Accuracy–Performance Trade-offs in Ozaki-Inspired FP16 Matrix Multiplication on Qualcomm Mobile Hardware

Asli Elif Ozgur | Durham University | Supervisor: Dr Christopher Marcotte

LAIDLAW RESEARCH · 2026

0.80× error

split FP16 error relative to direct FP16

1.10× → 2.19×

split/direct GPU runtime from N=256 to N=1024

17.4× → 67.4×

direct GPU FP16 speedup over the documented CPU FP64 baseline

1 Research question

On the tested Qualcomm platform, does mobile-GPU FP16 provide competitive GEMM runtime relative to the documented CPU FP64 baseline, and how much numerical accuracy does a one-residual, two-component decomposition recover for its additional runtime?

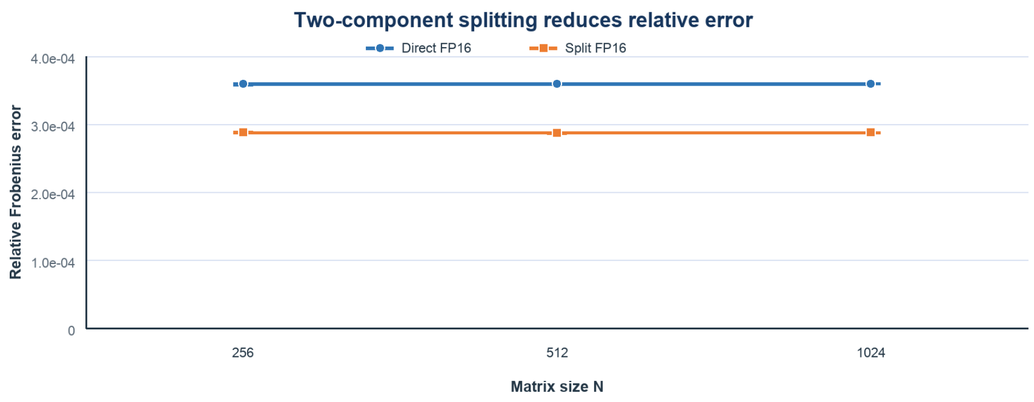
2 Method

- Direct GPU FP16 was compared with a two-component split graph and an eight-thread CPU FP64 baseline.
- Square matrices N = 256, 512 and 1024; five independent Gaussian matrix pairs per size.
- Direct graph: one MatMul. Split graph: four MatMul and three Add operations.
- QAIRT 2.47 used 16-bit floating-point graph tensors on the Qualcomm GPU.
- Each GPU case ran ten times; the cold first run was excluded and nine were averaged.
- Accuracy used relative Frobenius error against a NumPy FP64 reference, with 95% CIs across matrix cases.
- The CPU baseline was transparent hand-written C++, not tuned BLAS.

$$A \approx A_1 + A_2 \qquad B \approx B_1 + B_2$$

$$C_{split} = A_1B_1 + A_1B_2 + A_2B_1 + A_2B_2$$

3 Accuracy result

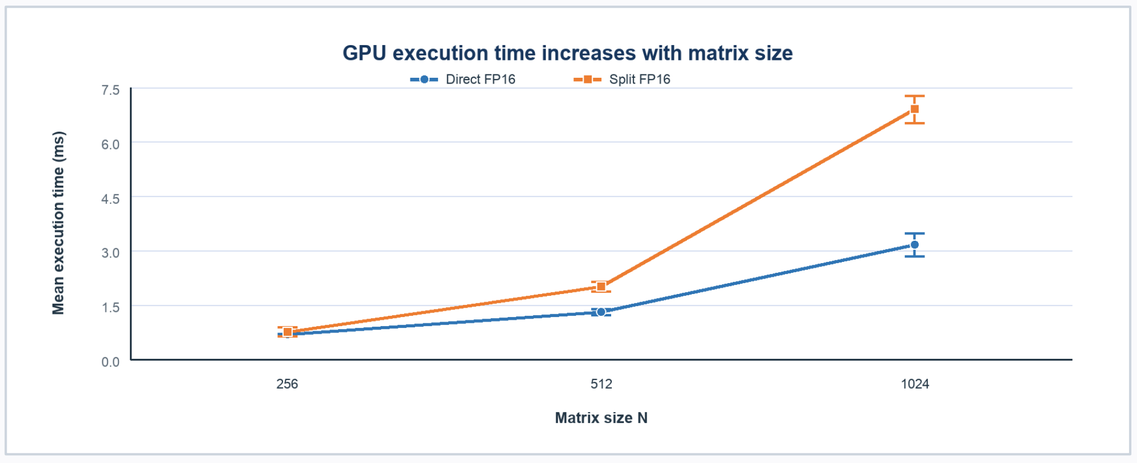


Across all three sizes, mean relative error fell from about 3.59×10^{-4} for direct FP16 to 2.88×10^{-4} for split FP16. Paired reductions ranged from 19.78% to 20.20%. The nearly flat curves show that the benefit was consistent for these Gaussian inputs. CPU FP64 error remained roughly 11 to 12 orders of magnitude smaller, so splitting recovered only part of the precision lost by direct FP16.

