Hindawi
Mathematical Problems in Engineering
Volume 2017, Article ID 1729287, 10 pages
https://doi.org/10.1155/2017/1729287

Research Article

Accurate and Efficient Evaluation of Chebyshev Tensor Product Surface

Keshan He,[1] Peibing Du,[1] Hao Jiang,[2] Chongwen Duan,[3] Hongxia Wang,[1] and Lizhi Cheng[1]

[1]*School of Science, National University of Defense Technology, Changsha, China*
[2]*College of Computer, National University of Defense Technology, Changsha, China*
[3]*State Key Laboratory of Astronautic Dynamics, Xi'an, China*

Correspondence should be addressed to Hao Jiang; haojiang@nudt.edu.cn

Received 10 March 2017; Accepted 3 July 2017; Published 27 September 2017

Academic Editor: Elisa Francomano

A Chebyshev tensor product surface is widely used in image analysis and numerical approximation. This article illustrates an accurate evaluation for the surface in form of Chebyshev tensor product. This algorithm is based on the application of error-free transformations to improve the traditional Clenshaw Chebyshev tensor product algorithm. Our error analysis shows that the error bound is $u + \mathcal{O}(u^2) \times \mathtt{cond}(P, x, y)$ in contrast to classic scheme $u \times \mathtt{cond}(P, x, y)$, where u is working precision and $\mathtt{cond}(P, x, y)$ is a condition number of bivariate polynomial $P(x, y)$, which means that the accuracy of the computed result is similar to that produced by classical approach with twice working precision. Numerical experiments verify that the proposed algorithm is stable and efficient.

1. Introduction

Chebyshev polynomials have been extended to almost all mathematical and physical discipline, including spectral methods, approximation theory, and representation of potentials [1–4]. Bivariate Chebyshev polynomials have gained attention of the computer vision researchers [5, 6]. Over the years, researchers have focused on the implementation of Chebyshev tensor product series in image analysis [5–7]. The Chebyshev tensor product series can be used to approximate an image, which is essentially regarded as a two-dimensional spatial function [8]. Two separable univariate Chebyshev polynomials that are discrete and orthogonal can approximate two-dimensional signal. Mukundan et al. [5] introduce a new discrete Chebyshev tensor product based on Chebyshev polynomials. This discrete Chebeshev tensor product functions show the effectiveness as feature descriptors. Rahmalan et al. [6] propose a novel approach based on discrete orthogonal Chebyshev tensor product for an efficient image compression. Recently, Omar et al. [7] propose a novel method for fusing images using Chebyshev tensor product series. All above need an image reconstruction from a finite Chebyshev tensor product surface. Thus, developing fast and reliable algorithms to evaluate the Chebyshev tensor product series are of challenging interest [9]. The Clenshaw tensor product algorithm (CTP) [10] is one of algorithms that are used to evaluate Chebyshev tensor product series.

In order to get a high-precision approximation of an image, it is essential to evaluate the series accurately. Particularly, we require higher level of accurate numeric results for ill-conditioned cases. Li et al.'s double-double [11] (double-double numbers are represented as an unevaluated sum of a leading double and a trailing double) is a library used to improve the accuracy of numerical computation. However, the algorithm is time-consuming when an input image becomes larger.

Error-free transformation studied by Ogita et al. [12] is another direct possible method to improve the accuracy apart from increasing the working precision. Compensated algorithms to evaluate the univariate polynomials in different basis have been proposed in [12–15]. Inspired by their work, we extend the univariate compensated algorithm to tensor product case using the compensated Clenshaw algorithm [16] for evaluation of Chebyshev series [15]. We perform a com-

pensated Clenshaw tensor product algorithm (for simplicity we denote it by `CompCTP` algorithm) to evaluate the polynomials expressed in Chebyshev tensor product form. The proposed algorithm produces the same accuracy as using twice working precision.

Since the image is fundamentally treated as two-dimensional spatial function, we use general two-dimensional function to illustrate our algorithm in the sequel. This paper has the following layout. In Section 2, we introduce some preliminaries and basic algorithms underlying our algorithm. In Section 3, we propose the compensated algorithm to compute surface in form of Chebyshev tensor product. In Section 4, we analyze forward error bound of the algorithm. In Section 5, a series of numerical experiments illustrate the accuracy and efficiency of the proposed algorithm.

2. Preliminaries and Error-Free Transformations

2.1. Basic Notations. At the present time, IEEE 64-bit floating arithmetic standard is implemented, which is sufficiently accurate for most scientific applications. Throughout the paper, we presume that all the computations are performed using IEEE-754 [17] standard in double precision so that neither overflow nor underflow occurs.

We assume that the computations are produced in a floating-point arithmetic which yields the models

$$\mathrm{fl}(x \ \mathrm{op} \ y) = (x \ \mathrm{op} \ y)(1+\varepsilon_1) = \frac{(x \ \mathrm{op} \ y)}{1+\varepsilon_2}, \quad |\varepsilon_1|, |\varepsilon_2| \le u, \tag{1}$$

where $\mathrm{op} \in \{+, -, \times, /\}$ and u is the working precision. For brevity we denote $\mathrm{fl}(x \ \mathrm{op} \ y) = x \circ y$, $\circ \in \{\oplus, \ominus, \otimes, \oslash\}$. Besides, we denote $\gamma_n \coloneqq nu/(1-nu) = nu + \mathcal{O}(u^2)$ [18] and use $\widehat{a}$ and $\mathrm{fl}(a)$ as the computed element of a.

Finally, we recall the Chebyshev polynomials and analogous Chebyshev polynomials [15, 19]. The forms of Chebyshev polynomials and analogous Chebyshev polynomials definite in the interval $[-1, 1]$ with three term recurrence are shown in (2) and (3), respectively.

$$\begin{aligned} T_0(x) &= 1, \\ T_1(x) &= x, \\ &\vdots \\ T_{n+1}(x) &= 2xT_n(x) - T_{n-1}(x), \end{aligned} \tag{2}$$

$$\begin{aligned} \widetilde{T}_0(x) &= 1, \\ \widetilde{T}_1(x) &= x, \\ &\vdots \\ \widetilde{T}_{n+1}(x) &= 2x\widetilde{T}_n(x) + \widetilde{T}_{n-1}(x). \end{aligned} \tag{3}$$

2.2. Error-Free Transformations. Rounding errors are an unavoidable consequence of working in finite precision arithmetic [18]. Error-free transformations (EFTs) are a technology of the floating-point operation $+, -, \times$, which transforms any pair of floating-point numbers (a, b) into a new pair (x, y) with $x = \mathrm{fl}(a \ \mathrm{op} \ b)$ and $a \ \mathrm{op} \ b = x + y$ to obtain an accurate result. Two algorithms of EFTs are Donald et al.'s `TwoSum` [20] (compensated summation of two floating-point numbers) and Dekker's `TwoProd` algorithm [21] (compensated product of two floating-point numbers).

