New Results on the Distance Between a Segment and $\mathbb{Z}^2$. Application to the Exact Rounding

Vincent Lefèvre

Loria / INRIA Lorraine

Arith'17

June 27–29, 2005

Introduction / Outline

1976: More general case (Hirschberg and Wong).

1997: Find a lower bound on the distance between a segment and $\mathbb{Z}^2$.
I presented a first *efficient* algorithm (with low-level operations).
Complex proof. In fact, *exact* distance on a larger domain.

→ A more geometrical and intuitive proof.

→ A variant/improvement of the algorithm.

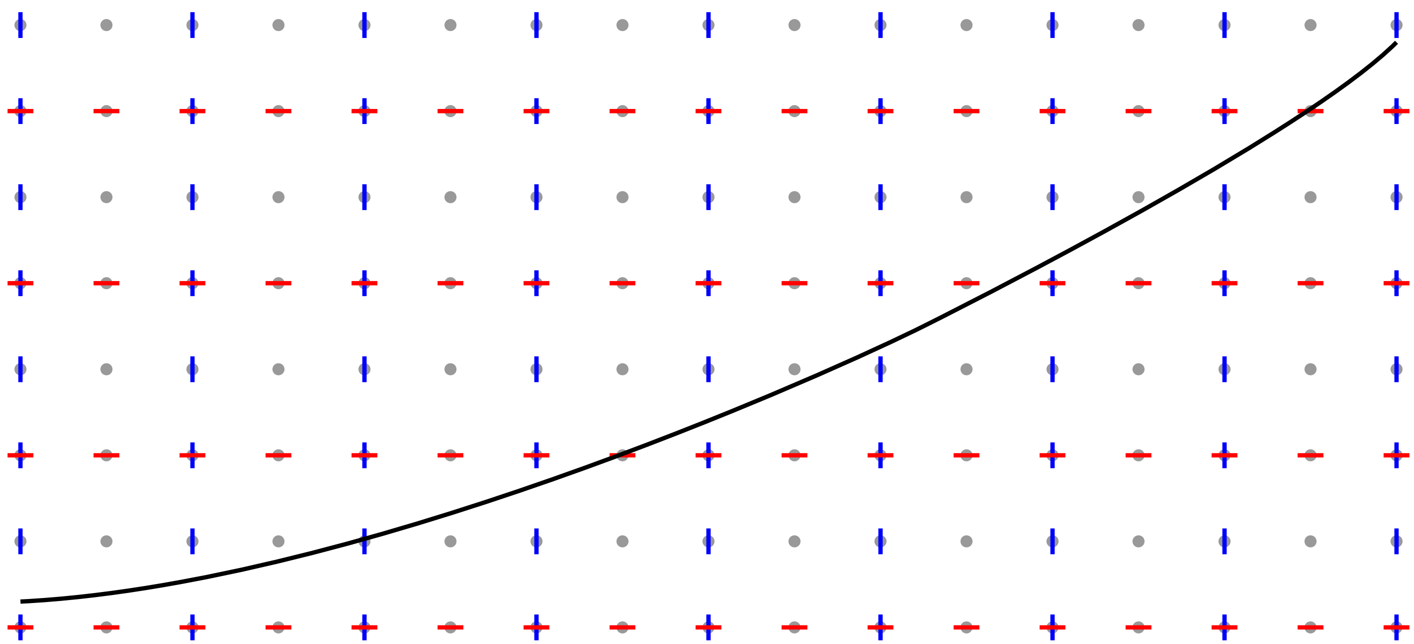
→ Timings (comparisons of various forms of the algorithms).

→ Comparison with an implementation (`wclr22`) of the SLZ algorithm (Arith'16).

The Problem (Without Details)

Goal: the exhaustive test of the elementary functions for the TMD in a fixed precision (e.g., in double precision), i.e. "find the breakpoint numbers x such that $f(x)$ is very close to a breakpoint number".

Breakpoint number: machine number or "half-machine number".