4 Interpretation

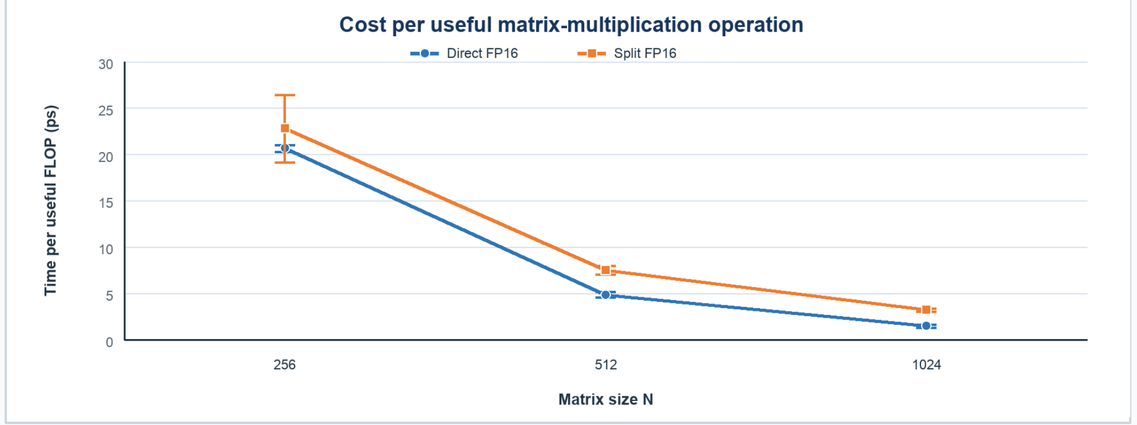
The result supports the motivation of Ozaki-style methods: residual components can preserve information lost in one direct rounding step. This experiment used only one residual per input, so it offers an intermediate balance between accuracy and runtime rather than a full error-free Ozaki implementation or an FP64 replacement. At N=512, splitting added about 0.71 ms for a 20% error reduction. Whether that exchange is worthwhile depends on the application's tolerance for latency and numerical error.

5 Runtime cost



Split/direct runtime rose from 1.10× at N=256 to 1.55× at N=512 and 2.19× at N=1024. At the smallest size, the 95% interval included parity; the overhead was clear at the two larger sizes. Doubling N did not produce simple eightfold timing growth, so these sizes were not yet in a purely asymptotic compute-bound regime.

6 Cost per useful FLOP



Using the same useful-work count, $2N^3$, direct cost fell from 20.65 to 1.47 ps per useful FLOP; split cost fell from 22.80 to 3.21 ps. This indicates improving GPU utilisation as matrices grew. The normalization supports fair comparison across sizes, but it does not change the split/direct ratio at a fixed N or establish energy efficiency.

7 Limitations

- One Qualcomm platform, three square sizes and five Gaussian cases per size.
- CPU FP64 used hand-written eight-thread C++, not tuned BLAS; no matched CPU FP32 test was run.
- Only zero and one residual were measured, so performance for more components is untested.
- QAIRT did not disclose accumulator width, kernel schedule or memory traffic.
- Android power readings were unreliable, so runtime is not presented as an energy result.

CONCLUSION

One residual component reduced direct FP16 matrix-multiplication error by about 20%, but the runtime penalty increased with matrix size. On this Qualcomm platform, splitting offered a measurable middle ground between direct FP16 speed and FP64-level accuracy, but it was not a substitute for higher precision.

References [1] Ozaki et al., NOLTA 4(1), 2–11, 2013. [2] Utsugiri & Kouya, arXiv:2301.09960, 2023. [3] Mukunoki et al., ISC High Performance 2020, 230–248. [4] Higham, Accuracy and Stability of Numerical Algorithms, 2nd ed., 2002. [5] IEEE 754-2019. [6] PyTorch numerical-accuracy documentation. [7] Qualcomm QIDK/QAIRT documentation.

Values are means across five independent matrix cases; error bars show 95% confidence intervals.

Contact: phrr81@durham.ac.uk | LinkedIn: linkedin.com/in/asli-ozgr | *With thanks to my supervisor, Dr Christopher Marcotte, for his guidance throughout this project.*