The Clenshaw algorithm [16] is a recursive method to compute a linear combination of Chebyshev series $p(x) = \sum_{j=0}^{n} c_j T_j(x)$. Reviewing work [15], the forward error bound of Clenshaw algorithm satisfies (4).

Theorem 1 (see [15]). *Let $p(x) = \sum_{j=0}^{n} c_j T_j(x)$ be a polynomial at point x and Clenshaw (p, x) denote the numerical result of Clenshaw algorithm; then*

$$\left|\mathtt{Clenshaw}(p, x) - p(x)\right| \le \gamma_{3n-1} \sum_{j=0}^{n} \left|c_j\right| \widetilde{T}_j(|x|). \tag{4}$$

Combining EFTs with Clenshaw algorithm, [15] proposes a compensated Clenshaw algorithm (`CompClenshaw` in Algorithms 3–7) to evaluate univariate finite Chebyshev series. The algorithm shows a smaller forward error bound than Clenshaw algorithm.

Theorem 2 (see [15]). *Let $p(x) = \sum_{j=0}^{n} c_j T_j(x)$ be a finite Chebyshev series. The forward error bound of compensated Clenshaw algorithm (`CompClenshaw`) verifies*

$$\begin{aligned} &\left|\mathtt{CompClenshaw}(p, x) - p(x)\right| \\ &\quad \le u\left|p(x)\right| + \gamma_{3n-1}^2 \sum_{j=0}^{n} \left|c_j\right| \widetilde{T}_j(|x|). \end{aligned} \tag{5}$$

Since the element γ_{3n-1}^2 is $\mathcal{O}(u^2)$, comparing to (4), `CompClenshaw` is more stable to get an accurate result.

3. Accurate Algorithm to Evaluate Chebyshev Tensor Product Surface

In this section, we perform a compensated algorithm (`CompCTP`) to evaluate Chebyshev tensor product series based on EFTs. The technology is to extend compensated Clenshaw algorithm to polynomials expressed in Chebyshev tensor product. In order to extend Clenshaw algorithm to tensor product case, we express the series as

$$\begin{aligned} P(x, y) &= \sum_{i=0}^{m} \left(\sum_{j=0}^{n} a_{ij} T_j(y) \right) T_i(x) \\ &\coloneqq \sum_{i=0}^{m} \alpha(i, y) T_i(x). \end{aligned} \tag{6}$$

Therefore, we write Clenshaw tensor product algorithm (`CTP`) to evaluate the Chebyshev tensor product series with a nested Clenshaw algorithm (Algorithm 1).

function res = CTP(P, x, y) % assuming P is the coefficients matrix of polynomial $P(x, y)$
$\alpha = zeros(1, m)$
for $i = 0 : m$ **do**
  $\widehat{\alpha}(i)$ = Clenshaw($P(i, :), y$)
end for
$\widehat{\alpha}(0)$ = Clenshaw($\widehat{\alpha}, x$)
CTP(P, x, y) = $\widehat{P}(x, y) = res = \widehat{\alpha}(0)$ % $\widehat{P}(x, y)$ is numerical result of $P(x, y)$ using Clenshaw algorithm

Algorithm 1: Clenshaw algorithm for evaluation of Chebyshev tensor product surface.

function [$res_{comp}, res_{org}, err$] = CompCTP($P, x, y$)
$\alpha = zeros(1, m)$
for $i = 0 : m$ **do**
  [$\widehat{\alpha}(i), \widehat{e}_1(i)$] = CompClenshaw($P(i, :), y$)
end for
[$\widehat{\alpha}(0), \widehat{e}_2$] = CompClenshaw($\widehat{\alpha}, x$)
$\widehat{e}_3$ = Clenshaw($\widehat{e}_1, x$)
$err = \widehat{e}_2 \oplus \widehat{e}_3$
$res_{org} = \widehat{P}(x, y) = \widehat{\alpha}(0)$
CompCTP(P, x, y) = $res_{comp} = res_{org} \oplus err$

Algorithm 2: Compensated Clenshaw algorithm for evaluation of Chebyshev tensor product surface.

Substituting CompClenshaw for Clenshaw, we put forward a compensated Clenshaw tensor product algorithm to improve CTP algorithm. We call the compensated CTP algorithm as CompCTP (Algorithm 2).

According to Algorithm 2, combining

$$\sum_{i=0}^{m} \widehat{\alpha}(i) T_i(x) = \widehat{P}(x, y) + e_2,$$
$$\alpha(i) = \widehat{\alpha}(i) + e_1(i), \quad 0 \le i \le m, \tag{7}$$
$$e_3 = \sum_{i=0}^{m} e_1(i) T_i(x), \quad 0 \le i \le m,$$

we have

$$\begin{aligned} \widehat{P}(x, y) + e_2 &= \sum_{i=0}^{m} (\alpha(i) - e_1(i)) T_i(x) \\ &= \sum_{i=0}^{m} \sum_{j=0}^{n} a_{ij} T_i(x) T_j(x) - \sum_{i=0}^{m} e_1(i) T_i(x) \\ &= \sum_{i=0}^{m} \sum_{j=0}^{n} a_{ij} T_i(x) T_j(x) - e_3; \end{aligned} \tag{8}$$

that is

$$P(x, y) = \widehat{P}(x, y) + e, \tag{9}$$

where $e = e_2 + e_3$, $e_1(i)$ is the theoretical error produced by $\widehat{\alpha}(i)$ = Clenshaw($P(i, :), y$) at ith step, and e_2 is the theoretical error generated by $\widehat{\alpha}(0)$ = Clenshaw($\widehat{\alpha}(i), x$).

Based on previous analysis, we apparently know that $\widehat{e} = \widehat{e}_2 \oplus \widehat{e}_3$ is the approximation of e. So the result of CompCTP algorithm $\overline{P}(x, y) = \widehat{P}(x, y) \oplus \widehat{e}$ is more accurate than the floating-point result $\widehat{P}(x, y)$ of Algorithm 1.

4. Error Analysis of CompCTP Algorithm

In this section, we carry out an error bound of ComCTP algorithm for Chebyshev tensor product surface. Firstly, we consider the error bound of the CTP algorithm. According to Theorem 1, we deduct the Theorem 3.

