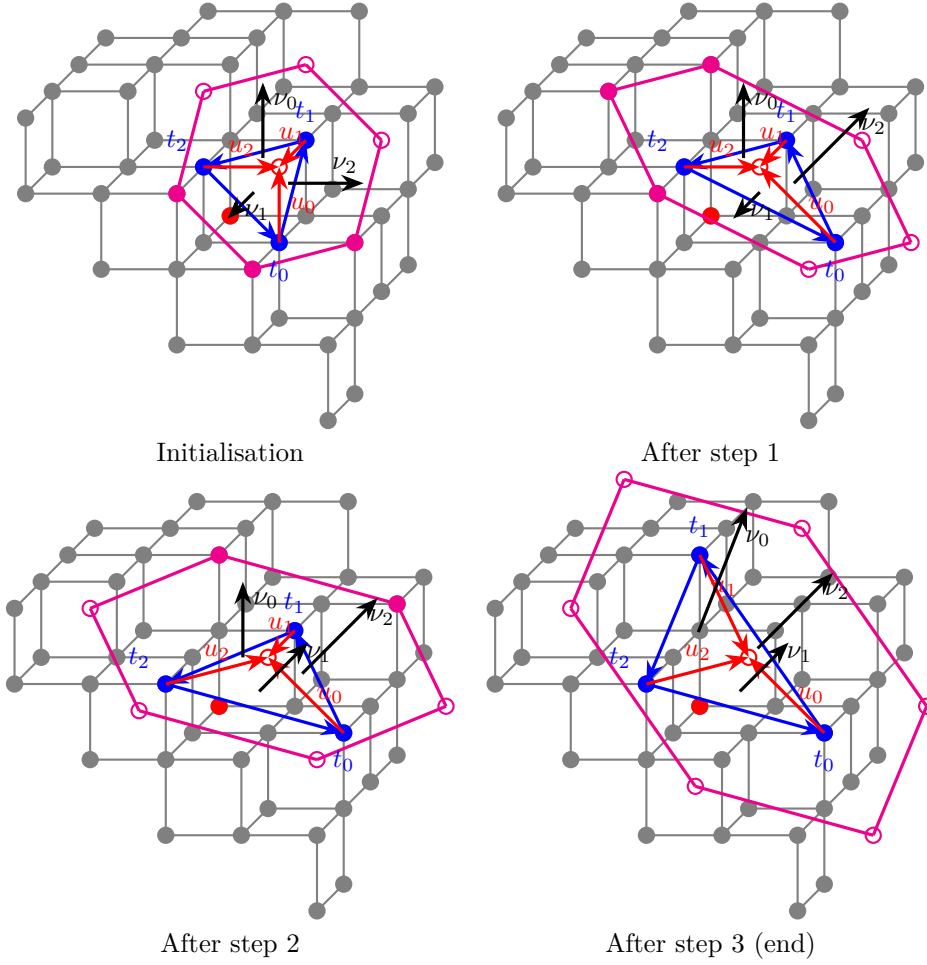


Figure 17 Illustration of plane probing H -algorithm: notations and evolution on a plane with normal $(2, 3, 4)$. The algorithm is initialised in a corner of the plane (a lower leaning point drawn as filled red disk). The outer diagonal point is just outside the plane (denoted q in the text) and is an invariant of the H -algorithm. The initial triangle $(t_0 t_1 t_2)$ is set at the tips of the corner, with ascending vectors $u_i := q - t_i$, $i \in \mathbb{Z}_3$. Vectors $d_i := t_{i+1} - t_i$ are drawn in blue. The three normal vectors ν_i to the triangles (q, t_{i+1}, t_{i+2}) are drawn in black. The hexagon of points $q + \mathcal{V}_H$ tested by the H -algorithm is drawn in magenta. At each iteration, a point is chosen among this hexagon to replace some vertex t_j , this point must also belong to the plane (filled magenta disk). The algorithm stops when every point of the hexagon is outside the plane (see last step). Note how the cone of black arrows tends toward the exact normal vector of the plane.

The triangle (ν_1, ν_2, ν_3) where the exact normal is lying is thus transformed into the smaller triangle $(\nu_1, \nu_1 + \nu_2, \nu_3)$ (see Figure 19, left side). It is thus clear that the H -algorithm splits the triangle in two along a median line, similarly to our proposed algorithm where each chord splits the Stern-Brocot triangle tree along a median line. However it does not follow exactly the same descent since our algorithm extracts one of the six sub-triangles leaning to the median point. We display on Figure 18 how the H -algorithm repetitively splits in two the Stern-Brocot triangle on the same example as Figure 17. The triangle of

possible normal vectors gets closer to the target normal at each iteration. The plane being rational, the algorithm stops when the median of the triangle is the sought normal vector.

