Redundant CORDIC Timmermann

20161116

Copyright (c) 2015 Young W. Lim.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Low Latency Time CORDIC Algorithms - Timmermann (1992)
Redundant and on-line CORDIC - Ercegovac & Lang (1990)

(I) Constant Scale Factor & Redundant CORDIC

SD (Signed Digit) → Redundant Number System

Sign estimation of p MSD's (Most Significant Bits) of the residual angle z_i

⇒ determine $\sigma_i \in \{-1, +1\}$ angle direction

- would like to have $\sigma_i = 0$ as a valid Choice when p MSD's are all zero
- but can't be used because of scaling factors.

→ instead of rotating a vector just inc / dec the length of a vector to maintain the same scale factor

scale factor compensation

Ⅱ σ_i recording a priori is possible $\quad z_i \rightarrow \sigma_i$

→ parallel processing

$m=0$ bit conversion rule (binary → SD)

$m=\pm 1$ not simple.

- apply the simple conversion rule
- correct the prediction error by repeating the last iteration

convergence criterion $|z_k| < 2 \cdot \alpha_{m,k}$

Upper bound on prediction error

$$\boxed{k \leq 3j + 1.5}$$

$i \in [j, k]$

parallel

[$\sigma_{1..4}$	$j=1$	$1 \cdot 3 + 1 = 4$ → new j	$1 \sim 4$	repetition
[$\sigma_{4..13}$	$j=4$	$4 \cdot 3 + 1 = 13$ → new j	$4 \sim 13$	
[$\sigma_{13..40}$	$j=13$	$13 \cdot 3 + 1 = 40$ → new j	$13 \sim 40$	

$1 \cdots 4 \quad 4 \cdots 13 \quad 13 \cdots 40$

* termination algorithm

* CSD (Canonic Sign Digit)

$$\sigma_i \in \{-1, 0, +1\}$$

$$\alpha_{m,i} = 2\,\alpha_{m,i+1}$$

$$0111110 \Rightarrow 1000000\bar{1}0$$

* Timmermann 1992

Low Latency Time CORDIC Algorithms.

— from Swartzlander Hybrid CORDIC

rotations are applied sequentially
after parallel generation of
the corresponding group of rotation directions

parallel [$\sigma_{1..4}$] [$\sigma_{4..13}$] [$\sigma_{13..40}$]

rotation directions are generated in parallel
only within the group of associated CORDIC iterations

rotation directions of one group only can be
generated after the completion of the previous group

group

0 → 1
1 → 4 $\quad 3 \cdot 1 + 1 = 4$
4 → 13 $\quad 3 \cdot 4 + 1 = 13$
13 → 40 $\quad 3 \cdot 13 + 1 = 40$
40 → 121 $\quad 3 \cdot 40 + 1 = 121$

$j \sim 3j+1$

$2j+2$

The a priori knowledge of σ_i could prompt an attempt to parallelize the iteration and reduce the latency time

→ Baker's prediction scheme

$$\begin{cases} m = 0 & \text{linear} \\ m = 1 & \text{circular} \\ m = -1 & \text{hyperbolic} \end{cases}$$

rotation mode & $m = 0$ linear

$$y_n = y_0 + z_0 x_0$$

the initial value z_0

$$z_0 = \sum_i z_i 2^{-i} \qquad z_i \in \{0, 1\} \quad \text{ordinary binary}$$

$$\updownarrow$$

$$z_0 = \sum_i \sigma_i 2^{-i} \qquad \sigma_i \in \{-1, 1\} \quad \text{Signed Digit binary}$$

$m = 0$

$$
\boxed{
\begin{aligned}
x_{i+1} &= x_i - m\,\sigma_i \boxed{2^{-S(m,i)}}\, y_i \\
y_{i+1} &= y_i + \sigma_i \boxed{2^{-S(m,i)}}\, x_i \\
z_{i+1} &= z_i - \sigma_i \boxed{\alpha_{m,i}}
\end{aligned}
}
$$

m : coordinate system

$m=0$	linear
$m=+1$	circular
$m=-1$	hyperbolic

σ_i rotation **direction**

α_i rotation **angle**

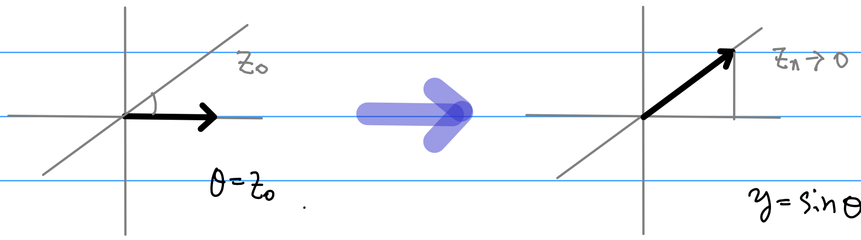
$S(m,i)$ the shift sequence $2^{-S(m,i)}$

$$\boxed{\alpha_{m,i}} = \frac{1}{\sqrt{m}} \tan^{-1}\left(\sqrt{m}\, 2^{-S(m,i)}\right)$$

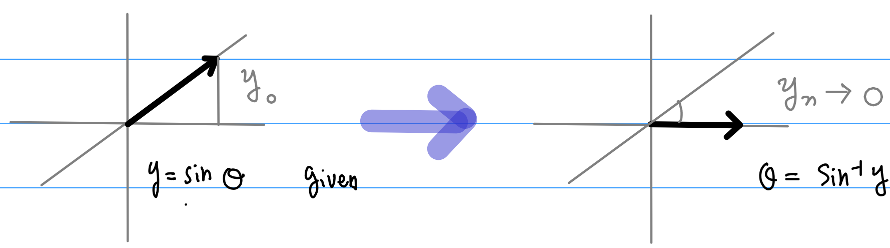
Low Latency Time CORDIC Algorithms - Timmerman (1992)