Theorem 3. *Let us consider a Chebyshev tensor product surface* $P(x, y) = \sum_{i=0}^{m} \sum_{j=0}^{n} a_{ij} T_i(x) T_j(y)$ *and suppose that* $3(m + n)u < 1$, *where* u *is the working precision. Then the value* $\widehat{P}(x, y)$ *computed in floating-point arithmetic through Algorithm 1 satisfies*

$$\begin{aligned} &\left|\widehat{P}(x, y) - P(x, y)\right| \\ &\quad \le \gamma_{3(m+n)-2} \sum_{i=0}^{m} \sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_i(|x|) \widetilde{T}_j(|y|). \end{aligned} \tag{10}$$

Proof. Suppose $\widehat{P}_y(x) := \sum_{i=0}^{m} \widehat{p}_i(y) T_i(x)$, where $\widehat{p}_i(y)$ is the numerical result of Clenshaw algorithm by $\sum_{j=0}^{n} a_{ij} T_j(y)$ at ith step. Using Theorem 1, we obtain

$$\begin{aligned} &\left|P(x, y) - \widehat{P}(x, y)\right| \\ &\quad \le \left|P(x, y) - \widehat{P}_y(x)\right| + \left|\widehat{P}_y(x) - \widehat{P}(x, y)\right| \\ &\quad \le \sum_{i=0}^{m} \left|T_i(|x|)\right| \left|\sum_{j=0}^{n} a_{ij} T_j(y) - \widehat{p}_i(y)\right| \\ &\qquad + \left|\sum_{i=0}^{m} \widehat{p}_i(y) T_i(x) - \widehat{P}(x, y)\right| \\ &\quad \le \gamma_{3n-1} \sum_{i=0}^{m} \sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_i(|x|) \widetilde{T}_j(|y|) \\ &\qquad + \gamma_{3m-1} \sum_{i=0}^{m} \left|\widehat{p}_i(y)\right| \widetilde{T}_i(|x|) \end{aligned}$$

$[p, vact, cact] = \texttt{GenPolyCTP}(m, n, x, y, v, cexp)$
% v—the expected evaluation of the polynomial;
% $cexp$—the expected condition number;
% x, y—the coordinates;
% $vact$—the actual evaluation of the polynomial;
% $cact$—the actual condition number;
$a_{i,j} = 0$, for $i = 0 : m$, $j = 0 : n$;
$num = (m + 1) * (n + 1)$;
$d2 = ceil(num/2)$;
$vb = \log2(cexp * abs(v))$;
$I = J = zeros(1, num)$;
$perm = randperm(num)$;
Change $perm$ to coordinates (I, J),
where $(I(i), J(i))$ is a random coefficient of the surface in accord with $perm(i)$ in some sort order;
$a_{I(1),J(1)} = (2 * rand - 1)/T_{I(1)}(x)T_{J(1)}(y)$;
$a_{I(2),J(2)} = (2 * rand - 1) * 2^{vb}/T_{I(2)}(x)T_{J(2)}(y)$;
for $i = 3 : d2$ **do**
    $a_{I(i),J(i)} = (2 * rand - 1) * 2^{vb*rand}/T_{I(i)}(x)T_{J(i)}(y)$;
end for
$\log2v = \log2(abs(v))$;
$m = [(2 * rand(1, num - d2) - 1) * 2^{linspace(vb,\log2v,num-d2)}]$;
for $i = d2 + 1 : num - 1$ **do**
    $a_{I(i),J(i)} = (m(i - d2) - \texttt{QDCTPs}(F, x, y))/T_{I(i)}(x)T_{J(i)}(y)$;
end for
$a_{I(num),J(num)} = (v - \texttt{QDCTP}(F, x, y))/T_{I(num)}(x)T_{J(num)}(y)$;
$vact = \texttt{QDCTP}(P, x, y)$;
$cact = \texttt{QDCTP}(abs(P), x, y)/abs(vact)$;

Algorithm 3: Generate polynomial $P(x, y) = \sum_{i=0}^{m} \sum_{j=0}^{n} a_{ij} T_i(x) T_i(y)$ in the form of Chebyshev tensor product.

function $res = \texttt{Clenshaw}(p, x)$
$b_{n+2} = b_{n+1} = 0$
for $j = n : -1 : 1$ **do**
    $b_j = 2xb_{j+1} - b_{j+2} + c_j$
end for
$b_0 = xb_1 - b_2 + c_0$
$\texttt{Clenshaw}(p, x) = res = b_0$

Algorithm 4: [16] Clenshaw algorithm to evaluate finite Chebyshev series.

$$\begin{aligned}
&\le \gamma_{3n-1} \sum_{i=0}^{m} \sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_i\left(|x|\right) \widetilde{T}_j\left(\left|y\right|\right) \\
&\quad + \gamma_{3m-1}\left(1 + \gamma_{3n-1}\right) \sum_{i=0}^{m} \sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_i\left(|x|\right) \widetilde{T}_j\left(\left|y\right|\right) \\
&\le \gamma_{3(m+n)-2} \sum_{i=0}^{m} \sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_i\left(|x|\right) \widetilde{T}_j\left(\left|y\right|\right). \qquad (11)
\end{aligned}$$

□

function $res = \texttt{CompClenshaw}(p, x)$
$\widehat{b}_{n+2} = \widehat{b}_{n+1} = 0$
$\varepsilon b_{n+2} = \varepsilon b_{n+1} = 0$
for $j = n : -1 : 1$ **do**
    $[s, \pi_j] = \texttt{TwoProd}(\widehat{b}_{j+1}, 2x)$
    $[v, \sigma_j] = \texttt{TwoSum}(s, -\widehat{b}_{j+2})$
    $[\widehat{b}_j, \beta_j] = \texttt{TwoSum}(v, c_j)$
    $\widehat{\omega}_j = \pi_j \oplus \sigma_j \oplus \beta_j$
    $\varepsilon\widehat{b}_j = 2x \otimes \varepsilon\widehat{b}_{j+1} \ominus \varepsilon\widehat{b}_{j+2} \oplus \widehat{\omega}_j$
end for
$[s, \pi_0] = \texttt{TwoProd}(\widehat{b}_1, x)$
$[v, \sigma_0] = \texttt{TwoSum}(s, -\widehat{b}_2)$
$[\widehat{b}_0, \beta_0] = \texttt{TwoSum}(v, c_0)$
$\widehat{\omega}_0 = \pi_0 \oplus \sigma_0 \oplus \beta_0$
$\varepsilon\widehat{b}_0 = x \otimes \varepsilon\widehat{b}_1 \ominus \varepsilon\widehat{b}_2 \oplus \widehat{\omega}_0$
$\texttt{CompClenshaw}(p, x) = res = \widehat{b}_0 \oplus \varepsilon b_0$