In order to get a similar descent to our proposed chord algorithm, it is possible to slightly adapt the H -algorithm. It suffices to choose two well-chosen points of $q + \mathcal{V}_H$ and to move two points of T instead of one. This splits the triangle along two median lines, which corresponds to one descent in the Stern-Brocot tree (see Figure 19, right side). Of course it means that $(q + \mathcal{V}_H) \cap$

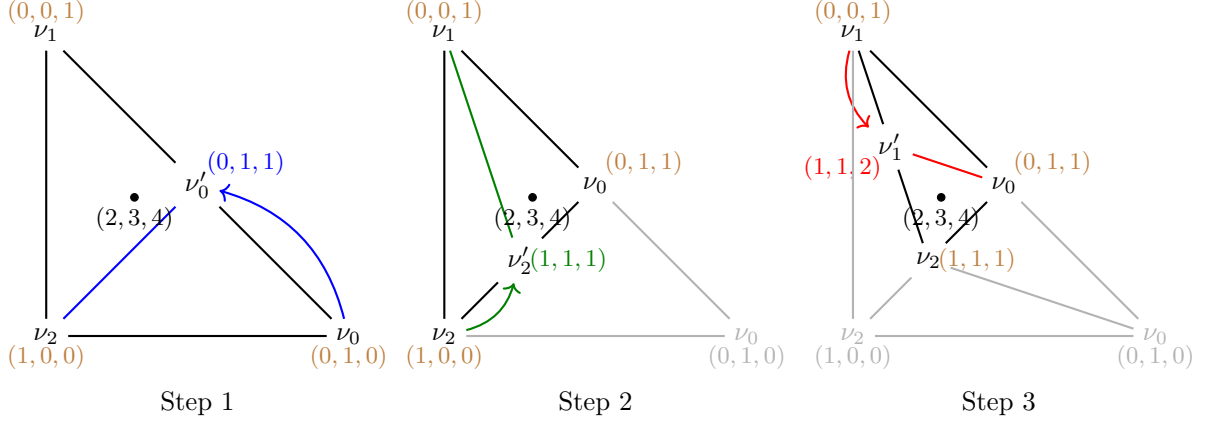


Figure 18 Illustration of plane probing H -algorithm: evolution of normal estimates on a plane with normal $(2, 3, 4)$. We display here the evolution of the triangle of possible normal vectors (ν_0, ν_1, ν_2) during the plane probing illustrated on Figure 17. We see that the triangle is cut in two at each step, but the direction of cutting depends on which vertex is moved. The target normal $(2, 3, 4)$ is displayed as the black disk.

\mathbb{P} contains at least two points, such that they correspond to two different extremities t_i and t_j .

This was for instance the case for the example of plane probing with normal $(2, 3, 4)$ when gathering the two first steps, as can be seen on Figures 17 and 18.

Relation with R - and L -algorithm. Similar computations can be done for the variants R and L of plane probing algorithms, presented in a series of paper by the same authors [20, 22]. In these variants, the new point t' is sought in the neighbourhoods $q + \mathcal{V}_R$ and $q + \mathcal{V}_L$ (respectively for the R and L variant), with

$$q + \mathcal{V}_R \stackrel{\text{def}}{=} \{q \pm d_k + \lambda u_{k+2}\},$$

$$q + \mathcal{V}_L \stackrel{\text{def}}{=} \{q \pm d_k + \lambda u_{k+2} + \mu u_{k+1}\},$$

and λ and μ are both integers with $\lambda \geq 1, \mu \geq 1$. Let us compute how these 2 algorithms evolve the triangle of possible normals.