2 operational modes

① rotation $\boxed{z_n \to 0}$ residual angle $\to 0$



② Vectoring $\boxed{y_n \to 0}$ x-axis vector



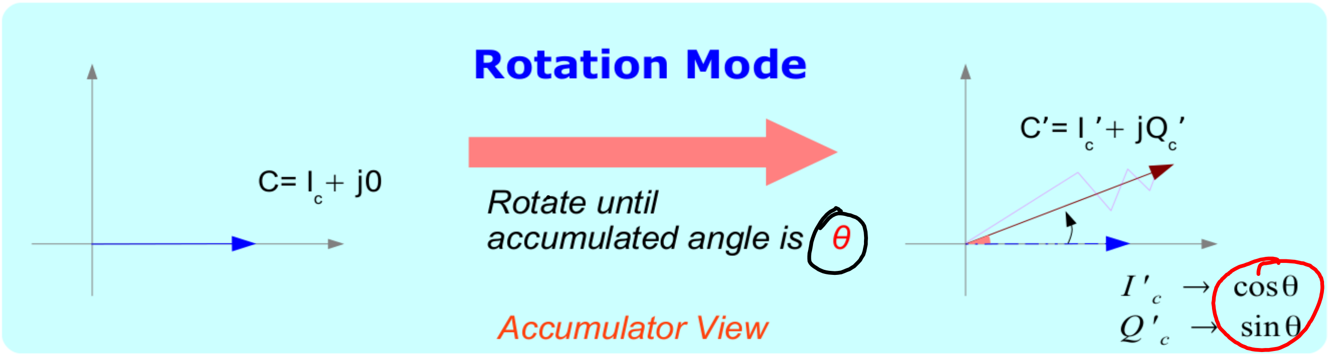
Rotation Direction

$$\sigma_i = \begin{cases} \text{sign}(z_i) & : \text{rotation } z_n \to 0 \\ -\text{sign}(x_i)\,\text{sign}(y_i) & : \text{vectoring } y_n \to 0 \end{cases}$$

Scaling Factor

$$k_m = \prod_i \sqrt{1 + m\,\sigma_i^2\, 2^{-2S(m,i)}}$$

① rotation $\boxed{z_n \to 0}$ residual angle $\to 0$



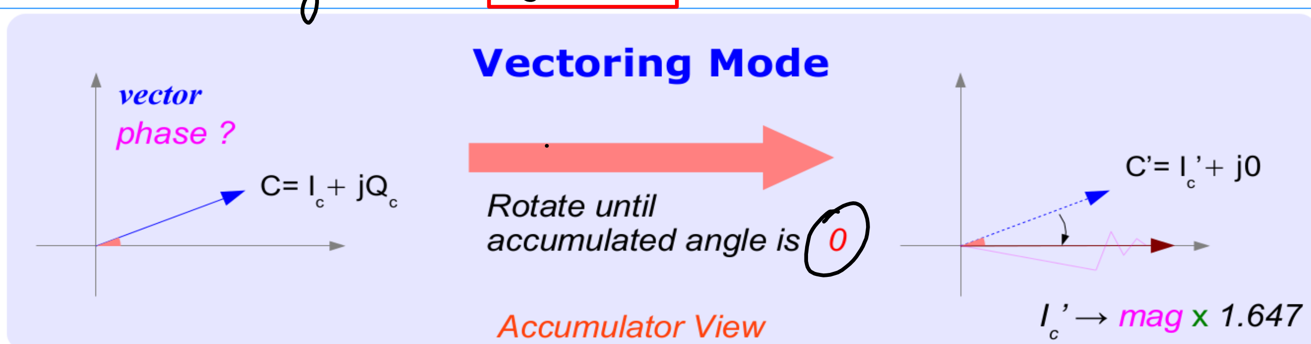
$$x_n = k_m \cdot \left(x_0 \cos(\sqrt{m}\, z_0) - \sqrt{m}\, y_0 \sin(\sqrt{m}\, z_0) \right)$$

$$y_n = k_m \cdot \left(y_0 \cos(\sqrt{m}\, z_0) + \frac{1}{\sqrt{m}} x_0 \sin(\sqrt{m}\, z_0) \right)$$

$$z_n \to 0$$

$z_0 \cdots\cdots z_n \to 0$

② Vectoring $\boxed{y_n \to 0}$ x-axis vector



$$x_n = k_m \sqrt{x_0^2 + m\, y_0^2}$$

$$y_n \to 0$$

$$z_n = z_0 + \frac{1}{\sqrt{m}} \cdot \tan^{-1}\left(\sqrt{m}\, y_0 / x_0\right)$$

① rotation $\boxed{z_n \to 0}$ residual angle $\to 0$

② Vectoring $\boxed{y_n \to 0}$ x-axis vector

Rotation Mode

$$\boxed{z_0 \leftarrow \phi \quad (desired\ angle)}$$

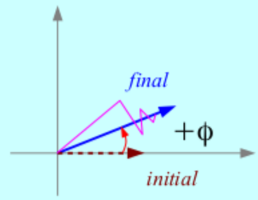
$$z_n \to 0$$

$$\boxed{z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})}$$

$$d_i = -1 \quad if \quad z_i < 0$$

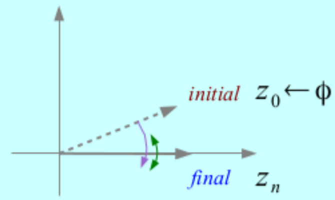
$$d_i = +1 \quad otherwise$$

Vector View



Minimize the residual angle

Accumulator View



Subtract *angles at each step*

$$\boxed{\begin{aligned} x_{i+1} &= x_i - y_i \cdot d_i \cdot 2^{-i} \\ y_{i+1} &= y_i + x_i \cdot d_i \cdot 2^{-i} \\ z_{i+1} &= z_i - d_i \cdot \tan^{-1}(2^{-i}) \end{aligned}}$$

$$\boxed{\begin{aligned} d_i &= -1 \quad if \quad z_i < 0 \\ d_i &= +1 \quad otherwise \end{aligned}}$$

