 CMC, vol.25, no.3, pp.239-263, 2011

A Highly Accurate Multi-Scale Full/Half-Order Polynomial Interpolation

Chein-Shan Liu[1]

Abstract: For the computational applications in several areas, we propose a single-scale and a multi-scale diagonal preconditioners to reduce the condition number of Vandermonde matrix. Then a new algorithm is given to solve the inversion of the resulting coefficient matrix after multiplying by a preconditioner to the Vandermonde matrix. We apply the new techniques to the interpolation of data by using very high-order polynomials, where the Runge phenomenon disappears even the equidistant nodes are used. In addition, we derive a new technique by employing an m-order polynomial with a multi-scale technique to interpolate $2m+1$ data. Numerical results confirm the validity of present polynomial interpolation method, where only a constant parameter R_0 needs to be specified in the multi-scale expansion. For the Differential Quadrature (DQ), the present method provides a very accurate numerical differential. Then, by a combination of this DQ and the Fictitious Time Integration Method (FTIM), we can solve nonlinear boundary value problems effectively.

Keywords: Multi-scale polynomial interpolation, Half-order polynomial interpolation, Differential quadratures, Vandermonde matrices

1 Introduction

The Vandermonde matrices arise in a variety of mathematical applications in polynomial interpolation, numerical differentiation, approximation of linear functional, rational Chebyshev approximation, and differential quadrature (DQ). In these applications, finding the solution of a linear system with the Vandermonde matrix as a coefficient matrix, and the inversion of the Vandermonde matrix are required. So an efficient method to find the inversions of Vandermonde matrices is desirable.

[1] Department of Civil Engineering, National Taiwan University, Taipei 106-17, Taiwan, E-mail: liucs@ntu.edu.tw

The condition number of Vandermonde matrix may be quite large [Gautschi (1975)], causing a large error when one computes the inversion of Vandermonde matrix appeared in a large scale linear system. Vandermonde matrices are notoriously ill-conditioned. Previously, Liu, Hong and Atluri (2010) have developed a natural regularization method to solve those ill-posed linear problems for the Vandermonde system. Beckermann (2000) and Li (2006) gave what the optimal condition number of Vandermonde matrix could be expected. Several authors have therefore proposed algorithms which exploit the structure of Vandermonde matrix in operations to compute numerically stable solutions, instead of that required by the Gaussian elimination [Higham (1987); Higham (1988); Björck and Pereyra (1970); Calvetti and Reichel (1993)]. These methods mainly rely on constructing first a Newton interpolation of the polynomial and then converting it to the monomial form.

Wertz (1965) has suggested a simple numerical procedure, which can greatly facilitate the computation of the inverse of Vandermonde matrix. Neagoe (1996) has found an analytic formula to calculate the inverse of Vandermonde matrix. However, a direct application of Neagoe's formula will result in a tedious algorithm with $O(n^3)$ flops. Some discussions about the numerical algorithms for the inversions of Vandermonde matrices are summarized by Gohberg and Olshevsky (1997).

Since the works pioneered by Bellman and Casti (1971), and Bellman, Kashef and Casti (1972), Differential Quadrature (DQ) has been developed by many researchers. However, due to its ill-conditioned property, this method is limited to the small scale engineering problems. Shu (2000) has developed a systematic method to compute the weighting coefficients, under the analysis of a high-order polynomial approximation and the analysis in a linear vector space. Recently, Shen and Liu (2011) have employed a least-square method to determine the weighting coefficients, such that the burden in solving the ill-conditioned Vandermonde problem can be alleviated.

Interpolation is a process of using known data values to estimate unknown data values. Various interpolation techniques have been used in engineering sciences. It is by now a commonplace observation that the needs of automatic digital computation have spurred an enormous revival of interest in methods of interpolating data or continuous functions by functions which depend only on a finite number of parameters. Apart from its applications approximation theory is a lively branch of mathematical analysis. Although the Weierstrass theorem guarantees that a continuous function defined on a finite interval can be approximated uniformly within any preassigned error bound by polynomials, in a practical numerical computation there appeared a counterexample due to Runge. Of course, the Runge phenomenon is a computational problem and is not a mathematical problem, causing by the use of uniform nodes in the interpolation. If right nodes are picked, the Runge's phe-

nomenon cannot occur. Unfortunately, for most cases in application we have to interpolate the data with an equidistant sample.

