Error-free Tables for Trigonometric Function Evaluation
Hugues de Lassus Saint-Geniès, David Defour, Guillaume Revy

To cite this version:

HAL Id: lirmm-01273490
https://hal-lirmm.ccsd.cnrs.fr/lirmm-01273490
Submitted on 12 Feb 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
1. How does a modern processor calculate sine and cosine?

- **Range reduction**: use of trigonometric identities
  - \( \sin(\pi - x) = \sin(x) \)
  - \( \sin(x) = \pm \sqrt{1 - x^2} \) with \( f_1 \in \{\sin, \cos\} \)
  - Range reduction \( f_3 \rightarrow \{0, \frac{\pi}{2}\} \)

- **Evaluation**: use of other trigonometric properties
  - \( \sin(x) = \frac{\text{f1}(x)}{\sqrt{1 + f_1^2}} \)
  - \( \cos(x) = \frac{1}{\sqrt{1 + f_1^2}} \)

- **Tabulated values for sine and cosine**: [Tan91]

2. Pythagorean Triples

- **What is a Pythagorean triple?**

  \[ (a, b, c) \in \mathbb{N}^3 \]
  \[ a^2 + b^2 = c^2 \]
  \[ \sin(\theta) = \frac{a}{c} \quad \cos(\theta) = \frac{b}{c} \]

- **Primitive Pythagorean Triple**: a Pythagorean triple \((a, b, c)\) for which \(\gcd(a, b, c) = 1\).

3. Primitive Pythagorean Triple Generation

- **Barning-Hall ternary-tree structure**:

  \[
  \begin{array}{cccc}
  \{1, -2, 2\} & \{1, 2, 2\} & \{2, 2, 2\} \\
  \{2, -1, 2\} & \{1, 2, 2\} & \{2, 2, 2\} \\
  \{2, -1, 2\} & \{2, -1, 2\} & \{2, 2, 2\} \\
  \end{array}
  \]

- **Several equivalent trees, easy to implement**

- **Proven to generate all primitive triples by increasing hypotenuse lengths**: [Bar63]

4. Primitive Pythagorean Triple Selection

- **Only one triple per entry needed**

- **Division by 4 removed by incorporating it in polynomial approximations**

- **Same hypotenuse**: \(c\) needed for all entries

- **Scale**: select PPTs to the least common multiple of their hypotenuses

5. Exhaustive Search for a Small Common Hypotenuse

- **Algorithm**
  1. \( n \leftarrow 4 \)
  2. repeat
  3. Generate all PPTs \((a, b, c)\) such that \(c \leq 2^n \)
  4. Search for the LCM \(L\) among all generated hypotenuses \(c\).
  5. \( k \leftarrow n + 1 \)
  6. until such a \( k \) is found
  7. Build tabulated values \((A, B, ccorr)\) for every entry.

- **Results**

6. Heuristic Search

- **Prime factorization** of found common multiples
  - \( c \in \mathbb{P} \)
  - \( r_1 \in (1, 11) \) if \( p_1 \neq 5 \)
  - \( r_2 \in \mathbb{N} \)
  - \( r_3 \neq 0 \) else
  - \( r_4 \) and \( p_1 \in \mathbb{P} \)

- **Results**

7. Theoretical Gains

- **Comparison between three table-based range reductions**, for \( p = 10 \): The number of memory accesses (MA) and the number of floating point operations (FLOP) are reported.

- **Heuristic**: store primitive Pythagorean triples satisfying \( c \in \mathbb{P} \) with

- **Results**

References