$$x_n = A_n \left[x_0 \cos z_0 - y_0 \sin z_0 \right]$$

$$y_n = A_n \left[y_0 \cos z_0 + x_0 \sin z_0 \right]$$

$$\boxed{z_n = 0}$$

$$A_n = \prod_{i=1}^{n} \sqrt{1 + 2^{-2i}}$$

Vectoring Mode

$$z_0 \leftarrow 0$$

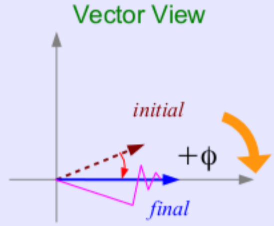
$$\boxed{z_n \to z_0 + \tan^{-1}(y_0 / x_0)}$$

$$\boxed{z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})}$$

$$d_i = +1 \quad if \quad y_i < 0$$

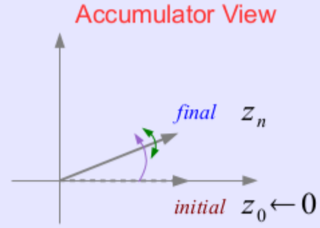
$$d_i = -1 \quad otherwise$$

Vector View



Minimize the residual y component

Accumulator View



Add *angles at each step*

$$\boxed{\begin{aligned} x_{i+1} &= x_i - y_i \cdot d_i \cdot 2^{-i} \\ y_{i+1} &= y_i + x_i \cdot d_i \cdot 2^{-i} \\ z_{i+1} &= z_i - d_i \cdot \tan^{-1}(2^{-i}) \end{aligned}}$$

$$\boxed{\begin{aligned} d_i &= +1 \quad if \quad y_i < 0 \\ d_i &= -1 \quad otherwise \end{aligned}}$$

$$x_n = A_n \sqrt{x_0^2 + y_0^2}$$

$$\boxed{y_n = 0}$$

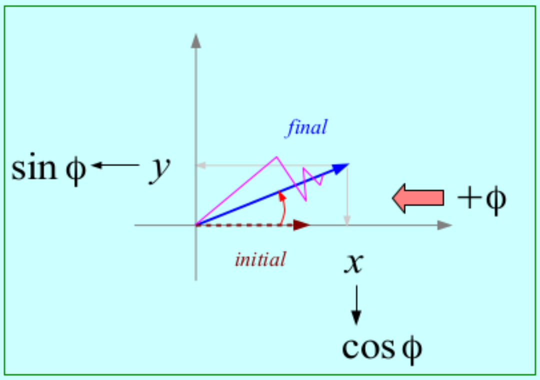
$$z_n = z_0 + \tan^{-1}(y_0 / x_0)$$

$$A_n = \prod_{i=1}^{n} \sqrt{1 + 2^{-2i}}$$

Rotation Mode

Input angle is given

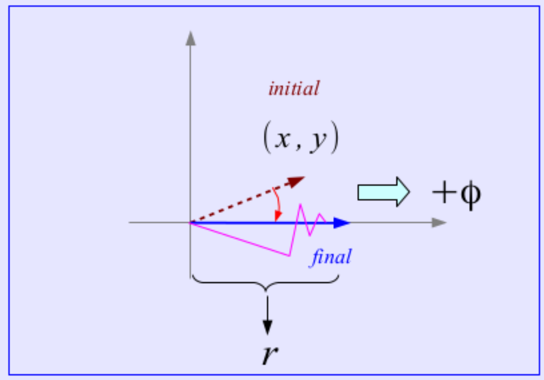
- **sin** and **cos**
- $(r, \theta) \rightarrow (x, y)$
- General vector rotation



Vectoring Mode

Finding the resulting angle

- **tan^{-1}**
- Vector Magnitude
- $(x, y) \rightarrow (r, \theta)$



① rotation $\boxed{z_n \to 0}$ residual angle $\to 0$

② Vectoring $\boxed{y_n \to 0}$ x-axis vector

Represent arbitrary angle θ

in terms of $\pm\theta_0, \ \pm\theta_1, \ \pm\theta_2, \ \pm\theta_3, \ \cdots, \ \pm\theta_l, \ \cdots$ $\left(k_l = \tan\theta_l = \dfrac{1}{2^l}, \quad l = 0, 1, 2, \cdots \right)$

Binary Search ➡ Shift and Add ➡ No multiplier

Phase of R

$\theta_0 = \tan^{-1}(2^0) =$ 45º

$\theta_1 = \tan^{-1}(2^{-1}) =$ 26.56505º

$\theta_2 = \tan^{-1}(2^{-2}) =$ 14.03624º

$\theta_3 = \tan^{-1}(2^{-3}) =$ 7.12502º

$\theta_4 = \tan^{-1}(2^{-4}) =$ 3.57633º

$\theta_5 = \tan^{-1}(2^{-5}) =$ 1.78991º

$\theta_6 = \tan^{-1}(2^{-6}) =$ 0.89517º

$\theta_7 = \tan^{-1}(2^{-7}) =$ 0.44761º

•••

$\approx \frac{1}{2}$ $\approx \frac{1}{2}$ $\approx \frac{1}{2}$ $\approx \frac{1}{2}$ $\approx \frac{1}{2}$ $\approx \frac{1}{2}$ $\approx \frac{1}{2}$

> 92º

$\theta \in$ [-180, +180]

$R = 0 \pm j$

$\theta \in$ [-90, +90]