Indeed, the polynomial interpolation is an ill-posed problem and it makes the interpolation by higher-order polynomials not being easy to numerical implementation. In order to overcome these difficulties, Liu and Atluri (2009a) have introduced a characteristic length into the high-order polynomials expansion, which improved the numerical accuracy for the applications to solve some ill-posed linear problems. At the same time, Liu, Yeih and Atluri (2009) have developed a multi-scale Trefftz-collocation Laplacian conditioner to deal with the ill-conditioned linear systems. This concept of multi-scale Trefftz-collocation method has been later employed by Chen, Yeih, Liu and Chang (2012) to solve the sloshing wave problem. In this paper we extend the work by Liu and Atluri (2009a), and propose a new multi-scale interpolation technique by high-order polynomials of full and half orders, which can overcome the above-mentioned ill-conditioned behavior. This paper is organized as follows. The new preconditioners are introduced in Section 2, where a single characteristic length and a multi-scale characteristic length are developed. In Section 3, based-on the single characteristic length we develop a fast algorithm to find the inversion of Vandemonde matrix. Some numerical tests and applications to polynomial interpolations and differential quadratures are reported in Section 4. Section 5 is a major contribution by introducing a multi-scale and half-order polynomial interpolation technique, where we use m-order polynomial to interpolate $2m+1$ data. Finally, the conclusions are drawn in Section 6.

2 New preconditioning techniques for the Vandermonde matrices

Polynomial interpolation as its name is the interpolation of a given set of data by a polynomial. In other words, given some data points as available by sampling a measurement, the aim is to find a polynomial which goes exactly through these points of data.

Given a set of n data points (x_i, y_i) where no two x_i are the same, one is looking for a polynomial $p(x)$ of order at most $n-1$ with the following property:

$$p(x_i) = y_i, \quad i = 1, \ldots, n, \tag{1}$$

where $x_i \in [a,b]$, and $[a,b]$ is a spatial interval of our problem domain.

The unisolvence theorem states that such a polynomial $p(x)$ exists and is unique, and can be proved by using the Vandermonde matrix:

$$p(x) = \sum_{i=1}^{n} a_i x^{i-1}, \tag{2}$$

where $\{x^i,\ i=0,1,\ldots\}$ constitute a monomial basis.

2.1 Transforming the Vandermonde matrix into another one

The statement that $p(x)$ interpolates the data points means that Eq. (1) must hold. If we substitute Eq. (2) into Eq. (1), we can obtain a system of linear equations to determine the coefficients a_i, which in a matrix-vector form reads as

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^{n-2} & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \ldots & x_2^{n-2} & x_2^{n-1} \\ \vdots & \vdots & \vdots & \ldots & \vdots & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \ldots & x_{n-1}^{n-2} & x_{n-1}^{n-1} \\ 1 & x_n & x_n^2 & \ldots & x_n^{n-2} & x_n^{n-1} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_{n-1} \\ a_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix}. \tag{3}$$

We have to solve the above system for a_i to construct the interpolant $p(x)$ expressed in Eq. (2). The transpose of the coefficient matrix on the left-side is commonly referred to as a Vandermonde matrix, denoted by $\mathbf{V}$:

$$\mathbf{V} = \begin{bmatrix} 1 & 1 & \ldots & 1 & 1 \\ x_1 & x_2 & \ldots & x_{n-1} & x_n \\ x_1^2 & x_2^2 & \ldots & x_{n-1}^2 & x_n^2 \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ x_1^{n-2} & x_2^{n-2} & \ldots & x_{n-1}^{n-2} & x_n^{n-2} \\ x_1^{n-1} & x_2^{n-1} & \ldots & x_{n-1}^{n-1} & x_n^{n-1} \end{bmatrix}, \tag{4}$$

where the real numbers $\{x_1,\ldots,x_n\}$ are called nodes. The determinant of $\mathbf{V}$ is nonzero, which proves the unisolvence theorem: there exists a unique interpolating polynomial in Eq. (2) to interpolate n data.

