Numerical Behavior of GPU Matrix Operations

M. Fasi*, N. J. Higham, M. Mikaitis & S. Pranesh

*Durham University
University of Manchester

Motivation

- Tensor cores (NVIDIA) and matrix engines (AMD) run matrix multiply-accumulate (MMA) in hardware.
- 145 of the June 2022 TOP500 supercomputers have **V100**, **A100** or **MI250X** GPUs with hardware MMA.
- Widely used in scientific computing.
- Numerical behavior is not necessarily the same as IEEE 754 software MMA.
- At least **six precisions now available**: fp8, fp16, bfloat16, TF32, fp32, fp64. Still increasing as NVIDIA H100 (late 2022) adds fp8.
- Explaining numerical behavior of hardware is essential for interpreting numerical results, gauging **reproducibility**, performing error analysis.

Hardware matrix ops

Tensor cores and matrix engines perform MMA on various size matrices. In the V100 and T4 GPUs, all of A, B, C, and D are 4×4. In the A100, up to 8×8, depending on the numerical type. Various sizes in the MI250X up to $32\times8\times32$. These are chained together to perform MMAs of any size. Mixed precision—inputs (binary16 in the V100) have lower precision than outputs (binary32). Features such as rounding, normalization, subnormal support might be different to software MMA with IEEE 754 arithmetic. Most of these features are not specified in the GPU manuals.

Tests for NVIDIA V100 and A100 hardware's IEEE 754-compliance.

$$\underbrace{D}_{\text{binary16 or binary32}} = \underbrace{C}_{\text{binary16 or binary32}} + \underbrace{A}_{\text{binary16}} \times \underbrace{B}_{\text{binary16}}$$

We found **round-toward-zero** in addition, >24 **bits in addition**, **normalization at the end** rather than after each addition.

For paper scan the QR.

Here we show a few details of testing the V100 GPU, but the techniques are independent of the device.

Accuracy of dot products

Set the first row of A and the first column of B to $1-2^{-11}$, $c_0 = 0$. Each product $a_{1,i} \times b_{i,1}$ should evaluate to $1-2^{-10}-2^{-22}$ if held exactly, or to $1-2^{-10}$ if rounded back to binary16. This test showed that binary16 products are not rounded back to binary16, and we demonstrated that this is true irrespective of whether C and D are set to binary16 or 32. For testing precision of addition, we set the first row of A to 1, which gives

$$d_{11} = b_{11} + b_{21} + b_{31} + b_{41} + c_{11}.$$

Then we ran 5 different permutations of four addends set to 2^{-24} (targeting binary32), with the 5th set to 1. All permutations returned $d_{11} = 1$, which means there are **up to four rounding errors** and **the addition starts from the highest magnitude addend**.

Rounding modes

In the expression of d_{11} above, we set $b_{11} = 2$, $b_{21} = 2^{-23} + 2^{-24}$, and the rest to 0. In binary32, which tensor cores notionally use in adding, we expect

$$\mathrm{RN}(b_{11}+b_{21}) = \mathrm{RU}(b_{11}+b_{21}) = 2+2^{-22},$$
$$\mathrm{RZ}(b_{11}+b_{21}) = \mathrm{RD}(b_{11}+b_{21}) = 2.$$

We did get 2. Then set $b_{11} = -2$, $b_{21} = -2^{-23} - 2^{-24}$. Here we expect $\mathrm{RZ}(b_{11}+b_{21}) = -2$, $\mathrm{RD}(b_{11}+b_{21}) = -2-2^{-22}$. Again 2 was the result, and we concluded that the **rounding mode in computing dot products is RZ**.

NLA Group Manchester