$R = 1 \pm j\,k_l$

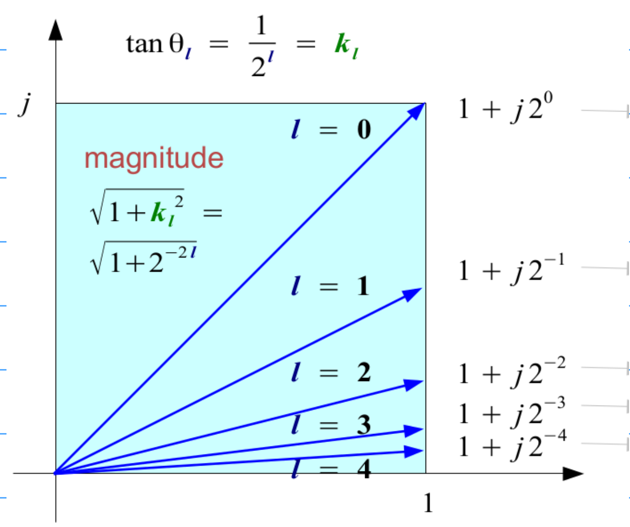
4 or more rotations

45º + 26.56505º + 14.03624º + 7.12502º
= 92.726º > 92º

$R = 1 \pm j k_l$	Magnitude of R	$\sqrt{1^2 + k_l^2}$ > 1.0	Cumulative Magnitude CORDIC Gain
$R_0 = 1 \pm j\,(1/2^0)$	$\sqrt{1^2 + 1^2} =$	1.41421356	1.414213562
$R_1 = 1 \pm j\,(1/2^1)$	$\sqrt{1^2 + (1/2)^2} =$	1.11803399	1.581138830
$R_2 = 1 \pm j\,(1/2^2)$	$\sqrt{1^2 + (1/2^2)^2} =$	1.03077641	1.629800601
$R_3 = 1 \pm j\,(1/2^3)$	$\sqrt{1^2 + (1/2^3)^2} =$	1.00778222	1.642484066
$R_4 = 1 \pm j\,(1/2^4)$	$\sqrt{1^2 + (1/2^4)^2} =$	1.00195122	1.645688916
$R_5 = 1 \pm j\,(1/2^5)$	$\sqrt{1^2 + (1/2^5)^2} =$	1.00048816	1.646492279
$R_6 = 1 \pm j\,(1/2^6)$	$\sqrt{1^2 + (1/2^6)^2} =$	1.00012206	1.646693254
$R_7 = 1 \pm j\,(1/2^7)$	$\sqrt{1^2 + (1/2^7)^2} =$	1.00003052	1.646743507
		...	...
			1.647

Iteration

The actual CORDIC Gain depends on the number of iterations



$$\alpha_0 \rightarrow \cdots \rightarrow \alpha_i \rightarrow \alpha_{i+1} \rightarrow \cdots \rightarrow \alpha_n \to 0$$

$$\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \rightarrow \cdots \rightarrow \begin{pmatrix} x_i \\ y_i \end{pmatrix} \rightarrow \begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} \rightarrow \cdots \rightarrow \begin{pmatrix} x_n \\ y_n \end{pmatrix}$$

$$x_{i+1} = x_i \cos\theta_i - y_i \sin\theta_i = \cos\theta_i\,(x_i - y_i \tan\theta_i) = \left(1/\sqrt{1+\tan^2\theta_i}\right)(x_i - y_i \tan\theta_i)$$

$$y_{i+1} = x_i \sin\theta_i + y_i \cos\theta_i = \cos\theta_i\,(x_i \tan\theta_i + y_i) = \left(1/\sqrt{1+\tan^2\theta_i}\right)(x_i \tan\theta_i + y_i)$$

Choose θ_i such that $\tan\theta_i = \begin{cases} +2^{-i} \\ -2^{-i} \end{cases}$

$$\tan\theta_i = \sigma_i 2^{-i} \qquad \sigma_i \in \{+1, -1\}$$

$$\begin{aligned} x'_{i+1} &= (x'_i - y'_i\,\sigma_i 2^{-i}) \\ y'_{i+1} &= (x'_i\,\sigma_i 2^{-i} + y'_i) \\ \theta_{i+1} &= \theta_i - \tan^{-1}(\sigma_i 2^{-i}) \end{aligned}$$

$$\begin{pmatrix} +\cos\theta & -\sin\theta \\ +\sin\theta & +\cos\theta \end{pmatrix}$$

$$= \frac{1}{\sqrt{1+2^{-2\cdot n}}}\begin{pmatrix} +1 & \mp 2^{-n} \\ \pm 2^{-n} & +1 \end{pmatrix} \cdots \frac{1}{\sqrt{1+2^{-2\cdot 1}}}\begin{pmatrix} +1 & \mp 2^{-1} \\ \pm 2^{-1} & +1 \end{pmatrix} \cdot \frac{1}{\sqrt{1+2^{0}}}\begin{pmatrix} +1 & \mp 2^{0} \\ \pm 2^{0} & +1 \end{pmatrix}$$

$$= \underline{\frac{1}{\sqrt{1+2^{-2\cdot n}}} \cdots \frac{1}{\sqrt{1+2^{-2\cdot 1}}} \cdot \frac{1}{\sqrt{1+2^{-2\cdot 0}}}}\ \underline{\begin{pmatrix} +1 & \mp 2^{-n} \\ \pm 2^{-n} & +1 \end{pmatrix} \cdots \begin{pmatrix} +1 & \mp 2^{-1} \\ \pm 2^{-1} & +1 \end{pmatrix}\begin{pmatrix} +1 & \mp 2^{-0} \\ \pm 2^{-0} & +1 \end{pmatrix}}$$

$\Rightarrow$ $K = \prod 1/\sqrt{1+\tan^2\theta_i} = 0.607$ $\Rightarrow$ $\begin{pmatrix} +\cos(\Sigma\theta_i) & -\sin(\Sigma\theta_i) \\ +\sin(\Sigma\theta_i) & +\cos(\Sigma\theta_i) \end{pmatrix}$

$$\begin{pmatrix} +\cos(\Sigma\theta_i) & -\sin(\Sigma\theta_i) \\ +\sin(\Sigma\theta_i) & +\cos(\Sigma\theta_i) \end{pmatrix} = \frac{1}{K} \cdot \begin{pmatrix} +\cos\theta & -\sin\theta \\ +\sin\theta & +\cos\theta \end{pmatrix}$$