Algorithm 5: [15] compensated Clenshaw algorithm to evaluate Chebyshev series accurately.

function DDClenshaw(ph, pl, x)
$b = zeros(2, N + 2)$
for $j = N : -1 : 2$ **do**
$[rh1, rl1] =$ prod_dd_d$(b(1, j + 1), b(2, j + 1), 2 * x)$
$[rh2, rl2] =$ add_dd_dd$(rh1, rl1, -b(1, j + 2), -b(2, j + 2))$
$[b(1, j), b(2, j)] =$ add_dd_dd(rh2, rl2, ph(j), pl(j))
end for
$[rh1, rl1] =$ prod_dd_d$(b(1, 2), b(2, 2), 2 * x)$
$[rh2, rl2] =$ add_dd_dd$(rh1, rl1, -b(1, 3), -b(2, 3))$
$[b(1, j), b(2, j)] =$ add_dd_dd(rh2, rl2, ph(1), pl(1))
$[rh, rl] = [b(1, j), b(2, j)]$

Algorithm 6: [15] Clenshaw algorithm in double-double format.

function $[rh, rl] =$ DDCTP(P, x, y)
for $i = 1 : m$ **do**
$[f_1(i), f_2(i)] =$ DDClenshaw$(A(i, :), a_0, y)$% here $a_0 = zeros(n + 1, 1)$
end for
$[rh, rl] =$ DDClenshaw(f_1, f_2, x)

Algorithm 7: Clenshaw Chebyshev tensor product in double-double format.

In order to analyze the error bound of CompCTP algorithm, we need a lemma (Lemma 5). Firstly, we review a lemma in [15].

Lemma 4 (see [15]). *Given* $p(x) = \sum_{j=0}^{n} c_j T_j(x)$ *and* π_j, σ_j, β_j, $j = n - 1 : -1 : 0$, *is the round-off error of EFTs in* CompClenshaw *(Algorithm 5), one obtains*

$$\sum_{j=0}^{n-1} \left(\left|\pi_j\right| + \left|\sigma_j\right| + \left|\beta_j\right|\right) \widetilde{T}_j(|x|) \leq 3nu\left(1 + \gamma_{3n}\right) \sum_{j=0}^{n} \left|c_j\right| \widetilde{T}_j(|x|). \tag{12}$$

Lemma 5. *Let* εb_0 *and* $\varepsilon \widehat{b}_0$ *be the error of theoretical and numerical error of* CompClenshaw, *respectively. One can get*

$$\left|\varepsilon b_0 - \varepsilon \widehat{b}_0\right| \leq \gamma_{3n-1}\gamma_{3n+1} \sum_{j=0}^{n} \left|c_j\right| \widetilde{T}_j(|x|). \tag{13}$$

Proof. Obviously, $\varepsilon \widehat{b}_0$ is the Clenshaw algorithm with coefficient $\widehat{\omega}_j$ in CompClenshaw (Algorithm 5). Using (4) we obtain

$$\begin{aligned}\left|\varepsilon b_0 - \varepsilon \widehat{b}_0\right| &\leq \gamma_{3n-1} \sum_{j=0}^{n} \left|\widehat{\omega}_j\right| \widetilde{T}_j(|x|) \\ &= \gamma_{3n-1} \sum_{j=0}^{n} \left|\pi_j \oplus \sigma_j \oplus \beta_j\right| \widetilde{T}_j(|x|) \\ &\leq \gamma_{3n-1} \sum_{j=0}^{n} \left(\left|\pi_j\right| + \left|\sigma_j\right| + \left|\beta_j\right|\right)\left(1 + \gamma_2\right) \widetilde{T}_j(|x|) \\ &\leq \gamma_{3n-1}\left(1 + \gamma_2\right) \sum_{j=0}^{n} \left(\left|\pi_j\right| + \left|\sigma_j\right| + \left|\beta_j\right|\right) \widetilde{T}_j(|x|).\end{aligned} \tag{14}$$

Using Lemma 4 to (14), we have

$$\left|\varepsilon b_0 - \varepsilon \widehat{b}_0\right| \leq 3nu\left(1 + \gamma_{3n-1}\right)\gamma_{3n-1}\left(1 + \gamma_2\right) \sum_{j=0}^{n} \left|c_j\right| \widetilde{T}_j(|x|). \tag{15}$$

We obtain $3nu\gamma_{3n-1}(1+\gamma_{3n-1})(1+\gamma_2) \leq \gamma_{3n-1} 3nu(1+\gamma_{3n+1}) \leq \gamma_{3n-1}\gamma_{3n+1}$. Then we deduct relation (13). □

Finally, we show the forward error bound of CompCTP algorithm using previous analysis.

Theorem 6. *Let* $P(x, y) = \sum_{i=0}^{m} \sum_{j=0}^{n} a_{ij} T_i(x) T_j(y)$ *be a Chebyshev tensor product surface. The* CompCTP *algorithm satisfies*

$$\begin{aligned}\left|\text{CompCTP}\left(P, x, y\right) - P\left(x, y\right)\right| &\leq u\left|P\left(x, y\right)\right| \\ &\quad + 3\left(\gamma_{3m+1}^2 + \gamma_{3n+1}^2\right) \sum_{i=0}^{m} \sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_i(|x|) \widetilde{T}_j(|y|).\end{aligned} \tag{16}$$

Proof. According to Algorithm 2, we get the numerical result of CompCTP algorithm as

$$\overline{P}\left(x, y\right) = \widehat{P}\left(x, y\right) \oplus \widehat{e} = \left(\widehat{P}\left(x, y\right) + \widehat{e}\right)\left(1 + \delta\right), \quad \delta \leq u. \tag{17}$$

Using relation (9) and

$$P(x,y) = \widehat{P}(x,y) + e,$$
$$e = e_2 + e_3,\ e_3 = \sum_{i=0}^{m} e_1(i)\, T_i(x), \tag{18}$$

we obtain

$$\overline{P}(x,y) = P(x,y)(1+\delta) + (\widehat{e} - e)(1+\delta). \tag{19}$$

Next, we consider the bound of $|\widehat{e} - e|$. For $\widehat{e} = \widehat{e}_2 \oplus \widehat{e}_3 = (\widehat{e}_2 + \widehat{e}_3)(1+\delta)$, $|\delta| < u$, and $e = e_2 + e_3 = (e_2 + e_3)(1+\delta) - \delta(e_2 + e_3)$, we get

$$|\widehat{e} - e| \le u\left|e_2 + e_3\right| + (1+u)\left(\left|e_2 - \widehat{e}_2\right| + \left|e_3 - \widehat{e}_3\right|\right). \tag{20}$$

