

Stochastic rounding: implementation, error analysis and applications

An update to the 2022 survey paper

Mantas Mikaitis

School of Computing, University of Leeds, Leeds, UK

European Conference on Numerical Mathematics and Advanced
Applications

Heidelberg, Germany, Sep. 1, 2025



MS17: Latest Advancements in Floating-Point Arithmetic with Probabilistic Rounding

Presentations:

16:30-16:55 *Stochastic rounding: implementation, error analysis and applications (an update to the 2022 survey paper).*

Mantas Mikaitis (Univ. Leeds, UK)

16:55-17:20 *Probabilistic error analysis of limited-precision stochastic rounding.*

Silviu-Ioan Filip (INRIA Rennes, France)

17:20-17:45 *On the practical implementation of few-bit stochastic rounding, with applications to large language models.*

Andrew Fitzgibbon (Graphcore, UK)

ROYAL SOCIETY
OPEN SCIENCE

royalsocietypublishing.org/journal/rsos

Review



Cite this article: Croci M, Fasi M, Higham NJ, Mary T, Mikaitis M. 2022 Stochastic rounding: implementation, error analysis and applications. *R. Soc. Open Sci.* **9**: 211631.
<https://doi.org/10.1098/rsos.211631>

Received: 14 October 2021
Accepted: 4 February 2022

Subject Category:
Computer science and artificial intelligence

Subject Areas:
applied mathematics/software/computer
modelling and simulation

Stochastic rounding: implementation, error analysis and applications

Matteo Croci¹, Massimiliano Fasi², Nicholas J. Higham³,
Theo Mary⁴ and Mantas Mikaitis³

¹Oden Institute, University of Texas at Austin, Austin, TX, 78712, USA

²Department of Computer Science, Durham University, Durham, DH1 3LE, UK

³Department of Mathematics, The University of Manchester, Manchester, M13 9PL, UK

⁴Sorbonne Université, CNRS, LIP6, Paris, 75005, France

MC, 0000-0003-1669-9445; MF, 0000-0002-6015-391X;
NJH, 0000-0001-5956-4976; TM, 0000-0001-9949-4634;
MM, 0000-0001-8706-1436

Stochastic rounding (SR) randomly maps a real number x to one of the two nearest values in a finite precision number system. The probability of choosing either of these two numbers is 1 minus their relative distance to x . This rounding mode was first proposed for use in computer arithmetic in the 1950s and it is currently experiencing a resurgence of interest. If used to compute the inner product of two vectors of length n in floating-point arithmetic, it yields an error bound with constant $\sqrt{n}\mu$ with high probability, where μ is the unit round-off. This is not necessarily the case for round to nearest (RN), for which the worst-case error bound has constant $n\mu$. A particular attraction of SR is that, unlike RN, it is immune to the phenomenon of saturation, whereby a sequence of times

Hardware aspects in the survey

102 references. Mostly reviewed patents; commercial hardware not clear at that point.

Introduction

- In binary floating-point hardware **round-to-nearest** (RN) is a default mode (standardized by IEEE 754).
- Deterministic, optimal accuracy per operation.
- Closest machine number to real answer—cannot improve.
- Over many rounding ops may accumulate error of factor n , where n a problem size.

What we get from today's talk

Learn about the latest hardware including **stochastic rounding** (SR) that has \sqrt{n} error growth.

Floating-point (FP) number representation

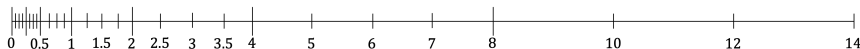
A floating-point system $F \subset \mathbb{R}$ is described with t, e_{min}, e_{max} with elements

$$x = \pm m \times 2^{e-t+1}.$$

Here t is precision, $e_{min} \leq e \leq e_{max}$ an exponent, $m \leq \beta^t - 1$ a significand ($m, t, e \in \mathbb{Z}$).

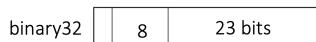
3-bit prec. example FP

Below: the positive numbers in $\mathbb{F}(t = 3, e_{min} = -2, e_{max} = 3)$.



Common floating-point formats

Main IEEE 754 formats (**double**, **single**, **half**):

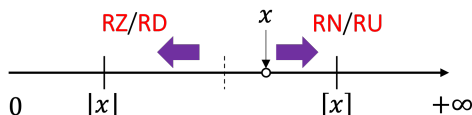


Some ubiquitous non-standard formats:



IEEE 754 standard FP arithmetic: rounding

- Round-to-nearest (**RN**) (ties even)
- Round-toward-zero (**RZ**)
- Round-down (**RD**)
- Round-up (**RU**)



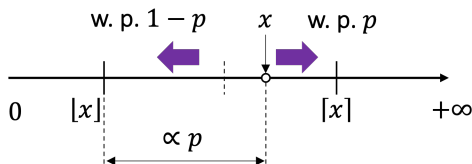
Use of rounding modes

RN is a default. *Directed modes* used for special cases, such as **interval arithmetic**.