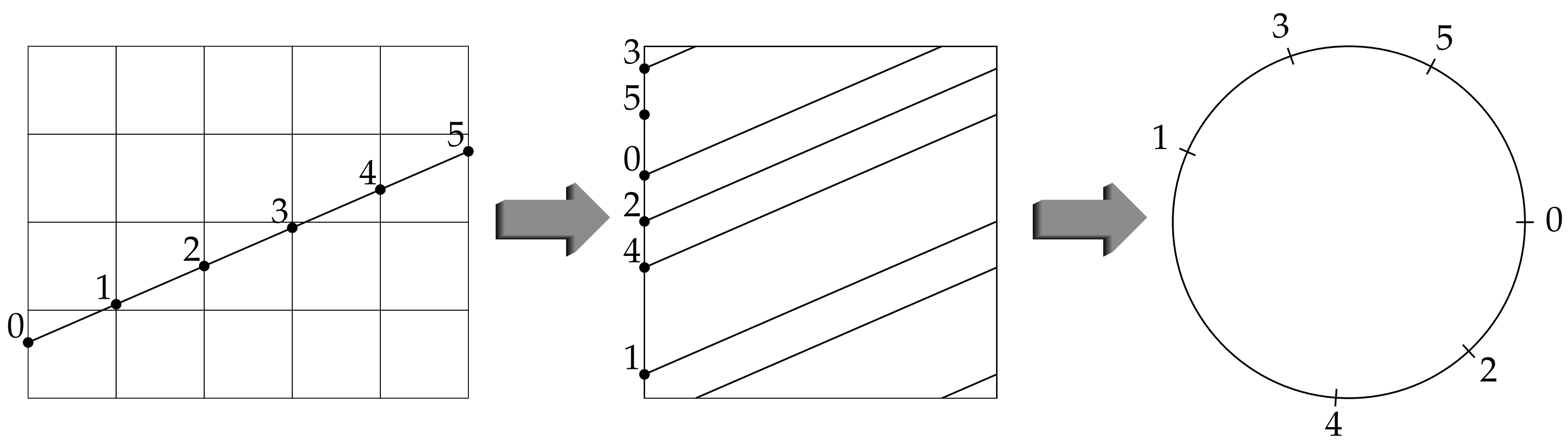


In each interval:

- f approached by a polynomial of degree $1 \rightarrow$ segment $y = b - ax$.
- Multiplication of the coordinates by powers of $2 \rightarrow$ grid $= \mathbb{Z}^2$.

One searches for the values n such that $\{b - n.a\} < d_0$,
where a, b and d_0 are real numbers and $n \in [\![0, N-1]\!]$.

$\{x\}$ denotes the positive fractional part of x.



- We chose a positive fractional part instead of centered.
  $\rightarrow$ An upward shift is taken into account in b and d_0.

- If a is rational, then the sequence $0.a$, $1.a$, $2.a$, $3.a$, $\ldots$ (modulo 1) is periodical.

  $\rightarrow$ This makes the theoretical analysis more difficult.

  $\rightarrow$ In the proof, one assumes a irrational, or equivalently, a rational number $+$ an arbitrary small irrational number.

  But in the implementation, a is rational.

  $\rightarrow$ Extension to rational numbers by continuity.

  $\rightarrow$ Care has been taken with the inequalities (strict or not).