$$1/K = \prod \sqrt{1+\tan^2\theta_i} = 1.647 = A = \textit{CORDIC Gain}$$

magnitude

$$\sqrt{1+k_l^{\,2}} = \sqrt{1+2^{-2l}}$$

digit-by-digit algorithm

linear convergence

Sequential behavior

n-bit precision ~ approximately n iteration

$(i+1)$-th iteration only after (i)-th iteration

Critical Path → Add/Sub Operations

CPA (Carry Propagating Adder) $\propto n$

SDA (Signed Digit Adder) $\propto 2\tau$

CSA (Carry Save Adder) τ : delay of a FA

Redundant Number

σ_i has to be estimated from some of MSD's to determine the sign of a redundant number

$$\sigma_i = \begin{cases} \mathrm{sign}(z_i) & : \text{rotation } z_n \to 0 \\ -\mathrm{sign}(x_i)\,\mathrm{sign}(y_i) & : \text{vectoring } y_n \to 0 \end{cases}$$

n-bit addition : critical

ⓐ **redundant binary**

radix-2 SD (Signed Digit) adder

N Takagi, 1987

ⓑ **Carry Save Adder**

Ercegovac, Lang 1990

Redundant and on-line CORDIC

redundant numbers in CORDIC arithmetic

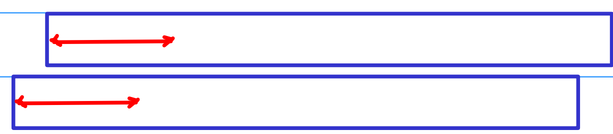
σ_i has to be estimated from the inspection of some of the most significant digits MSD's

if all the inspected digits are all zero, the proper value of σ_i cannot be determined without knowledge of the remaining digits

the best strategy $\sigma_i \leftarrow 0$ freezing iteration
but this affects k_m (data dependent)

may increase latency and chip area by at least 80 %
still requires n sequential decisions to generate all σ_i
→ preventing a parallelization

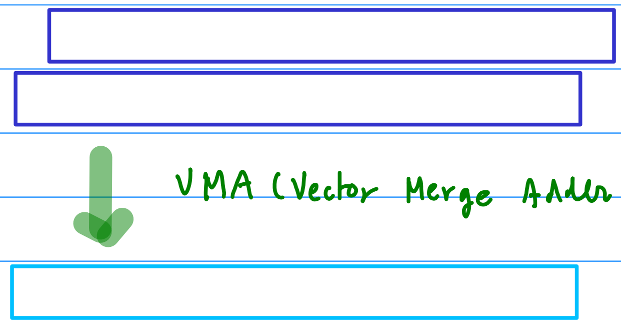
inspect some MSD's



intermediate sum
intermediate carry

Most Significant Digit

Low Latency Time CORDIC Algorithms - Timmerman (1992)



intermediate sum

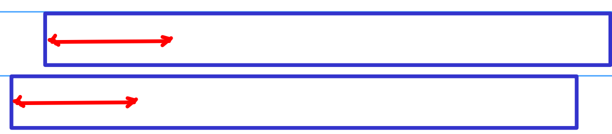
intermediate carry

VMA (Vector Merge Adder, ie CLA ...)

expensive

it would be straight forward to determine the sign of z_i once z_i is converted into an ordinary binary number.

inspect only a few MSD's



perform VMA over a few MSD's only

$$\begin{cases} + : & z_i > 0 \\ - : & z_i < 0 \\ 0 : & \text{the exact sign requires the remaining bits} \end{cases}$$

i : iteration

n : word length

$$k_m = \prod_i \sqrt{1 + m\,\sigma_i^2\, 2^{-2\boxed{s(m,i)}}}$$

assume $\boxed{s(m,i) \Rightarrow i}$

$\boxed{\prod_i \sqrt{1+2^{-2i}}}$

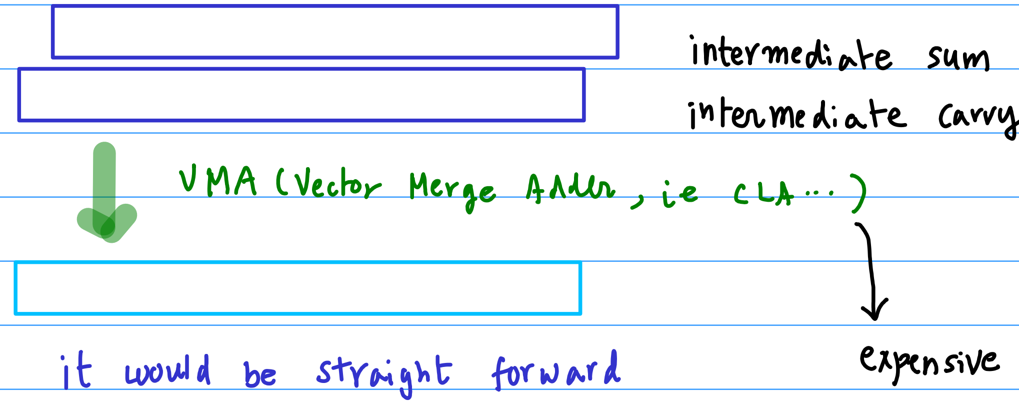
$$k_m = \prod_i \sqrt{1 + m \cdot 2^{-2i}}$$

assume $\boxed{\frac{1}{4}(n-3) < i}$

$\boxed{\prod_i (1+2^{-2i-1})}$

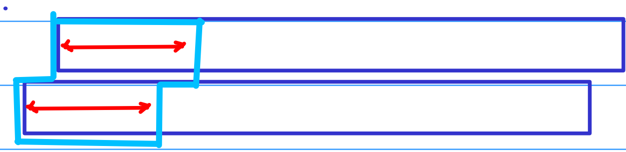
$$k_m = \prod_i \left(1 + m \cdot 2^{-2i-1}\right)$$

Vector Merging Adder (VMA)



it would be straight forward to determine the sign of z_i once z_i is converted into an ordinary binary number.

inspect only a few MSD's



perform VMA over a few MSD's only

$$\begin{cases} + : & z_i > 0 \\ - : & z_i < 0 \\ 0 : & \text{the exact sign requires the remaining bits} \end{cases}$$

