UNIVERSITY OF LEEDS

Discussion on Rounding (IEEE P3109)

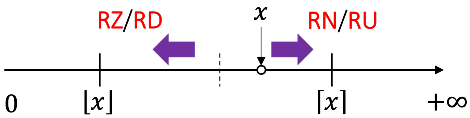
Mantas Mikaitis

School of Computing, University of Leeds, Leeds, UK

Standard for Arithmetic Formats for Machine Learning
IEEE P3109 Working Group
Oct. 16, 2023 (Virtual)

IEEE 754-2019: rounding for binary arithmetics

- Round-to-nearest (**RN**) (ties to even [default], ties to zero [augmented ops])
- Round-toward-zero (**RZ**)
- Round-down (**RD**)
- Round-up (**RU**)



Use of rounding modes

RN is usually enabled by default. Directed modes essential for **interval arithmetic**.

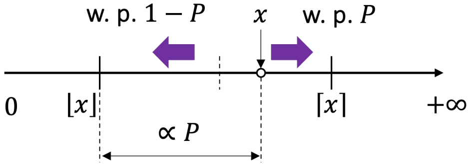
Round-Odd

- If inexact, round to the FP number with odd significand.
- Cheap to implement as we simply need to set the LSB of the significand to 1 if conditions met.
- No addition required for rounding. No need to check for overflow.
- Avoids issues when double rounding between three precisions, such as to 80 bits and then to 64 bits [Boldo and Melquiond, 2008]. Same applies in general, including low precisions.
- Appears in latest ARM instruction sets, for bfloat16 dot products and matrix multiply-accumulate. ARM reported 25% reduction in dot product area when only round-odd is implemented [Burgess et al. 2019].

Stochastic Rounding

Stochastic rounding (**SR**) rounds faithfully, rounding up/down with probabilities.

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 $P - 1$.



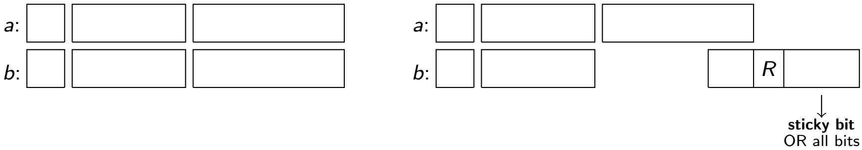
Mode 1 SR (nearness): $P = \frac{x - \lfloor x \rfloor}{\lceil x \rceil - \lfloor x \rfloor}$

Mode 1

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

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

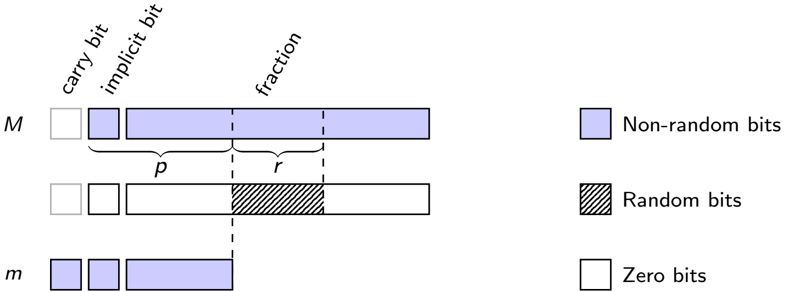
Guard bit

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 to be a high precision unrounded significand from an operation.

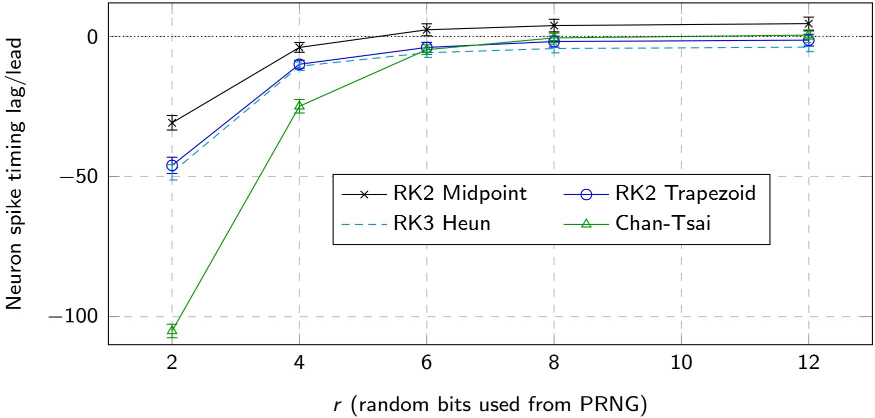
Take p to be source precision and r the precision of random numbers.



Random number precision experiments

The question of r, required number of random bits in SR, still open.

We did some experiments with ODE solvers in 32-bit fixed-point arithmetics [Hopkins et al, 2020]. Compared with binary64.



Towards the standardization of SR

We attempt to state basic properties [Croci et al. 2022]:

1. If the exact number is in the range of the target format, SR should be performed as though the number was originally held in $p + r$ bits and then rounded to p bits.
2. Overflows: if the exact number lies between the maximum representable number $\pm f_{max}$ and the neighbouring value that is not representable in the target format and will be treated as $\pm\infty$, SR is performed as though the value is representable, to preserve the statistical information about the round-off bits.

Towards the standardization of SR

3. When the exact number is smaller than the smallest value representable in the target format, SR should round stochastically to one of the two neighbouring floating-point values in the target format, either zero or the smallest representable value, maintaining the sign.
4. The above rule should apply even when subnormals are disabled, if that is supported in general.
5. $\pm\infty$, ± 0, and NaNs are not modified by stochastic rounding.

Key questions for IEEE P3109 discussion on SR

1. Should we set r to a specific value and ask for a specific PRNG algorithm, or leave these two parameters implementation defined? (We used *linear-feedback shift register* [Hopkins et al, 2020]).
2. What to do with the bits past the $p + r$ position before stochastic rounding occurs on the r bits?
3. (Conversation with J. Demmel which was shared with collegues) If for SR we need to hold $p + r$ bits of answer, why not just compute in $p + r$ bits with RN and round to p bits when done?

References I

S. Boldo, G. Melquiond
Emulation of a FMA and Correctly Rounded Sums: Proved Algorithms Using Rounding to Odd.
IEEE Trans. Comput. 57:4. 2008.

N. Burgess, J. Milanovic, N. Stephens, K. Monachopoulos, D. Mansell
Bfloat16 Processing for Neural Networks.
ARITH 2019.

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.

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.

This slide is intentionally blank.