Using symmetry and renumbering, we can reduce the R -algorithm to the case where t_1 is replaced by $t'_1 \stackrel{\text{def}}{=} t_1 + d_1 + \lambda u_3 = t_1 + u_2 + \lambda u_3$. Similar computations as above give:

$$u'_1 \stackrel{\text{def}}{=} q - t'_1 = q - t_1 - u_2 - \lambda u_3 = u_1 - u_2 - \lambda u_3$$

$$u'_2 \stackrel{\text{def}}{=} q - t'_2 = q - t_2 = u_2$$

$$u'_3 \stackrel{\text{def}}{=} q - t'_3 = q - t_3 = u_3$$

$$\nu'_1 \stackrel{\text{def}}{=} u'_2 \wedge u'_3 = u_2 \wedge u_3 = \nu_1$$

$$\nu'_2 \stackrel{\text{def}}{=} u'_3 \wedge u'_1 = u_3 \wedge (u_1 - u_2 - \lambda u_3) = \nu_2 + \nu_1$$

$$\nu'_3 \stackrel{\text{def}}{=} u'_1 \wedge u'_2 = (u_1 - u_2 - \lambda u_3) \wedge u_2 = \nu_3 + \lambda \nu_1$$

The triangle of possible normal vectors (ν_1, ν_2, ν_3) is thus replaced by $(\nu_1, \nu_1 + \nu_2, \nu_1 + \lambda \nu_3)$, as illustrated on Figure 20, middle.

Finally we can also reduce the L -algorithm to the case where t_1 is replaced by $t'_1 \stackrel{\text{def}}{=} t_1 + d_1 + \lambda u_3 + \mu u_2 = t_1 + (\mu + 1)u_2 + \lambda u_3$. Updated values of vectors are (intermediate computations are skipped but similar to above):

$$u'_1 \stackrel{\text{def}}{=} q - t'_1 = u_1 - (\mu + 1)u_2 - \lambda u_3$$

$$u'_2 \stackrel{\text{def}}{=} q - t'_2 = u_2$$

$$u'_3 \stackrel{\text{def}}{=} q - t'_3 = u_3$$

$$\nu'_1 \stackrel{\text{def}}{=} u'_2 \wedge u'_3 = \nu_1$$

$$\nu'_2 \stackrel{\text{def}}{=} u'_3 \wedge u'_1 = \nu_2 + (\mu + 1)\nu_1$$

$$\nu'_3 \stackrel{\text{def}}{=} u'_1 \wedge u'_2 = \nu_3 + \lambda \nu_1$$

The triangle of possible normal vectors (ν_1, ν_2, ν_3) is thus replaced by $(\nu_1, (\mu + 1)\nu_1 + \nu_2, \nu_1 + \lambda \nu_3)$, as illustrated on Figure 20, right.

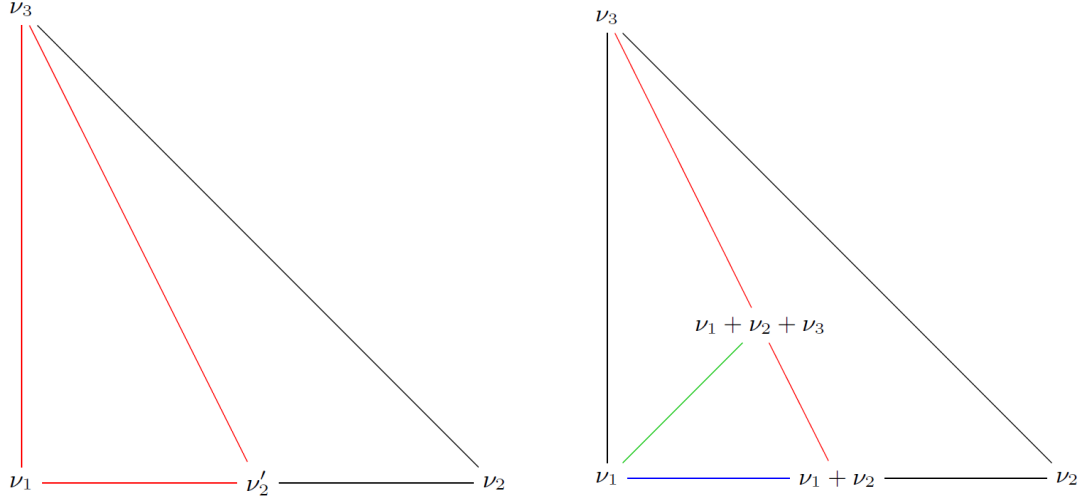


Figure 19 Splitting of the triangle of possible normal vectors for the H -algorithm (left) and for the modified H -algorithm (right).