Then, according to $|e| = |e_2 + e_3| = |P(x,y) - \widehat{P}(x,y)|$, we have

$$\left|e_2 + e_3\right| \le \gamma_{3(m+n)} \sum_{i=0}^{m}\sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_i(|x|)\, \widetilde{T}_j(|y|). \tag{21}$$

Let us consider $|e_2 - \widehat{e}_2|$. Because $\widehat{e}_2$ is the error from `CompClenshaw` algorithm, according to Lemma 5, we obtain

$$\left|e_2 - \widehat{e}_2\right| \le \gamma_{3m-1}\gamma_{3m+1} \sum_{i=0}^{m} |\widehat{\alpha}(i)|\, \widetilde{T}_i(|x|). \tag{22}$$

Using the relation

$$|\widehat{\alpha}(i)| \le (1 + \gamma_{3n-1}) \sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_j(|y|), \quad 0 \le i \le m \tag{23}$$

and (22), we have

$$\begin{aligned}&\left|e_2 - \widehat{e}_2\right| \\ &\le \gamma_{3m-1}\gamma_{3m+1}(1+\gamma_{3n-1}) \sum_{i=0}^{m}\sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_i(|x|)\, \widetilde{T}_j(|y|).\end{aligned} \tag{24}$$

Finally, we focus on the bound of $|e_3 - \widehat{e}_3|$. We assume $e_{3\mathrm{mid}} = \sum_{i=0}^{m} \widehat{e}_1(i) T_i(x)$, using triangle inequality

$$\left|e_3 - \widehat{e}_3\right| \le \left|e_3 - e_{3\mathrm{mid}}\right| + \left|e_{3\mathrm{mid}} - \widehat{e}_3\right|; \tag{25}$$

then apparently we have

$$\begin{aligned}\left|e_3 - e_{3\mathrm{mid}}\right| &= \sum_{i=0}^{m} \left|e_1(i) - \widehat{e}_1(i)\right| \widetilde{T}_i(|x|) \\ &\le \gamma_{3n-1}\gamma_{3n+1} \sum_{i=0}^{m}\sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_i(|x|)\, \widetilde{T}_j(|y|).\end{aligned} \tag{26}$$

Actually

$$\left|e_{3\mathrm{mid}} - \widehat{e}_3\right| \le \gamma_{3m-1} \sum_{i=0}^{m} \left|\widehat{e}_1(i)\right| \widetilde{T}_i(|x|); \tag{27}$$

then

$$\left|\widehat{e}_1(i)\right| \le \left|e_1(i)\right| + \left|e_1(i) - \widehat{e}_1(i)\right|, \quad 0 \le i \le m, \tag{28}$$

where $e_1(i)$ acts as the theoretical error of $\sum_{j=0}^{n} a_{ij} T_j(y)$ at ith step; combining Theorem 1 we get

$$\left|e_1(i)\right| \le \gamma_{3n-1} \sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_j(|y|). \tag{29}$$

Combining (13), (28), and (29) we obtain

$$\begin{aligned}\left|\widehat{e}_1(i)\right| &\le \gamma_{3n-1} \sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_j(y) \\ &\quad + \gamma_{3n-1}\gamma_{3n+1} \sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_j(|y|).\end{aligned} \tag{30}$$

Synthetically, combining (27) and (30) we derive

$$\begin{aligned}\left|e_{3\mathrm{mid}} - \widehat{e}_3\right| &\le \gamma_{3m-1} \sum_{i=0}^{m} \left|\widehat{e}_1(i)\right| \widetilde{T}_i(|x|) \\ &\le \gamma_{3m-1} \sum_{i=0}^{m} \left(\left|e_1(i)\right| + \left|e_1(i) - \widehat{e}_1(i)\right|\right) \widetilde{T}_i(|x|) \\ &\le \gamma_{3m-1}\left(\gamma_{3n-1} + \gamma_{3n-1}\gamma_{3n+1}\right) \\ &\quad \cdot \sum_{i=0}^{m}\sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_i(|x|)\, \widetilde{T}_j(|y|).\end{aligned} \tag{31}$$

According to (25), (26), and (31), we obtain

$$\begin{aligned}\left|e_3 - \widehat{e}_3\right| &\le \left(\gamma_{3n-1}\gamma_{3n+1} + \gamma_{3m-1}\left(\gamma_{3n-1} + \gamma_{3n-1}\gamma_{3n+1}\right)\right) \\ &\quad \cdot \sum_{i=0}^{m}\sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_i(x)\, \widetilde{T}_j(y),\end{aligned} \tag{32}$$

using (20), (21), (24), and (32), the error bound yields

$$|e - \widehat{e}| \le \alpha(m,n) \sum_{i=0}^{m}\sum_{j=0}^{n} \left|a_{ij}\right| \widetilde{T}_i(|x|)\, \widetilde{T}_j(|y|), \tag{33}$$

where $\alpha(m,n)$ is

$$\begin{aligned}\alpha(m,n) &= u\gamma_{3(m+n)} + (1+u)\left[\gamma_{3m-1}\gamma_{3m+1}(1+\gamma_{3n-1})\right. \\ &\quad \left. + \gamma_{3n-1}\gamma_{3n+1}(1+\gamma_{3m-1}) + \gamma_{3m-1}\gamma_{3n-1}\right].\end{aligned} \tag{34}$$

By the properties of the element γ_n [18], we deduct some inequations as follows:

$$\begin{aligned}&(1+u)\, u\gamma_{3(m+n)} \le \frac{1}{2}\left(\gamma_{3m+1}^2 + \gamma_{3n+1}^2\right); \\ &(1+u)^2(1+\gamma_{3m-1})\gamma_{3n-1}\gamma_{3n+1} \le 2(1+u)^2 \gamma_{3n-1}\gamma_{3n+1} \\ &\quad \le 2\gamma_{3n+1}^2; \\ &(1+u)^2 \gamma_{3n-1}\gamma_{3m-1} \le \gamma_{3m}\gamma_{3n} \le \frac{1}{2}\left(\gamma_{3m+1}^2 + \gamma_{3n+1}^2\right); \\ &(1+u)^2(1+\gamma_{3n-1})\gamma_{3m+1}\gamma_{3m-1} \le 2\gamma_{3m+1}^2.\end{aligned} \tag{35}$$

So we have $(1+u)\alpha(m,n) \le 3(\gamma_{3m+1}^2 + \gamma_{3n+1}^2)$ and combine (19) with (33); then we obtain relation (16). □

Let $P(x,y) = \sum_{i=0}^{m}\sum_{j=0}^{n} a_{ij}T_i(x)T_j(y)$ be a polynomial in Chebyshev tensor product. If the condition number for polynomial evaluation of the $P(x,y)$ at entry (x,y) is defined by

$$\begin{aligned}\mathtt{cond}(P,x,y) &= \frac{\widetilde{P}(|x|,|y|)}{|P(x,y)|} \\ &= \frac{\sum_{i=0}^{m}\sum_{j=0}^{n}|a_{ij}|\,\widetilde{T}_i(|x|)\,\widetilde{T}_j(|y|)}{\left|\sum_{i=0}^{m}\sum_{j=0}^{n}a_{ij}T_i(x)\,T_j(y)\right|},\end{aligned} \tag{36}$$

we show the relative error bound of the CTP and CompCTP algorithm in Corollary 7.

