Multiple Double Arithmetic on Graphics Processing Units

Jan Verschelde[†]

University of Illinois at Chicago
Department of Mathematics, Statistics, and Computer Science
http://www.math.uic.edu/~jan
https://github.com/janverschelde
https://www.youtube.com/@janverschelde5226
janv@uic.edu

HPC devroom, FOSDEM 2023, 5 February

[†]Supported by the National Science Foundation, DMS 1854513.

multiple doubles / error free transformations

A multiple double is an unevaluated sum of nonoverlapping doubles.

The 2-norm of a vector of dimension 64 of random complex numbers on the unit circle equals 8. Observe the second double:

```
double double : 8.00000000000000E+00 - 6.47112461314111E-32
  quad double : 8.00000000000000E+00 + 3.20475411419393E-65
  octo double : 8.00000000000000E+00 - 9.72609915198313E-129
```

If the result fits a 64-bit double,
then the second double represents the accuracy of the result.

Advantages and disadvantages:

- \+ predictable overhead, cost of double double $\approx$ complex double
- \+ exploits hardware double arithmetic
- – not true multiprecision, fixed multiples of bits in fractions
- – infinitesimal computations not possible because of fixed exponent

motivation: power series arithmetic

$$\exp(t) = \sum_{k=0}^{d-1} \frac{t^k}{k!} + O(t^d).$$

Assuming the quadratic convergence of Newton's method:

k	$1/k!$	recommended precision	`eps`
7	`2.0e-004`	double precision okay	`2.2e-16`
15	`7.7e-013`	use double doubles	`4.9e-32`
23	`3.9e-023`	use double doubles	
31	`1.2e-034`	use quad doubles	`6.1e-64`
47	`3.9e-060`	use octo doubles	`4.6e-128`
63	`5.0e-088`	use octo doubles	
95	`9.7e-149`	need hexa doubles	`5.3e-256`
127	`3.3e-214`	need hexa doubles	

`eps` is the multiple double precision

software packages

- QDlib by Y. Hida, X. S. Li, and D. H. Bailey.
  Algorithms for quad-double precision floating point arithmetic.
  In the *Proceedings of the 15th IEEE Symposium on Computer Arithmetic*, pages 155–162, 2001.

- GQD by M. Lu, B. He, and Q. Luo.
  Supporting extended precision on graphics processors. In the *Proceedings of the Sixth International Workshop on Data Management on New Hardware (DaMoN 2010)*, pages 19–26, 2010.

- CAMPARY by M. Joldes, J.-M. Muller, V. Popescu, and W. Tucker.
  CAMPARY: Cuda Multiple Precision Arithmetic Library and Applications. In *Mathematical Software – ICMS 2016, the 5th International Conference on Mathematical Software*, pages 232–240, Springer-Verlag, 2016.

cost overhead factors — Graphics Processing Units

The number of floating-point operations for multiple double arithmetic are cost overhead factors:

	+	$\star$	/	average
`double double`	20	32	70	37.7
`quad double`	89	336	893	439.3
`octo double`	269	1742	5126	2379.0

Teraflop performance compensates the overhead of quad doubles.

NVIDIA GPU	CUDA	#MP	#cores/MP	#cores	GHz
Pascal P100	6.0	56	64	3584	1.33
Volta V100	7.0	80	64	5120	1.91
GeForce RTX 2080	7.5	46	64	2944	1.10

The double precision peak performance of the P100 is 4.7 TFLOPS. At 7.9 TFLOPS, the V100 is 1.68 times faster than the P100.

customized software

The code for the arithmetical operations generated by the CAMPARY software was customized for each precision.

- Instead of representing a quad double by an array of four doubles, all arithmetical operations work on four separate variables, one for each double.
  By this customization an array of quad doubles is stored as four separate arrays of doubles and a matrix of quad doubles is represented by four matrices of doubles.
- The `double2` and `double4` types of the CUDA SDK work for double double and quad double, but not for the more general multiple double arithmetic.
- QDlib provides definitions for the square roots and various other useful functions for double double and quad double arithmetic. Those definitions are extended to octo double precision.

2-norm of vector of complex numbers

Let $\mathbf{z} = (z_1, z_2, \ldots, z_n)$ be a vector of complex numbers.

$$\|\mathbf{z}\|_2 = \sqrt{\mathbf{z}^H \mathbf{z}} = \sqrt{\overline{z_1} z_1 + \overline{z_2} z_2 + \cdots + \overline{z_n} z_n}$$

- For small n, only one block of threads is launched.
- The prefix sum algorithm proceeds in $\log_2(n)$ steps.
- For large n, subsums are accumulated for each block.

Specific for the accelerated multiple double implementation:

- The size of shared memory needs to be set for each precision.
- $\sqrt{}$ applies Newton's method for staggered precisions.
- The use of registers increases as the precision increases ...

two publications and a preprint

- **Accelerated polynomial evaluation and differentiation at power series in multiple double precision.**
  In the *2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 740–749. IEEE, 2021.
  `arXiv:2101.10881`

- **Least squares on GPUs in multiple double precision.** In the *2022 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 828–837. IEEE, 2022.
  `arXiv:2110.08375`

- **GPU Accelerated Newton for Taylor Series Solutions of Polynomial Homotopies in Multiple Double Precision.**
  `arXiv:2301.12659`

GPL-3.0 License, code at `github.com/janverschelde/PHCpack`