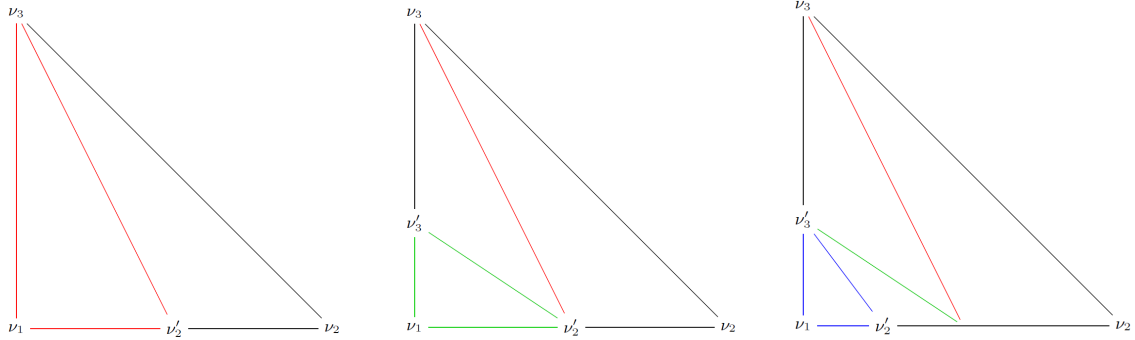


Figure 20 Splitting of the triangle of possible normal vectors for the H -algorithm (left, in red), for the R -algorithm (middle, in green), and for the L -algorithm (right, in blue).

To conclude this section, if the H -algorithm can be slightly modified to induce a similar descent in the Stern-Brocot tree as our proposed algorithm, the R - and L -algorithms seem more difficult to interpret. Indeed, the cuts in the triangle cone of normal vectors are not done according to the median point but along directions that do not respect the structure of the tree. However, they restrict faster the set of possible normal vectors, in cases where λ or μ are not equal to 1.

7 Conclusion

This paper presents a new approach for the detection of pieces of arithmetic hyperplanes from the Stern-Brocot tree. In dimension 2, we propose an algorithm that returns the minimal parameters of a piece of any discrete line S , in time $O(nD)$ in the worst case and in $O(n \ln^2(D))$ on average, where n denotes the size of S and D its diameter. In the case of discrete segments, we modify the algorithm to obtain a linear and incremental version, thus recovering the recognition algorithm of I. Debled-Rennesson and J.-P. Réveillac via the Stern-Brocot tree.

We then extend our algorithm to any finite dimension, in order to recognise arithmetic hyperplane pieces. The idea is based on the extension of the Stern-Brocot tree to any dimension proposed in [21] and on the notion of separating chords. The presence of separating chords in the hyperplane piece makes it possible to divide a Stern-Brocot simplex into several sub-simplexes, and to choose which sub-simplex to descend into. However, our algorithm has a number of limitations, particularly with regard to overflow problems. In some cases, the chords considered become too large to be detected in the piece under study. Chord detection is a key stage of the algorithm that needs to be optimised, for example by considering the leaning points for each plane step.

The separating chords used in our recognition algorithm characterise the zone in which the normal vector v of the hyperplane lives in the Stern-Brocot tree. These chords thus have a geometric interest in the structure of the hyperplane. In dimension 2, the separating chords have a simple expression that fully characterises intervals of slope. In higher dimensions, we propose a combinatorial study of the chords, observing in particular their link with the directing vectors of the plane. Finally, we interpret plane probing algorithms as cuts in the cone of normals. In particular, a slightly modified H algorithm allows us to go down the Stern-Brocot tree while respecting its structure.

Beyond the recognition of arithmetic hyperplanes, the multidimensional Stern-Brocot tree is a very powerful combinatorial tool for efficiently enumerating d -uplets of integers that are prime to each other. In dimension 2, many parallels have been drawn between discrete lines and this tree. These parallels deserve to be explored further in higher dimensions.

8 Acknowledgements

The authors thank Tristan Roussillon for its tool to visualize the plane probing algorithm, which was used to make Figure 17.