Corollary 7. *Let $P(x,y) = \sum_{i=0}^{m}\sum_{j=0}^{n} a_{ij}T_i(x)T_j(y)$ be a polynomial in Chebyshev tensor product. The relative error bound of the* CTP *algorithm and* CompCTP *algorithm yields*

$$\begin{aligned}&\frac{|\mathtt{CTP}(P,x,y) - P(x,y)|}{|P(x,y)|} \le \gamma_{3(m+n)-2}\,\mathtt{cond}(P,x,y), \\ &\frac{|\mathtt{CompCTP}(P,x,y) - P(x,y)|}{|P(x,y)|} \\ &\quad\le u + 3\left(\gamma_{3m+1}^2 + \gamma_{3n+1}^2\right)\mathtt{cond}(P,x,y).\end{aligned} \tag{37}$$

As Corollary 7 illustrates that if $3(\gamma_{3m+1}^2 + \gamma_{3n+1}^2)\mathtt{cond}(P,x,y) < u$, the error of the result evaluated by CompCTP is bounded by working precision u. Besides that, the computed result generated by the CompCTP scheme is as accurate as if evaluated in double-double precision.

5. Experimental Results

In this section, we perform a series of numerical experiments. The programs are written in Matlab code and run with the software of MATLAB R2015b. We consider the polynomials with floating-point numbers coefficients and floating-point element (x,y) expressed in Chebyshev tensor product. Considering the efficiency, we use CTP with quad-double arithmetic (QDCTP) [22] to accurately evaluate polynomial instead of symbolic toolbox.

Generally, we can get an accurate result using Algorithm 1 as the Chebyshev tensor product series is well-conditioned. But, we need a more accurate algorithm when the problem is ill-conditioned. We consider a bivariate polynomial in area $[0,1]\times[0,1]$ proposed by [14]

$$\begin{aligned}&P(x,y) \\ &\quad= (x-0.75)^3(x-0.2)^3(y-0.75)^3(y-0.2)^3\end{aligned} \tag{38}$$

and convert it to a Chebyshev tensor product form.

We extend the univariate conversion algorithm [23] to change a bivariate polynomial in power form to a Chebyshev tensor product form. We transform the coefficients via the Matlab symbolic toolbox. Since the coefficients of polynomials in Chebyshev tensor product which are evaluated by us are floating-point numbers, they need to be rounded to the nearest floating-point elements.

We use CTP, CompCTP, and QDCTP (CTP algorithm along with quad-double arithmetic) [22] to compute the value of Chebyshev tensor product polynomial $P(x,y)$ at 400 grid points near the multiple root $(0.75, 0.2)$. Figure 1 shows the surface generated by different algorithms. It is obvious that the compensated algorithm can approximate the expected smooth surface as accurate as that using CTP algorithm with quad-double arithmetic, when the results are rounded to the working precision. Observe that the surface of CTP is a folding interface and varies slightly in the direction x. The reason is that we firstly compute $\widehat{\alpha}(i) = \mathtt{Clenshaw}(P(i,:),y)$ at the ith step in the loop of the CTP algorithm and then compute $\widehat{\alpha}(0) = \mathtt{Clenshaw}(\widehat{\alpha},x)$ leading to the more obviously influences of the round-off errors along x. Thus, the CompCTP algorithm can compute a desired result.

Figure 2 performs an absolute forward error using the algorithms CTP and CompCTP for 400 points. It is clear that the CompCTP reduces the error better than CTP algorithm. Besides, we observe that the relative errors of CompCTP algorithm are smaller than the working precision u even near the point $(0.75, 0.2)$. Therefore, the experiments verify our estimation of relation (37).

Next we consider the relative error bounds for ill-conditioned polynomials. We produce a series of ill-conditioned polynomials in Chebyshev tensor product and evaluate them using CTP, CompCTP, and DDCTP (CTP with double-double precision in Algorithms 3–7). We choose degree $m \times n$ (where m,n is parameter from (6)) with condition numbers changing from 10^3 to 10^{36}. The algorithm generating the ill-conditioned polynomials is shown in Algorithms 3–7 (GenPolyCTP), which is similar to algorithm GenPoly in [24]. Considering the size of the problem and computational efficiency, we choose $m \times n = 6 \times 7$. We plot the results in Figure 3. Obviously, we observe that the CompCTP illustrates an expected result, where the CompCTP is more stable and accurate than CTP. The relative errors are equal to or smaller than u when $\mathtt{cond}(P,x,y) \le 1/u$. While $\mathtt{cond}(P,x,y) \in [1/u, 1/u^2]$, the relative errors increase almost linearly. Meanwhile, we notice that the CompCTP shows high accuracy because the rounding errors are recorded to approximate the real errors even under ill-condition. Besides, we also compute the ill-conditioned polynomials using DDCTP based on the Bailey's *quad-double* [2, 25] arithmetic. Comparing with DDCTP, we can find that the results of CompCTP algorithm are as accurate as those computed using double-double arithmetic, which are illustrated by our numerical experiment.

Finally, we focus on the computational complexity of all the algorithms.