Notations / Properties

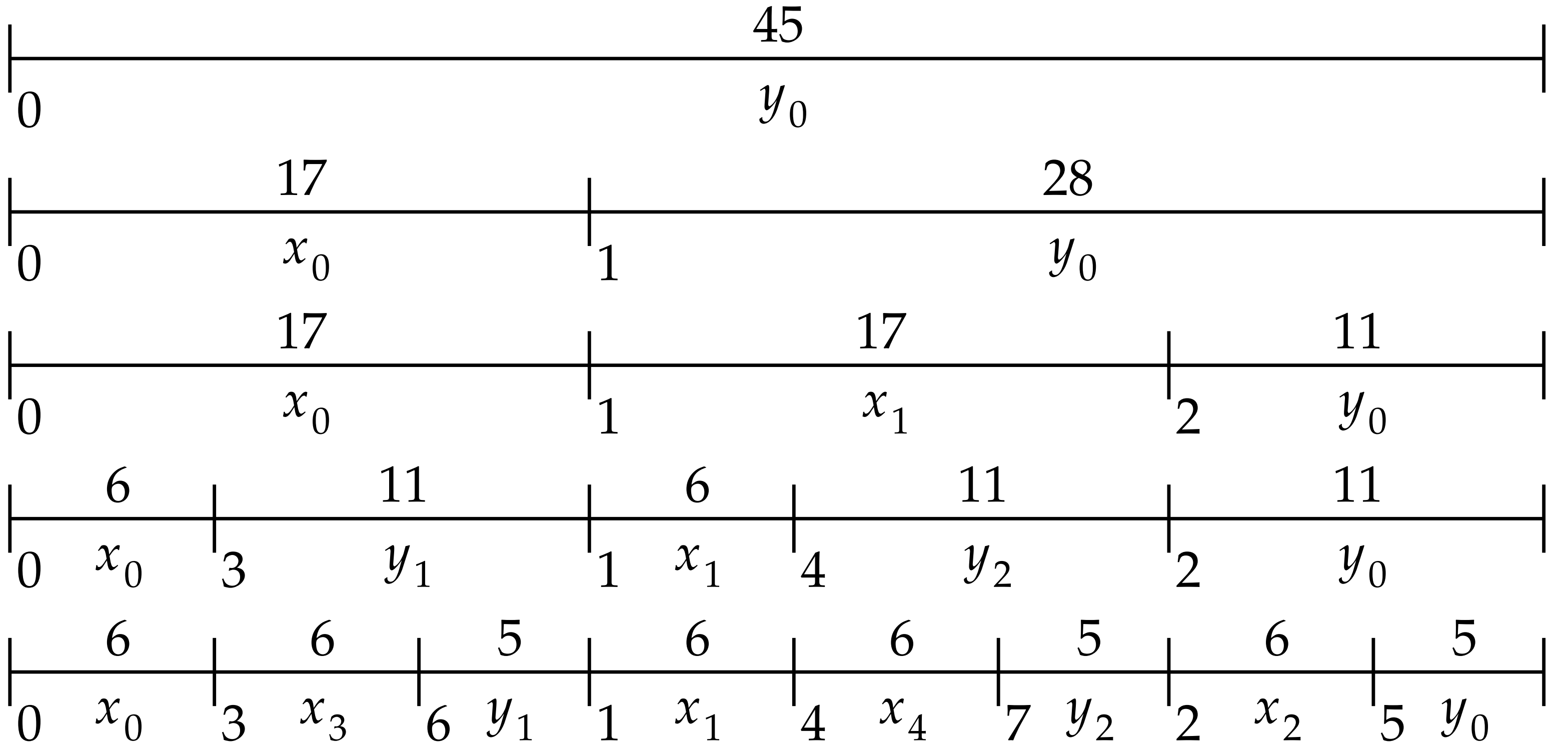
Configuration properties to be proved by induction:

- Intervals $x_0, x_1, \ldots, x_{u-1}$ of length x, where x_0 is the left-most interval and $x_r = x_0 + r.a$ (translation by $r.a$ modulo 1).
- Intervals $y_0, y_1, \ldots, y_{v-1}$ of length y, where y_0 is the right-most interval and $y_r = y_0 + r.a$ (translation by $r.a$ modulo 1).
- Total number of points (or intervals): $n = u + v$.

Initial configuration: $n = 2, u = v = 1$.

From a Configuration to the Next One

- Since a is irrational, $n.a$ is strictly between 2 points of smaller indices, one of which, denoted r is non zero.
- Therefore the points of indices $r-1$ and $n-1$ (obtained by a translation) are adjacent, and their distance ℓ is either x or y.
  $\rightarrow$ Same distance ℓ between the points of indices r and n.
- Thus the new point n splits an interval of length $h = \max(x, y)$ into 2 intervals of respective lengths $\ell = \min(x, y)$ and $h - \ell$.
- The length $h - \ell$ is new, therefore the corresponding interval does not have an inverse image (i.e. by adding $-a$).
- Therefore this interval has as a boundary point of index 0.


45
0
y_0
17
28
0
x_0
1
y_0
17
17
11
0
x_0
1
x_1
2
y_0
6
11
6
11
11
0
x_0
3
y_1
1
x_1
4
y_2
2
y_0
6
6
5
6
6
5
6
5
0
x_0
3
x_3
6
y_1
1
x_1
4
x_4
7
y_2
2
x_2
5
y_0

$\rightarrow$ As a consequence, the point of index n is completely determined.

The other intervals of length h will be split in the same way, one after the other with increasing indices (translations by a).