References

- [1] Eric Andres, Raj Acharya, and Claudio Sibata. Discrete analytical hyperplanes. *Graphical Models and Image Processing*, 59(5):302–309, 1997.
- [2] Sebastián Barbieri and Sébastien Labbé. Indistinguishable asymptotic pairs and multi-dimensional sturmian configurations. *arXiv preprint arXiv:2204.06413*, 2022.
- [3] Valérie Berthé. Discrete geometry and symbolic dynamics. In *The Kiselmanfest: An International Symposium in Complex Analysis and Digital Geometry*, 2006.
- [4] Valentin Brimkov, David Coeurjolly, and Reinhard Klette. Digital planarity - A review. *Discrete Applied Mathematics*, 155(4):468–495, 2007.
- [5] Achille Brocot. *Calcul des rouages par approximation: nouvelle méthode*. Imprimerie Paul Dupont, 45 rue de Grenelle Saint-Honoré, Paris, 1862.
- [6] Alfred M Bruckstein. Self-similarity properties of digitized straight lines. *Contemporary Mathematics*, 119:1–20, 1991.
- [7] Lilian Buzer. *Reconnaissance des plans discrets: simplification polygonale*. PhD thesis, Clermont-Ferrand 1, 2002.
- [8] Emilie Charrier and Lilian Buzer. An efficient and quasi linear worst-case time algorithm for digital plane recognition. In *International Conference on Discrete Geometry for Computer Imagery*, pages 346–357. Springer, 2008.
- [9] Isabelle Debled-Rennesson. *Etude et reconnaissance des droites et plans discrets*. PhD thesis, Université Louis Pasteur (Strasbourg)(1971-2008), 1995.
- [10] Isabelle Debled-Rennesson and Jean-Pierre Reveillès. A linear algorithm for segmentation of digital curves. *International Journal of Pattern Recognition and Artificial Intelligence*, 9(4):635–662, 1995.
- [11] Leo Dorst and Arnold WM Smeulders. Discrete representation of straight lines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (4):450–463, 1984.
- [12] Isabelle Debled et Jean-Pierre Reveillès. An incremental algorithm for digital plane recognition. In *4th International Conference on*

- Discrete Geometry for Computer Imagery*, 1994.
- [13] Thomas Fernique. *Pavages, fractions continues et géométrie discrète*. PhD thesis, Université Montpellier II-Sciences et Techniques du Languedoc, 2007.
 - [14] Herbert Freeman. Boundary encoding and processing. *Picture processing and psychopictorics*, pages 241–263, 1970.
 - [15] Yan Gérard, Isabelle Debled-Rennesson, and Paul Zimmermann. An elementary digital plane recognition algorithm. *Discrete Applied Mathematics*, 151(1-3):169–183, 2005.
 - [16] Reinhard Klette and Aziel Rosenfeld. Digital straightness - a review. *Discrete Applied Mathematics*, 139(1-3):197–230, 2004.
 - [17] Bastien Laboureur and Isabelle Debled-Rennesson. Recognition of arithmetic line segments and hyperplanes using the Stern-Brocot tree. In *International Conference on Discrete Geometry and Mathematical Morphology*, pages 16–28. Springer, 2024.
 - [18] Jacques-Olivier Lachaud, Xavier Provençal, and Tristan Roussillon. Computation of the normal vector to a digital plane by sampling significant points. In *International Conference on Discrete Geometry for Computer Imagery*, pages 194–205. Springer, 2016.
 - [19] Jacques-Olivier Lachaud, Xavier Provençal, and Tristan Roussillon. An output-sensitive algorithm to compute the normal vector of a digital plane. *Theoretical Computer Science*, 624:73–88, 2016.
 - [20] Jacques-Olivier Lachaud, Xavier Provençal, and Tristan Roussillon. Two plane-probing algorithms for the computation of the normal vector to a digital plane. *Journal of Mathematical Imaging and Vision*, 59(1):23–39, 2017.
 - [21] Håkan Lennerstad. The n-dimensional Stern-Brocot tree. *International Journal of Number Theory*, 15(06):1219–1236, 2019.
 - [22] Jui-Ting Lu, Tristan Roussillon, and David Coeurjolly. A new lattice-based plane-probing algorithm. In Étienne Baudrier, Benoît Naegel, Adrien Krähenbühl, and Mohamed Tajine, editors, *Discrete Geometry and Mathematical Morphology*, pages 366–381, Cham, 2022. Springer International Publishing.
 - [23] Mohammed Mostefa Mesmoudi. A simplified recognition algorithm of digital planes pieces. In *Discrete Geometry for Computer Imagery: 10th International Conference*, volume 2301 of *LNCS*, pages 404–416, 2002.
 - [24] Jean-Pierre Reveillès. *Géométrie discrète, calcul en nombres entiers et algorithmique*. PhD thesis, Université Louis Pasteur, 1991.
 - [25] Moritz Stern. Über eine zahlentheoretische funktion. 1858.
 - [26] François De Vieilleville and Jacques-Olivier Lachaud. Revisiting digital straight segment recognition. In *Discrete Geometry for Computer Imagery: 13th International Conference*, volume 4245 of *LNCS*, pages 355–366. Springer, 2006.
 - [27] Andrew C Yao and Donald E Knuth. Analysis of the subtractive algorithm for greatest common divisors. *Proceedings of the National Academy of Sciences*, 72(12):4720–4722, 1975.