What is stochastic rounding

With **stochastic rounding (SR)**, we are not rounding a number to the same direction, but to either direction with probability.

Given some x and FP neighbours $\lfloor x \rfloor$, $\lceil x \rceil$, we round to $\lceil x \rceil$ with prob. p and $\lfloor x \rfloor$ with $p - 1$.



Mode 1 SR (nearness): $p = \frac{x - \lfloor x \rfloor}{\lceil x \rceil - \lfloor x \rfloor}$

Mode 2 SR: $p = 0.5$

Mode 2

With **Mode 1 SR** we round x depending on its distances to the nearest two FP numbers, **cancelling out errors of different signs**.

Stagnation in floating-point sums

Take binary floating-point numbers a and b , such that $a \gg b$ and $\text{fl}(a + b) = a$ (round-to-nearest).

In sums of arbitrary length, $s_n = x_1 + x_2 + \cdots + x_n$, *stagnation* appears if, for example, $\text{fl}(x_1 + x_i) = x_1$ for all $i \leq n$ and therefore $\hat{s}_n = x_1$.

If $x_i > 0$, the total error is $x_2 + \cdots + x_n$, which is a growth of factor $2^{-t}(n - 1)$.

Stagnation/swamping

Whole, part, or parts of a running sum do not change the intermediate value, and addends contribute fully to the error.

Stagnation in floating-point sums

With SR, stagnation *is not as severe* as with RN.

Take again a and b , such that $a \gg b$ and with RN $\text{fl}(a + b) = a$.

With SR $\text{fl}(a + b)$ will yield a or the next floating-point value with probability $\frac{b}{\text{ulp}(a)}$ where $\text{ulp}(a)$ is the gap between a and next fl. val.

$2^{-t} \sqrt{n}$ error growth.

Stagnation with SR

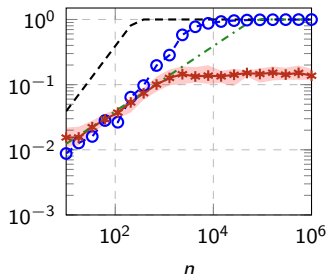
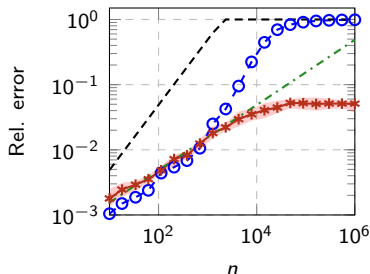
Stagnation can still occur if b is so small that its significand gets shifted past the random bits in SR; error growth drops back to factor nu .

Example error growth with SR in mat-vec prod

Backward error in $y = Ax$ where $A \in \mathbb{R}^{100 \times n}$ with entries from uniform dist over $[0, 10^{-3}]$ and $x \in \mathbb{R}^n$ over $[0, 1]$: $\max_i \frac{|\hat{y} - y|_i}{(|A||x|)_i}$.

(a) binary16 arithmetic

(b) bfloat16 arithmetic



—○— RN

—*— SR

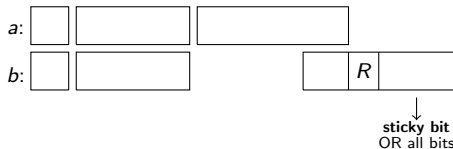
SR range

--- $\min(nu, 1)$

-.- $\min(\sqrt{n}u, 1)$

How do we implement this? First, consider standard modes

Consider $a, b \in \mathbb{F}$ with $a, b > 0$ and $a > b$.



round-sticky	RD	RU	RN
00	D	D	D
01	D	U	D
10	D	U	D/U
11	D	U	U

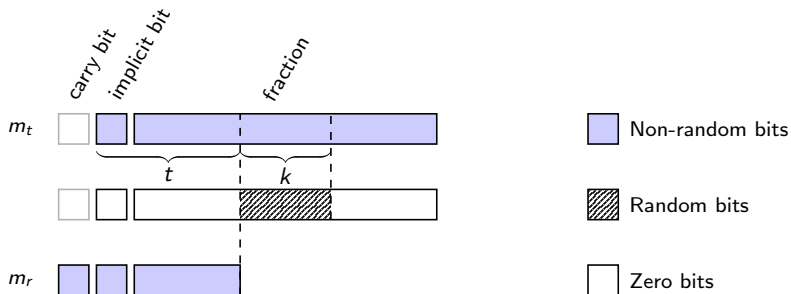
Guard bit

Guard bit is a complication that arises when we consider non-normalized floating-point significands, to compute the R bit correctly.

