Accurate dot products with FMA

S. Graillat, Ph. Langlois and N. Louvet
(graillat, langlois, nicolas.louvet)@univ-perp.fr
DALI-LP2A Laboratory, University of Perpignan, France

Accuracy of Classic Dot Product

We consider dot products without FMA:

- Classic dot product
  \[ \text{function } r = \text{Dot}(x,y) \text{ for } n \in \mathbb{N} \]
  \[ x \in \mathbb{R}_n, y \in \mathbb{R}_n \]

  - The condition number for dot product computation is
  \[ \text{cond}(x,y) = \frac{\|x\| \cdot \|y\|}{\|\text{Dot}(x,y)\|} = \frac{\|x\| \cdot \|y\|}{\|x \cdot y\|} \]

  - Worst case accuracy: FMA does not improve the accuracy of computed dot product since Dot and DotFMA both verify:
  \[ \text{cond}(x,y) = \frac{\|x\| \cdot \|y\|}{\|\text{Dot}(x,y)\|} \]

  - Practical accuracy: FMA only slightly improves the actual accuracy.

  \[ \text{accuracy (computed) vs (actual)} \]

Compensated Dot Products

- More accuracy can be achieved thanks to double-double computations (see Algorithms DotXBLAS below).
- or with compensated algorithms: the forward error in the floating point evaluation of \( r' \) is
  \[ \varepsilon = r' - \text{computed}(x,y) \]

  - The main idea is to compute an approximate \( r \) of the global error \( \varepsilon \) thanks to Error Free Transformations (EFT). Then a compensated result \( r^c \) is provided correcting the computed \( r' \) as follows.
  \[ r^c = r + \varepsilon = r' + \text{corrected}(x,y) \]

  - From Dot and DotFMA, we derive two compensated algorithms using EFT (259u: 2ProdFMA and 3ProdFMA are presented in the EFT paper below).
  - CompDigit: correcting \( r \) and \( x \in \text{Dot} \) with 259u and 2ProdFMA (see [1]).
  - CompDigitFMA: correcting FMA in DotFMA with 3FMA.

Error Free Transformations (EFT)

- Error Free Transformations are properties and algorithms to compute the generated rounding errors at the working precision \( u \). The following table sums up the EFT for \( \gamma \) and FMA.

  | \( x \), \( y \), \( z \) and \( k \) belongs in \( \mathbb{R}_n \) if \( n \) is in \( \mathbb{N} \) |
|-----------------|-----------------|
| 1 < \( \gamma \) < 2 | 2 < \( k \) < 5 |
| 2 < \( k \) < 5 | 5 < \( k \) < 10 |
| 5 < \( k \) < 10 | \( k \) belongs in \( \mathbb{R}_n \) |

Remark: \( x \), \( y \), \( z \) and \( k \) belong to \( \mathbb{R} \) when \( n \) and \( k \) are in \( \mathbb{N} \).

Conclusions

- XBLAS Dot Product
  - BLAS – Bailey’s double-double – x16 rounded and mixed precision BLAS [3]
  - A double-double number \( \text{unrounded} \) sum of two IEEE-754 double precision numbers \( \gamma \) at least 138 significant bits.
  - DotXBLAS – Classic dot product \( \text{Dot} \) – double-double.
  - DotXBLAS also benefits from the availability of FMA.

- XBLAS with FMA – Bailey’s double-double – x16 rounded and mixed precision BLAS [5]
  - A double-double number \( \text{unrounded} \) sum of two IEEE-754 double precision numbers \( \gamma \) at least 138 significant bits.

- CompDigitFMA is about 8 times faster than XBLAS algorithms.

References