Although the above theorem is theoretically feasible, in a practical computation by solving Eq. (3) for the polynomial interpolation the problem is ill-posed and it makes the interpolation by using higher-order polynomials not being easy to numerical implementation. Our strategy to solve this ill-conditioned problem of the inversion of the Vandermonde matrix $\mathbf{V}$ is achieved by considering a preconditioning matrix:

$$\mathbf{R} = \begin{bmatrix} 1 & 0 & \ldots & 0 & 0 \\ 0 & \frac{1}{R_0} & \ldots & 0 & 0 \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ 0 & 0 & \ldots & \frac{1}{R_0^{n-2}} & 0 \\ 0 & 0 & \ldots & 0 & \frac{1}{R_0^{n-1}} \end{bmatrix}, \tag{5}$$

where R_0 is a characteristic length of the problem domain with a requirement that all the nodes $x_i \in [-R_0, R_0],\ i = 1, \ldots, n$.

Let

$$\mathbf{A} = \mathbf{RV} = \begin{bmatrix} 1 & 1 & \cdots & 1 & 1 \\ \frac{x_1}{R_0} & \frac{x_2}{R_0} & \cdots & \frac{x_{n-1}}{R_0} & \frac{x_n}{R_0} \\ \left(\frac{x_1}{R_0}\right)^2 & \left(\frac{x_2}{R_0}\right)^2 & \cdots & \left(\frac{x_{n-1}}{R_0}\right)^2 & \left(\frac{x_n}{R_0}\right)^2 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \left(\frac{x_1}{R_0}\right)^{n-2} & \left(\frac{x_2}{R_0}\right)^{n-2} & \cdots & \left(\frac{x_{n-1}}{R_0}\right)^{n-2} & \left(\frac{x_n}{R_0}\right)^{n-2} \\ \left(\frac{x_1}{R_0}\right)^{n-1} & \left(\frac{x_2}{R_0}\right)^{n-1} & \cdots & \left(\frac{x_{n-1}}{R_0}\right)^{n-1} & \left(\frac{x_n}{R_0}\right)^{n-1} \end{bmatrix}. \tag{6}$$

Hence, from Eqs. (4) and (6) we can find the inverse of $\mathbf{V}$ by

$$\mathbf{V}^{-1} = \mathbf{A}^{-1}\mathbf{R}. \tag{7}$$

In order to overcome the difficulties appeared in the conventional collocation Trefftz method to solve the Laplace equation, Liu (2007a, 2007b, 2007c, 2008a), Chen, Liu and Chang (2009), and Chen, Liu, Chang and Chang (2010) have proposed a modified Trefftz method, refined this method by taking the characteristic length into the T-complete functions, such that the condition number of the resulting linear equations system can be greatly reduced. The same idea is employed here to reduce the ill-condition of the original Vandermonde matrix by including a characteristic length R_0 into the matrix equation.

2.2 Transforming the Vandermonde matrix into a non-Vandermonde matrix

The previous technique may be not effective for some Vandermonde matrices. In order to derive another well-conditioned matrix from the Vandermonde matrix in Eq. (4), let us consider a new set of coefficients:

$$\bar{a}_1 = a_1,\ \bar{a}_i = R_{i-1}^{i-1} a_i,\ i = 2, \ldots, n, \tag{8}$$

where $R_i,\ i = 1, \ldots, n-1$ is a sequence of numbers, and R_{i-1}^{i-1} is the $(i-1)$th power of R_{i-1}.