- Indices of the intervals of length $h - \ell$: these are the indices of the corresponding intervals of length h.
- Indices of the intervals of length ℓ: assume that $\ell = x$ (same reasoning for $\ell = y$); the first interval of length x is obtained by a translation of an old interval of length x (as shown in previous slide), necessarily x_{u-1} (the last one) since the image of x_{i-1} is x_i for all $i < u$. Thus this interval is x_u and we have $x_u = x_0 + u.a$. The next intervals: x_{u+1}, x_{u+2}, etc.

Algorithms

Basic algorithm (1997): returns a lower bound on $\{b - n.a\}$ (in our context, $\geqslant d_0$ in most cases, allowing to immediately conclude that there are no points such that $\{b - n.a\} < d_0$).

New algorithm (mentioned in 1998): returns the index $n < N$ of the first point such that $\{b - n.a\} < d_0$, otherwise any value $\geqslant N$ if there are no such points.

We are interested only in the position of b amongst the other points.
$\rightarrow$ Just keep the necessary data…

The necessary data:

- lengths x and y, numbers u and v of these intervals;
- a binary value saying whether b is in an interval of length x or y;
- the index r of this interval (new algorithm only);
- the distance d between b and the lower boundary of this interval.

Immediate consequence of the properties:

- The lower boundary of an interval x_r has index r.
- The lower boundary of an interval y_r has index $u + r$.

Algorithm (Subtractive Version)

Initialization: $x = \{a\}\,;\ y = 1 - \{a\}\,;\ d = \{b\}\,;\ u = v = 1\,;\ r = 0\,;$

if $(d < d_0)$ **return** 0

Unconditional loop:

if $(d < x)$
- **while** $(x < y)$
  - **if** $(u + v \geqslant N)$ **return** N
  - $y = y - x;\ u = u + v;$
- **if** $(u + v \geqslant N)$ **return** N
- $x = x - y;$
- **if** $(d \geqslant x)$ $r = r + v;$
- $v = v + u;$

else
- $d = d - x;$
- **if** $(d < d_0)$ **return** $r + u$
- **while** $(y < x)$
  - **if** $(u + v \geqslant N)$ **return** N
  - $x = x - y;\ v = v + u;$
- **if** $(u + v \geqslant N)$ **return** N
- $y = y - x;$
- **if** $(d < x)$ $r = r + u;$
- $u = u + v;$

Timings: Notations

- Option `c=`k: subtractions are replaced by a single division when one needs to do at least 2^k subtractions without modifying the value d (—: subtractive algorithm only).
- Algo selection:

	—	`l=3`	`w`	old `w`
default	basic	basic	basic	new
if failed	naive	split	new	
if failed		naive		

8-split: the interval is split into $2^3 = 8$ subintervals and the basic algorithm is tried again.

Tests on a 2 GHz AMD Opteron (at MEDICIS).

	$\exp x$, exponent 0				2^x, exponent 0			
c	–	l=3	w	old w	–	l=3	w	old w
0	42.30	35.46	35.26	(39.22)	37.83	32.95	32.82	(49.24)
1	26.32	19.27	19.09	(18.40)	23.83	18.72	18.67	(20.45)
3	24.09	16.82	16.85	(16.67)	22.21	16.96	17.04	(18.79)
5	24.47	17.29	17.29	(16.76)	23.23	18.03	18.08	(19.04)
–	21.54	14.23	14.26	(15.38)	21.68	16.42	16.52	(18.36)

	$\sin x$, exponent 0				$\cos x$, exponent 0			
c	–	l=3	w	old w	–	l=3	w	old w
0	40.24	31.72	31.67	(42.88)	39.08	33.52	33.51	(36.04)
1	28.28	19.52	19.49	(19.58)	25.87	20.10	20.18	(19.61)
3	26.41	17.54	17.55	(17.72)	22.76	16.93	17.08	(17.11)
5	27.15	18.36	18.32	(17.55)	23.15	17.29	17.47	(17.24)
–	23.71	14.74	14.85	(16.11)	19.99	14.12	14.30	(15.20)