SD Adder

Takagi's Method

Takagi { a halving of each iteration
executing twice

2 smaller rotations

i-iteration for $m=1$ and $S(m,i)=i$

for $\sigma_i = \pm 1$ $\quad \pm\sigma_i \tan^{-1}(2^{-i-1})$ and $\pm\sigma_i \tan^{-1}(2^{-i-1})$

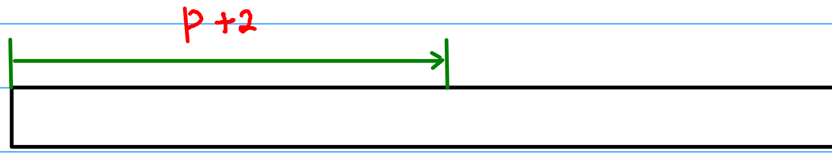
for $\sigma_i = 0$ $\quad \pm\sigma_i \tan^{-1}(2^{-i-1})$ and $\mp\sigma_i \tan^{-1}(2^{-i-1})$

2 successive rotation per iteration

Ⓐ Takagi's MSD Inspection

inspecting at most $p+2$ MSD's
of a redundant binary representation

p-th iteration



a **repetition** of each p-th iteration
suffices to ensure convergence

$1, 2, 3, \cdots, p, p$: iteration

θ_0	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7

θ_0	θ_1	θ_2	θ_3	$\pm\theta_3$	θ_4	θ_5	θ_6	θ_7	$\pm\theta_7$

$p=4$

the latency of inspection $\propto p$

* usually implemented using fast carry dependant adder
* $t_{inspection} < t_{adding}$

→ limits $p < 4$ or 5

(Noll's MSD Inspection)
if carry-save adder is used
and 4 MSD's are inspected
doubling of every second iteration is necessary

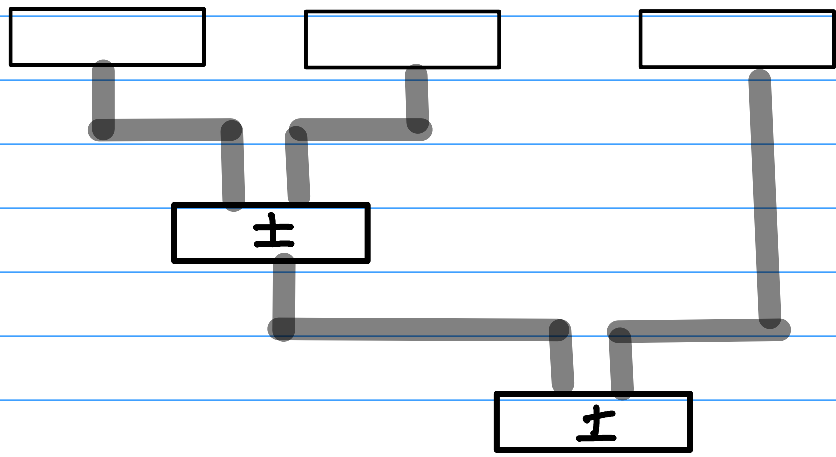
2 successive rotations per iteration

→ merge in one redundant 3-input add/sub operation

for $\sigma_i = +1$ $\quad + \tan^{-1}(2^{-i-1}) + \tan^{-1}(2^{-i-1})$

for $\sigma_i = -1$ $\quad - \tan^{-1}(2^{-i-1}) - \tan^{-1}(2^{-i-1})$

for $\sigma_i = 0$ $\quad + \tan^{-1}(2^{-i-1}) - \tan^{-1}(2^{-i-1})$



→ require 2 4-to-2 cells forming a 6-to-2 cell

4-to-2 cell a redundant adder with 2 redundant inputs (2 bit each)

3-to-2 cell a full adder (carry save adder)