Clenshaw: $3n + 4$

CompClenshaw: $36n + 38$

DDClenshaw: $52n + 53$

CTP: $(3n + 4)(m + 1) + 3m + 4$

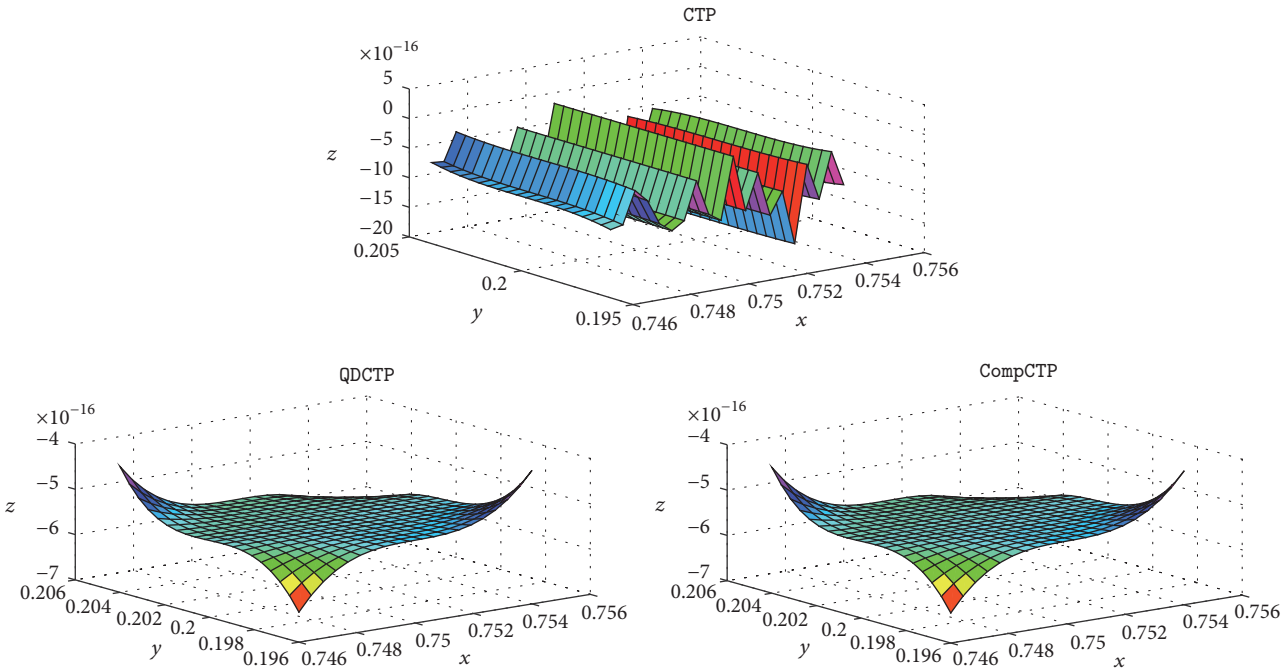

Figure 1: The surface using CTP, QDCTP, and CompCTP algorithm to evaluate Chebyshev tensor product series.

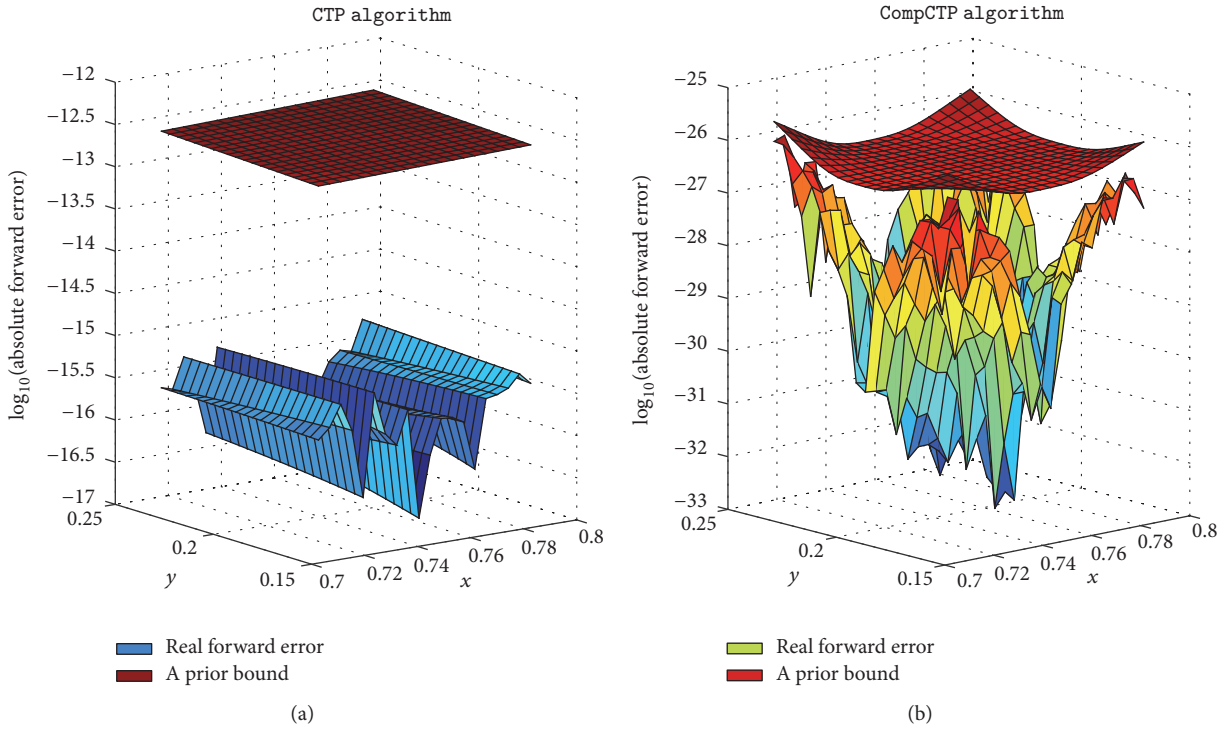

Figure 2: Accuracy of the absolute error of CTP algorithm (a) and CompCTP algorithm (b).

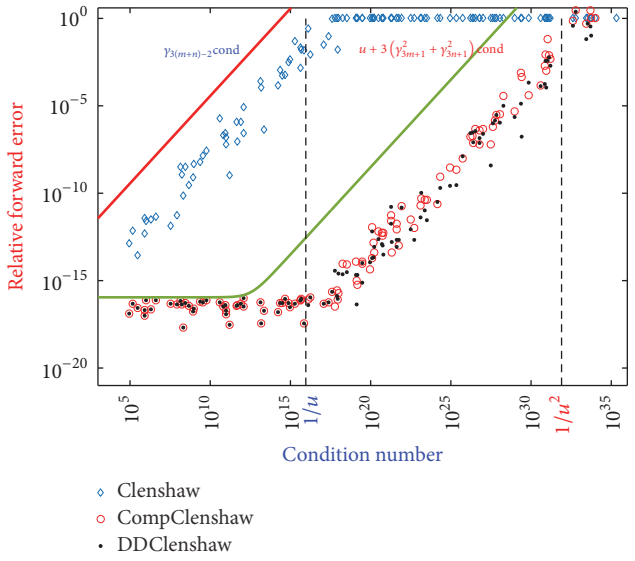

FIGURE 3: The relative forward error by using CTP, CompCTP, and DDCTP to evaluate ill-conditioned polynomials.

$$\text{CompCTP: } (36n + 38)(m + 1) + 39m + 43$$
$$\text{DDCTP: } (52n + 53)(m + 1) + (52m + 53)$$

Considering the previous comparisons of the accuracy, we can confirm that CompCTP algorithm is as accurate as computation with DDCTP algorithm (shown in Algorithms 3–7). However, CompCTP only requires on the average about 69.2% of flops. So our algorithm is more efficient than CTP with double-double arithmetic.