Inserting Eq. (8) for a_i into Eq. (2) we can obtain a new polynomial interpolant:

$$p(x) = \bar{a}_1 + \sum_{i=2}^{n} \bar{a}_i \left(\frac{x}{R_{i-1}}\right)^{i-1}. \tag{9}$$

If we substitute Eq. (9) into Eq. (1), we have a new system of linear equations in the coefficients $\bar{a}_i$:

$$\begin{bmatrix} 1 & \frac{x_1}{R_1} & \left(\frac{x_1}{R_2}\right)^2 & \cdots & \left(\frac{x_1}{R_{n-2}}\right)^{n-2} & \left(\frac{x_1}{R_{n-1}}\right)^{n-1} \\ 1 & \frac{x_2}{R_1} & \left(\frac{x_2}{R_2}\right)^2 & \cdots & \left(\frac{x_2}{R_{n-2}}\right)^{n-2} & \left(\frac{x_2}{R_{n-1}}\right)^{n-1} \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ 1 & \frac{x_{n-1}}{R_1} & \left(\frac{x_{n-1}}{R_2}\right)^2 & \cdots & \left(\frac{x_{n-1}}{R_{n-2}}\right)^{n-2} & \left(\frac{x_{n-1}}{R_{n-1}}\right)^{n-1} \\ 1 & \frac{x_n}{R_1} & \left(\frac{x_n}{R_2}\right)^2 & \cdots & \left(\frac{x_n}{R_{n-2}}\right)^{n-2} & \left(\frac{x_n}{R_{n-1}}\right)^{n-1} \end{bmatrix} \begin{bmatrix} \bar{a}_1 \\ \bar{a}_2 \\ \vdots \\ \bar{a}_{n-1} \\ \bar{a}_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix}. \tag{10}$$

Let $\mathbf{B}$ be the transpose of the above system matrix, i.e.,

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & \cdots & 1 & 1 \\ \frac{x_1}{R_1} & \frac{x_2}{R_1} & \cdots & \frac{x_{n-1}}{R_1} & \frac{x_n}{R_1} \\ \left(\frac{x_1}{R_2}\right)^2 & \left(\frac{x_2}{R_2}\right)^2 & \cdots & \left(\frac{x_{n-1}}{R_2}\right)^2 & \left(\frac{x_n}{R_2}\right)^2 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \left(\frac{x_1}{R_{n-2}}\right)^{n-2} & \left(\frac{x_2}{R_{n-2}}\right)^{n-2} & \cdots & \left(\frac{x_{n-1}}{R_{n-2}}\right)^{n-2} & \left(\frac{x_n}{R_{n-2}}\right)^{n-2} \\ \left(\frac{x_1}{R_{n-1}}\right)^{n-1} & \left(\frac{x_2}{R_{n-1}}\right)^{n-1} & \cdots & \left(\frac{x_{n-1}}{R_{n-1}}\right)^{n-1} & \left(\frac{x_n}{R_{n-1}}\right)^{n-1} \end{bmatrix}. \tag{11}$$

Upon defining a new diagonal matrix by

$$\mathbf{D} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & \frac{1}{R_1} & \cdots & 0 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & \cdots & \frac{1}{R_{n-2}^{n-2}} & 0 \\ 0 & 0 & \cdots & 0 & \frac{1}{R_{n-1}^{n-1}} \end{bmatrix}, \tag{12}$$

we have

$$\mathbf{B} = \mathbf{D}\mathbf{V}. \tag{13}$$

But $\mathbf{B}$ is not a Vandermonde matrix. However, by suitably choosing the numbers of R_i, $\mathbf{B}$ is a well-conditioned matrix, and we can apply the conjugate gradient method (CGM) to calculate the inverse of $\mathbf{B}$ [Liu, Hong and Atluri (2010)]. Hence, we have

$$\mathbf{V}^{-1} = \mathbf{B}^{-1}\mathbf{D}. \tag{14}$$

In Section 4.2 we will give a highly ill-conditioned matrix of **V** to test the performance for the reduction of the condition number by **B**.

The above problems are now searching a suitable diagonal matrix to reduce as much as possible the condition number of a real Vandermonde matrix **V** by a multi-scale row-scaling. Theoretically, there are theories of optimal scaling proposed by Bauer (1963) and van der Sluis (1969). A matrix is equilibrated if all its rows have the same norm. Generally speaking, choosing **D** such that **DV** is equilibrated minimizes the condition number of **V**. Usually, the optimally scaled Vandermonde matrices are still rather ill-conditioned. In Section 5 we will propose a better half-order polynomial interpolation and still with a multi-scaling technique.