Implementation of SR

Take m_t to be a high precision unrounded significand from an operation.

Take t to be source precision and k the precision of random numbers.

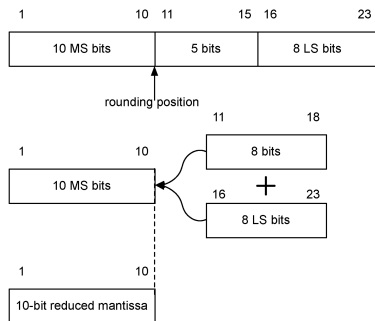


SR in NVIDIA's Instruction Sets

There are numerous patents for SR from industry: NVIDIA, AMD, IBM. See our SR survey [\[Crocì et al, 2022\]](#).

Here we focus on NVIDIA's ([\[NVIDIA, 2019\]](#)).

Below binary32 \rightarrow binary16 example.



- Does not use PRNG.
- Take 8 bottom discarded bits and add to the top 8.
- Deterministic and cheaper to implement.
- Effect on numerical results not known.

Graphcore Tile Vertex ISA 1.3.1

Following arith. instructions support SR:

- `f16v2absadd`: which performs basic op $\text{fp16_SR}(\text{fp32_RN}(|a| + |b|))$ on vectors.
- `f16v2add`: $\text{fp16_SR}(\text{fp32_RN}(a + b))$
- `f16v2sub`: $\text{fp16_SR}(\text{fp32_RN}(a - b))$
- `f16v2mul`: $\text{fp16_SR}(\text{fp32_RN}(a \times b))$
- `f16v4mix`: $\text{fp16_SR}(\text{fp32_RN}(a \times b + c \times d))$

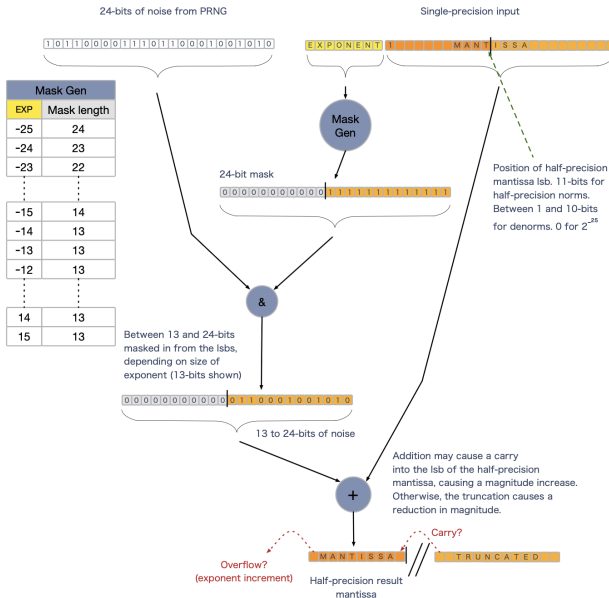
Conversion ops:

- `f16v2tof8`: convert two fp16 values to two fp8 values with SR, with pre-scaling.

fp16 arithmetic

This is in effect fp16 arithmetic with SR—assuming double rounding not a problem.

Graphcore Tile Vertex ISA 1.3.1



Key features:

- SR fp32 to fp16: 13 to 24 random bits.
- 13 bits for normalised numbers; more for subnorm.
- On underflows SR is not applied—round down to 0.0.
- Two fp8 formats supported: E4M3 and E5M2 (E - exponent bits, M - fraction bits).
- SR fp16 to fp8: 7 to 11 PRNG bits.

Stoch. round is part of cvt instruction: convert precisions.

Stoch. round fp32 values to {fp16, bf16, fp8, fp6, fp4}.

Similar concept

Perform operations by rounding to fp32, then use SR on fp32 to convert to target format. However, subnormal handling not specified.

How many random bits are used

fp16: 13 bits; bf16: 16 bits; for fp8, 6, 4 not specified—see fig below.

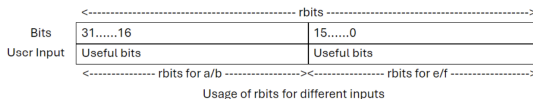


Figure 41: Random bits layout for .rs rounding with .e5m2x4/.e4m3x4/.e3m2x4/.e2m3x4 destination type

- `CVT_SR_FP8_F32`: stoch. round fp32 to E4M3 (20 random bits)
- `CVT_SR_BF8_F32`: fp32 to E5M2 (21 random bits).