	$\exp x$, exponent -6				2^x, exponent -6			
c	—	l=3	w	old w	—	l=3	w	old w
0	18.29	18.15	18.09	(59.08)	21.42	21.31	21.27	(81.95)
1	12.54	12.52	12.51	(18.05)	13.27	13.18	13.16	(22.15)
3	12.10	11.95	11.86	(17.07)	12.84	12.91	12.68	(21.26)
5	14.41	14.31	14.16	(17.65)	14.67	14.56	14.54	(22.34)
—	22.13	21.94	21.97	(26.25)	17.62	17.40	17.44	(21.31)

	$\sin x$, exponent -6				$\cos x$, exponent -6			
c	—	l=3	w	old w	—	l=3	w	old w
0	15.74	15.56	15.59	(16.21)	15.61	15.43	15.44	(19.10)
1	10.22	10.06	10.10	(9.79)	10.72	10.57	10.58	(10.74)
3	9.45	9.25	9.26	(9.33)	10.12	9.99	10.04	(10.58)
5	9.34	9.16	9.20	(9.30)	10.50	10.30	10.33	(10.72)
—	314.8	314.3	314.6	(369.9)	161.3	161.1	161.1	(188.6)

	$\exp x$, exponent 2			
`c`	—	`l=3`	`w`	old `w`
0	43.55	11.39	9.63	(11.00)
1	40.00	6.36	5.43	(5.28)
3	39.37	5.40	4.73	(4.61)
5	39.47	5.61	4.86	(4.71)
—	38.82	4.56	4.11	(4.26)

Note: the domain is 4 times as small as in the previous tables.

On the next slide: $\exp x$, with $x \approx \log(4)$, so that a is very close to a "simple" rational number…

	interval 50616				interval 50624			
c	—	l=3	w	old w	—	l=3	w	old w
0	1.79	1.12	1.15	(1.15)	1.67	1.06	1.03	(1.03)
1	1.44	0.81	0.78	(0.79)	1.37	0.77	0.77	(0.73)
3	1.40	0.77	0.78	(0.77)	1.35	0.72	0.72	(0.70)
5	1.39	0.76	0.76	(0.72)	1.35	0.73	0.70	(0.68)
—	20.63	20.70	20.15	(24.53)	40.42	40.54	40.19	(48.72)

	interval 50632				interval 50640			
c	—	l=3	w	old w	—	l=3	w	old w
0	1.15	0.59	7.72	(1653)	1.24	0.87	4.70	(708)
1	1.09	0.56	1.75	(279)	1.05	0.70	1.35	(120)
3	1.10	0.58	1.69	(259)	1.04	0.66	1.31	(111)
5	1.04	0.55	1.68	(259)	1.03	0.64	1.26	(111)
—	230	230	230	(323)	102	103	103	(137)

Comparison with SLZ (`wclr22`), 2^{40} Points

`test32f`: old algorithm, with divisions, and split into 8 subintervals (`-l=3`) in case of failure, i.e. if the lower bound d is too small.

program	interv.	# bits	rounding	lepuid	ay	marie
`test32f`	$[1/2,\ldots]$	64	D	23.8	81.8	11.5
`test32f`	$[1/2,\ldots]$	65	D & N	23.5	80.8	11.4
`test32f`	$[1,\ldots]$	64	D	26.6	86.8	13.2
`test32f`	$[1,\ldots]$	65	D & N	23.9	77.5	11.7
`wclr22`	$[-1/2,\ldots]$	64	N	26 – 28	111	12.4

D $\rightarrow$ for directed rounding modes; N $\rightarrow$ for rounding to nearest.

Machines: lepuid: Athlon; ay: PPC G4; marie: Opteron (MEDICIS).

Conclusion

- Improvements of my algorithm that computes a lower bound on the distance between a segment and $\mathbb{Z}^2$. The points with the smallest distance can be found (naive algorithm now useless).
- Can be used to find worst cases for correctly-rounded base conversion, possibly in a limited domain (see the paper).
- For math functions: limitations in the current implementation due to historical reasons. Most parts need a complete rewrite and new proofs (error bounds). But currently…
  $\rightarrow$ Worst cases for correctly-rounded double-precision functions: e^x, 2^x, 10^x, sinh, cosh, $\sin(2\pi x)$, $\cos(2\pi x)$, $1/x^2$, x^3; sin, cos, tan between $-\pi/2$ and $\pi/2$; the corresponding inverse functions.