6. Conclusion

We present an accurate and efficient algorithm for evaluation of Chebyshev tensor product surface, which is based on the Clenshaw algorithm and error-free transformations. The error analysis shows that CompCTP algorithm can get the same accuracy as that computed by the traditional CTP algorithm with twice working precision. Besides, this compensated algorithm can run more efficiently than CTP algorithm with double-double precision. Experiments illustrate that our algorithm is stable and accurate even in some ill-condition cases.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

Hao Jiang is partially supported by the National Natural Science Foundation of China (no. 61402495, no. 61602166, no. 61303189, and no. 61402496). Chongwen Duan is partially supported by the National Natural Science Foundation of China (no. 61401515). Hongxia Wang is partially supported by the National Natural Science Foundation of China (no. 61571008). Lizhi Cheng is partially supported by Science Project of National University of Defense Technology (JC120201) and National Natural Science Foundation of Hunan Province in China (13JJ2001).

References

[1] L. Fox and I. Parker, *Chebyshev Polynomials in Numerical Analysis*, Oxford University Press, 1968.

[2] D. H. Bailey, R. Barrio, and J. M. Borwein, "High-precision computation: mathematical physics and dynamics," *Applied Mathematics and Computation*, vol. 218, no. 20, pp. 10106–10121, 2012.

[3] T. J. Rivlin and J. Theodore, "The chebyshev polynomials," vol. 30, no. 134, p. 374, 1974.

[4] R. Barrio and A. Elipe, "Integration of orbital motions with chebyshev polynomial," in *Dynamics and Astrometry of Natural and Artificial Celestial Bodies*, pp. 419–424, Springer, 1997.

[5] R. Mukundan, S. H. Ong, and P. A. Lee, "Image analysis by Tchebichef moments," *IEEE Transactions on Image Processing*, vol. 10, no. 9, pp. 1357–1364, 2001.

[6] H. Rahmalan, N. A. Abu, and S. L. Wong, "Using tchebichef moment for fast and efficient image compression," *Pattern Recognition and Image Analysis*, vol. 20, no. 4, pp. 505–512, 2010.

[7] Z. Omar, N. Mitianoudis, and T. Stathaki, "Two-dimensional chebyshev polynomials for image fusion," in *Proceedings of the 28th Picture Coding Symposium (PCS '10)*, pp. 426–429, Japan, December 2010.

[8] N. A. Abu, N. Suryana, and R. Mukundan, "Perfect image reconstruction using discrete orthogonal moments," in *Proceedings of the Fourth IASTED International Conference on Visualization, Imaging, and Image Processing (VIIP '04)*, pp. 903–907, September 2004.

[9] N. A. Abu and M. R. K. Ariffin, "A novel psychovisual model on an independent video frame for an almost lossless compression," in *Proceedings of the 2014 10th International Conference on Information Assurance and Security (IAS '14)*, pp. 60–65, Japan, November 2014.

[10] W. Guobao and W. Shigang, "Recursive computation of Tchebichef moment and its inverse transform," *Pattern Recognition*, vol. 39, no. 1, pp. 47–56, 2006.

[11] X. S. Li, J. W. Demmel, D. H. Bailey et al., "Design, implementation and testing of extended and mixed precision blas," *ACM Transactions on Mathematical Software*, vol. 28, no. 2, pp. 152–205, 2002.

[12] T. Ogita, S. M. Rump, and S. Oishi, "Accurate sum and dot product," *SIAM Journal on Scientific Computing*, vol. 26, no. 6, pp. 1955–1988, 2005.

[13] S. Graillat, P. Langlois, and N. Louvet, "Compensated horner scheme," in *Proceedings of the In Algebraic and Numerical Algorithms and Computer-Assisted Proofs*, B. Buchberger, S. Oishi, M. Plum, and S. M. Rump, Eds., p. 05391, 2005.

[14] H. Jiang, S. Li, L. Cheng, and F. Su, "Accurate evaluation of a polynomial and its derivative in Bernstein form," *Computers & Mathematics with Applications*, vol. 60, no. 3, pp. 744–755, 2010.

[15] H. Jiang, R. Barrio, H. Li, X. Liao, L. Cheng, and F. Su, "Accurate evaluation of a polynomial in Chebyshev form," *Applied Mathematics and Computation*, vol. 217, no. 23, pp. 9702–9716, 2011.

[16] C. W. Clenshaw, "A note on the summation of chebyshev series," *Mathematics of Computation*, vol. 9, no. 51, pp. 118–120, 1955.

[17] IEEE Standards Committee and others, 754-2008 IEEE standard for floating-point arithmetic. IEEE Computer Society Std, 2008.

[18] N. J. Higham, *Accuracy and stability of numerical algorithms*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, USA, 2002.

[19] R. Barrio, H. Jiang, and S. Serrano, "A general condition number for polynomials," *SIAM Journal on Numerical Analysis*, vol. 51, no. 2, pp. 1280–1294, 2013.

[20] E. K. Donald et al., "The art of computer programming," *Sorting and Searching*, vol. 3, pp. 426–458, 1999.

[21] T. J. Dekker, "A floating-point technique for extending the available precision," *Numerische Mathematik*, vol. 18, pp. 224–242, 1971/72.

[22] Y. Hida, X. S. Li, and D. H. Bailey, "Algorithms for quad-double precision floating point arithmetic," in *Proceedings of the 15th IEEE Symposium on Computer Arithmetic*, pp. 155–162, Vail, Colo, USA, June 2001.

[23] P. Langlois and N. Louvet, "More instruction level parallelism explains the actual efficiency of compensated algorithms," Tech. Rep., DALI Research Team (2007), http://hal.archives-ouvertes.fr/hal-00165020.

[24] N. Louvet, *Compensated algorithms in floating point arithmetic: accuracy, validation, performances [Ph.D. thesis]*, Université de Perpignan Via Domitia, 2007.

[25] D. H. Bailey, *High-precision software directory: QD library, double-double library*, Berkeley National Laboratory, 2008, http://www.nersc.gov/dhbailey/mpdist/mpdist.html.