FP8 and BF8 operations

FP8 and BF8 are used as inputs to matrix-mult op. with fp32 outputs.

FP8/BF8 to fp32

`V_CVT_F32_BF8`, `V_CVT_F32_FP8` convert up to fp32—can do basic ops and SR back to fp8.

Amazon NeuronCore v2

Documentation very high level. Simply says RN and SR supported.

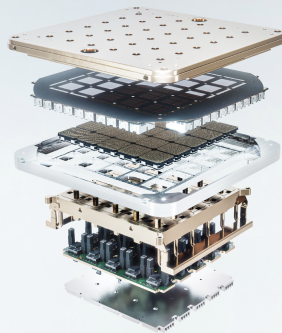
Data Type	S	Range	Precision	
FP32	1	8 bits	23 bits	
TF32	1	8 bits		10 bits
BF16	1	8 bits		7 bits
FP16	1	5 bits		10 bits
FP8_e5m21	1	5 bits		2 bits
FP8_e4m31	1	4 bits		3 bits
FP8_e3m41	1	3 bits		4 bits
UINT8		8 bits		

Tesla floating-point standard

- E4M3 and E5M2 8-bit formats (with configurable bias).
- Convert between fp8 and bf16/fp32 with RN or SR.
- Asks for reproducible arithmetic when seed is known.
- fp16 to/from fp32 with SR.
- unsigned fp16 (6 expon. bits) to/from fp32 with SR.

Tesla Dojo Technology

A Guide to Tesla's Configurable Floating Point Formats & Arithmetic



SR and random bit use across vendors

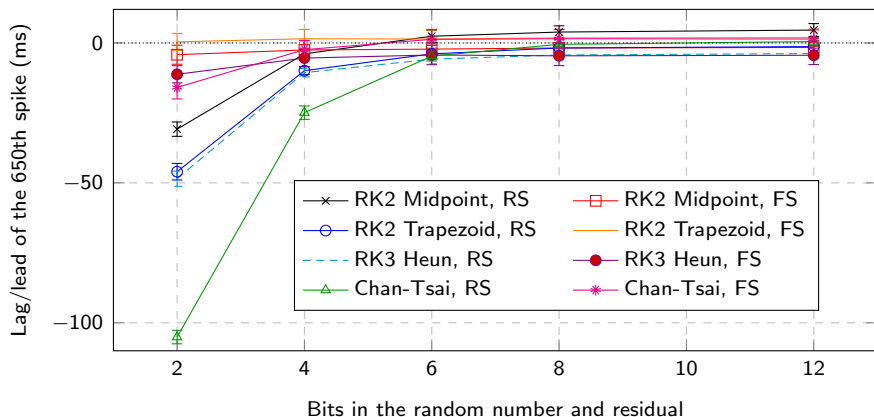
How many random bits are used in different SR conversion ops:

Vendor	fp32 to fp8	fp32 to fp16	fp32 to bf16	fp16 to fp8
Graphcore	-	13 – 24	-	7 – 11
NVIDIA	Up to 16	13	16	-
AMD	20/21	-	-	-
Amazon	?	?	?	?
Tesla	?	?	-	?

Towards standardisation: random number precision in SR

We may not need SR with 13 bits and higher.

We did some experiments with ODE solvers in fixed-point arithmetic ([Hopkins et al, 2020](#)). **See next talk for newer results.**




Key points

- SR available in latest NVIDIA and AMD—can be used for mixed-precision NLA algs.
- Random number precisions used differ across architectures—no reproducibility at present even if PRNG seeds are matched.
- No official standard, yet (see IEEE P3109 working in progress).

More details in the stochastic rounding survey paper

M. Croci, M. Fasi, N. J. Higham, T. Mary, and M. Mikaitis. *Stochastic rounding: implementation, error analysis and applications*. **R. Soc. Open Sci.** Mar. 2022.

 <https://bit.ly/3Kzw7mA>.

References I



J. M. Alben, P. Micikevicius, H. Wu, M. Y. Siu.
Stochastic Rounding of Numerical Values.
2019. Patent Status: [Active](#).



M. Fasi, M. Mikaitis
CPFloat: A C library for emulating low-precision arithmetic.
[ACM Trans. Math. Soft.](#), 49. 2023.



M. Hopkins, M. Mikaitis, D. R. Lester, S. Furber
Stochastic rounding and reduced-precision fixed-point arithmetic for
solving neural ordinary differential equations.
[Phil. Trans. R. Soc.](#), 378. 2020.