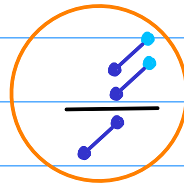
4-to-2 & 3-to-2 Cells

Takagi

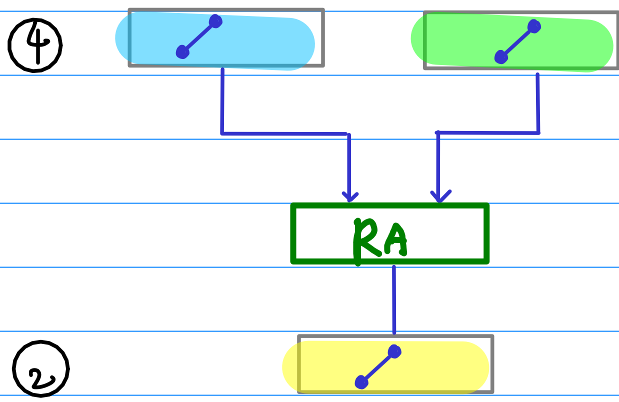
SD Adder

4-to-2 cell

2 inputs



each input having 2 bits
intermediate carry/sum

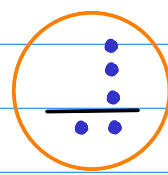


redundant adder

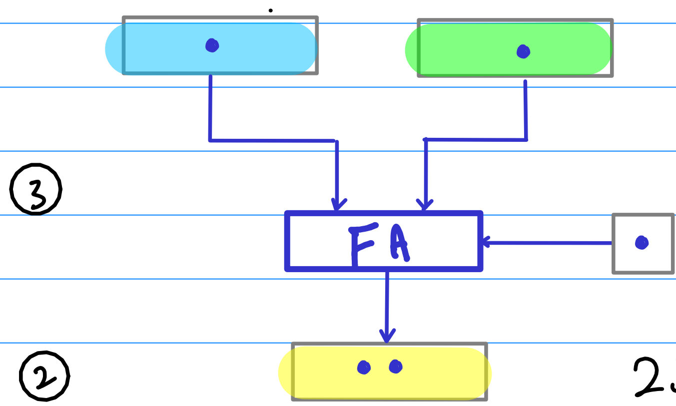
2bit output
intermediate carry / sum

3-to-2 cell

2 inputs



each input having 1 bit

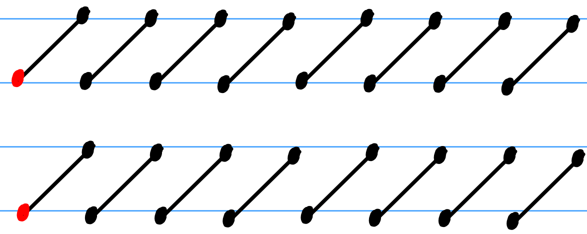


1 bit carry in

2bit output
carry / sum

Takagi

SD Adder



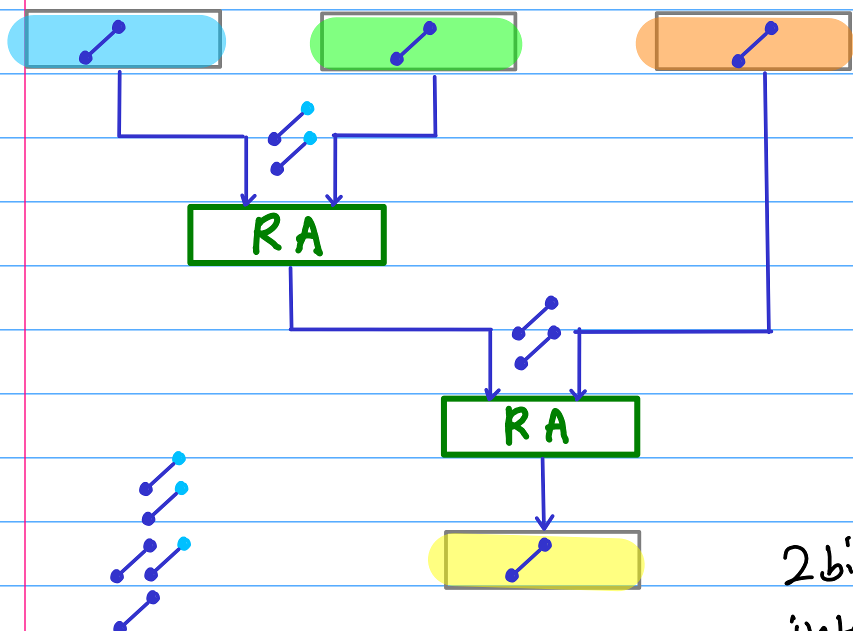
6-to-2 cell

Takagi

SD Adder

3 inputs

each input having 2 bits
intermediate carry/sum



redundant adder

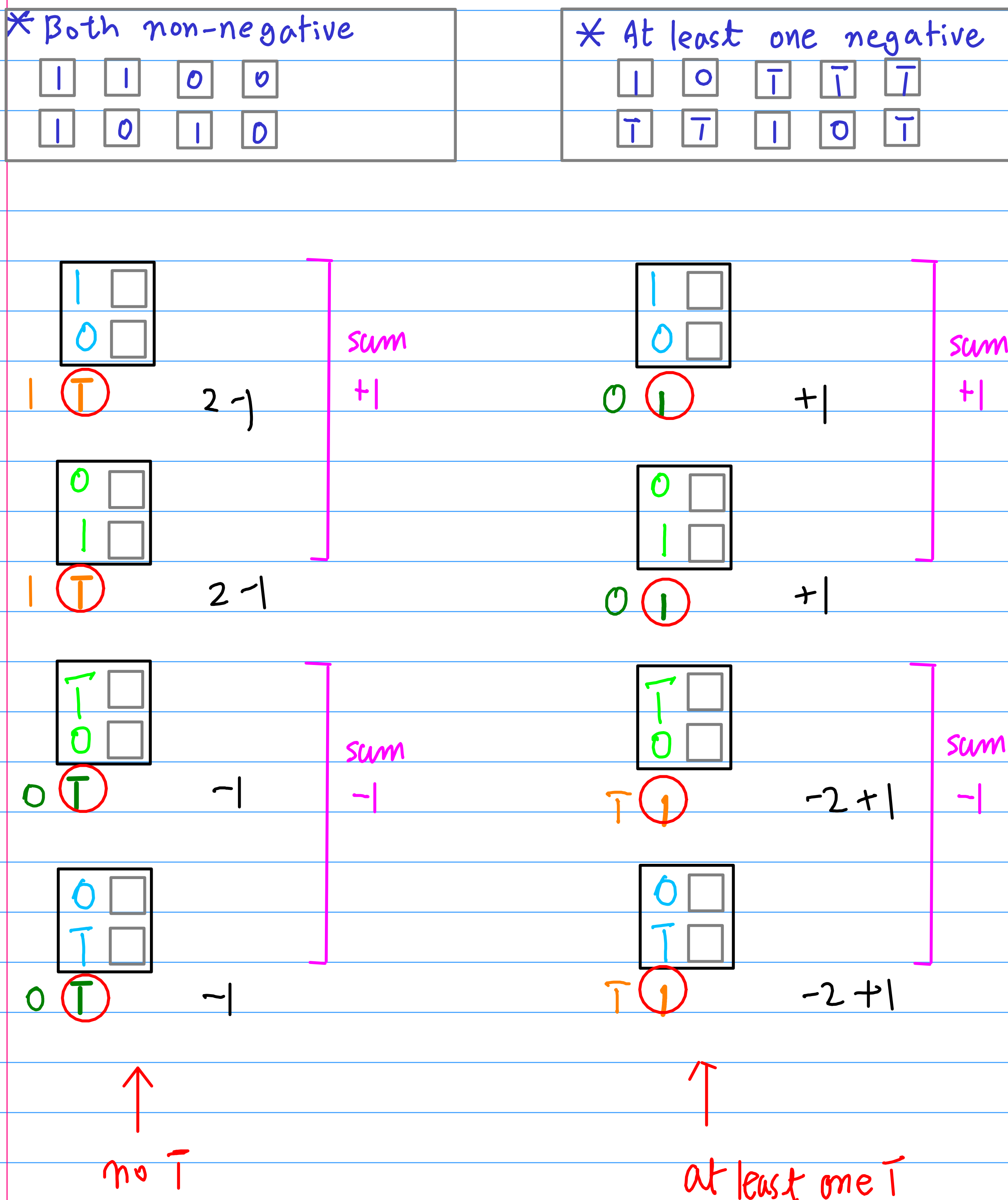
redundant adder

2bit output
intermediate carry / sum

Low Latency Time CORDIC Algorithms - Timmerman (1992)

Takagi

SD Adder



Takagi

SD Adder

p_i	q_i	c_i	s_i
1	1	1	0
1	0	1	$\bar{1}$
0	1	0	1
0	0	0	0
1	$\bar{1}$	0	0
$\bar{1}$	1	0	0
$\bar{1}$	0	0	$\bar{1}$
0	$\bar{1}$	$\bar{1}$	1
$\bar{1}$	$\bar{1}$	$\bar{1}$	0

augend	$0.010\bar{1}$ [SD2]	step 1
addend	$0.10\bar{1}0$ [SD2]	
intermediate sum →	$0.\bar{1}11\bar{1}$	step 2
intermediate carry →	$01.0\bar{1}0$	
Sum	$01.\bar{1}01\bar{1}$	

$$\begin{array}{cccccc} 0. & 0 & 1 & 0 & \bar{1} \\ 0. & 1 & 0 & \bar{1} & 0 \\ & & & 1 & \\ & & \bar{1} & & \end{array}
\qquad
\begin{array}{cccccc} 0. & 0 & 1 & 0 & \bar{1} \\ 0. & 1 & 0 & \bar{1} & 0 \\ & & 1 & & \\ & 0 & & & \end{array}
\qquad
\begin{array}{cccccc} 0. & 0 & 1 & 0 & \bar{1} \\ 0. & 1 & 0 & \bar{1} & 0 \\ & \bar{1} & & & \\ 1 & & & & \end{array}$$

(B) Nolls MSD Inspection

(1) Noll
(2) Künemund

} carry save + MSD

$$\boxed{\begin{aligned} X_j &= X_{j-1} - q_j \cdot 2^{-j} \cdot Y_{j-1} \\ Y_j &= Y_{j-1} + q_j \cdot 2^{-j} \cdot X_{j-1} \\ Z_j &= Z_{j-1} - q_j \cdot \tan^{-1} 2^{-j} \end{aligned}}$$

in each step, q_j is selected to decrease the remaining angle Z_j

$$q_j = \text{sgn}(Z_{j-1})$$

If the magnitude of the remaining angle is too small to allow exact detection of the sign only from inspection of a few MSD's

$\boxed{\frac{1}{4}(n-3) < i}$ near zero

00 ⋯ 0 □□ ⋯ □

Noll

CSA

One solution

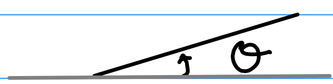
to **double** all the elements in the angle sequence
and therefore its total length

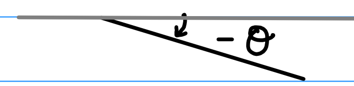
Now each iterative rotation
can be completely compensated
to a net angle of zero in the step

→ almost equivalent to allowing a single rotation by zero

j $\quad\quad \theta_j = 2 \cdot \theta_{j+1} \quad\quad$ j+1

$+1 \quad\quad -1$





$$\begin{cases} \theta_j - (2\,\theta_{j+1}) = 0 \\ -\theta_j + (2\,\theta_{j+1}) = 0 \end{cases}$$

θ_0	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7

θ_0	$-\theta_0$	θ_1	$-\theta_1$	θ_2	$-\theta_2$	θ_3	$-\theta_3$	θ_4	$-\theta_4$	θ_5	$-\theta_5$	θ_6	$-\theta_6$	θ_7	$-\theta_7$

Carry-save arithmetic for high speed digital signal processing, Noll,1990

doubling of each element of the sequence is not necessary

(# of the inspected digits
# of angle elements which have to be doubled

$p = 3$ or 4 MSD's

→ typically only each second sequence element has to be doubled

θ_0	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7

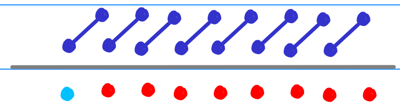
θ_0	θ_1	$-\theta_1$	θ_2	θ_3	$-\theta_3$	θ_4	θ_5	$-\theta_5$	θ_6	θ_7	$-\theta_7$

Carry-save arithmetic for high speed digital signal processing, Noll,1990

Noll

CSA

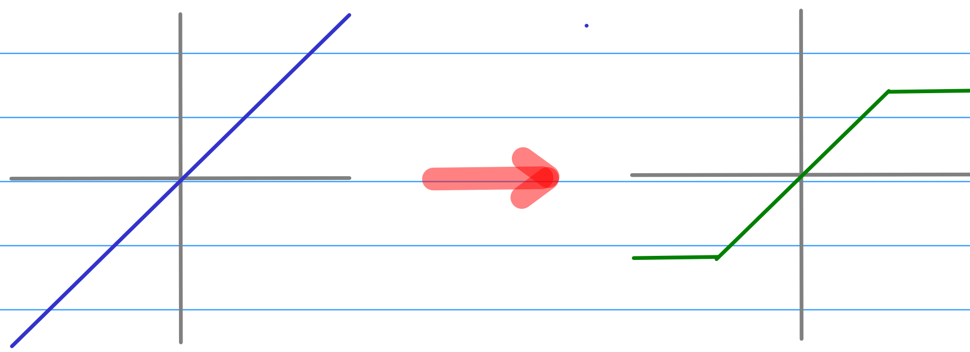
VMA (Vector Merging Adder)



final stage adder

CP (Carry Propagation) Adder

Level slicing problem



In the redundant number system

ⓐ exact comparison

VMA may be used

ⓑ magnitude estimation

by inspecting only a few MSD

(Most Significant Digits)

Carry-save arithmetic for high speed digital signal processing, Noll,1990