3 A new algorithm

Instead of directly finding the inverse of **V** in Eq. (4), we develop a new algorithm to find the inverse of **A** in Eq. (6), and then applying Eq. (7) to find the inverse of **V**. Let $a_i = x_i/R_0,\ i = 1,\ldots,n$ be the new nodes, and define the master polynomial by

$$P_n(x) = \prod_{k=1}^{n}(x-a_k) = x^n + \sum_{k=1}^{n} A_k x^{n-k}, \tag{15}$$

whose zeros are the nodes of **A**. The coefficients A_k are functions of the roots a_k.

To express A_i in terms of a_i as suggested by Wertz (1965), we can obtain polynomials with increasing degree. For $k = 1$ we have

$$P_1(x) = x - a_1,\ \ A_1^{(1)} = -a_1,$$

where the superscript (k) of $A_m^{(k)},\ m = 1,\ldots,k$ indicates the degree of the source polynomial.

For $k = 2$ we have

$$P_2(x) = x^2 - (a_1+a_2)x + a_1a_2,\ \ A_1^{(2)} = A_1^{(1)} - a_2,\ \ A_2^{(2)} = -a_2A_1^{(1)}.$$

For $k = 3,4,\ldots,n$ it can be verified that

$$\begin{aligned} A_k^{(k)} &= -a_kA_{k-1}^{(k-1)}, \\ A_m^{(k)} &= A_m^{(k-1)} - a_kA_{m-1}^{(k-1)},\ \ m = k-1,k-2,\ldots,2, \\ A_1^{(k)} &= A_1^{(k-1)} - a_k. \end{aligned} \tag{16}$$

The (j,k)-element of $\mathbf{A}^{-1}$ is given by Neagoe (1996) as

$$A_{j,n-k}^{(n-1)}/D_j,\ \ D_j=\prod_{k=1}^{j-1}(a_j-a_k)\prod_{k=j+1}^{n}(a_j-a_k), \tag{17}$$

where $A_{j,n-k}^{(n-1)}$ can be calculated by

$$\begin{aligned} A_{j,1}^{(n-1)} &= A_1^n + a_j, \\ A_{j,m}^{(n-1)} &= A_m^{(n)} + a_j A_{j,m-1}^{(n-1)},\ \ m=2,3,\ldots,n-1. \end{aligned} \tag{18}$$

After all elements of $\mathbf{A}^{-1}$ are obtained, we can insert them into Eq. (7) to calculate $\mathbf{V}^{-1}$. The present algorithm requires only $O(n^2)$ flops.

4 Numerical tests

4.1 Comparing the condition numbers of V *and* A

We compare the condition numbers of the coefficient matrices in Eqs. (4) and (6) for different n and R_0. The condition number of a non-singular square matrix $\mathbf{A}$ is defined by

$$\text{Cond}(\mathbf{A}) = \|\mathbf{A}\|\|\mathbf{A}^{-1}\|. \tag{19}$$

The norm for $\mathbf{A}$ is the Frobenius norm.

The interval of $[a,b]=[0,4]$ is fixed, and the nodal points are given by $x_i=(i-1)\Delta x=(i-1)(b-a)/(n-1)$. Fixing $n=51$, we apply the CGM to find the inverse matrix under a convergence criterion 10^{-10}. The condition number is calculated by Eq. (19), which is plotted with respect to R_0 in Fig. 1(a). It can be seen that when R_0 is smaller than 3 and up to $R_0=1$ the condition number increases very fast to a very large value over 10^{31}, which is the condition number of the Vandermonde matrix in Eq. (4). Conversely, when $R_0>3$ the condition number tends to a stable value smaller than 10^9.

In Fig. 1(b) we plot the condition number with respect to n in the range of $30\le n\le 60$ for $R_0=1$ and $R_0=4$. It can be seen that without taking the characteristic length into account, the condition number of the Vandermonde matrix increases exponentially with respect to n [Gautschi and Inglese (1988); Skrzipek (2004)], and with a huge value up to 10^{38} when $n=60$. Conversely, when the characteristic length is taken to be $R_0=4$ in Eq. (6), the condition number can be controlled almost in the order of 10^9.

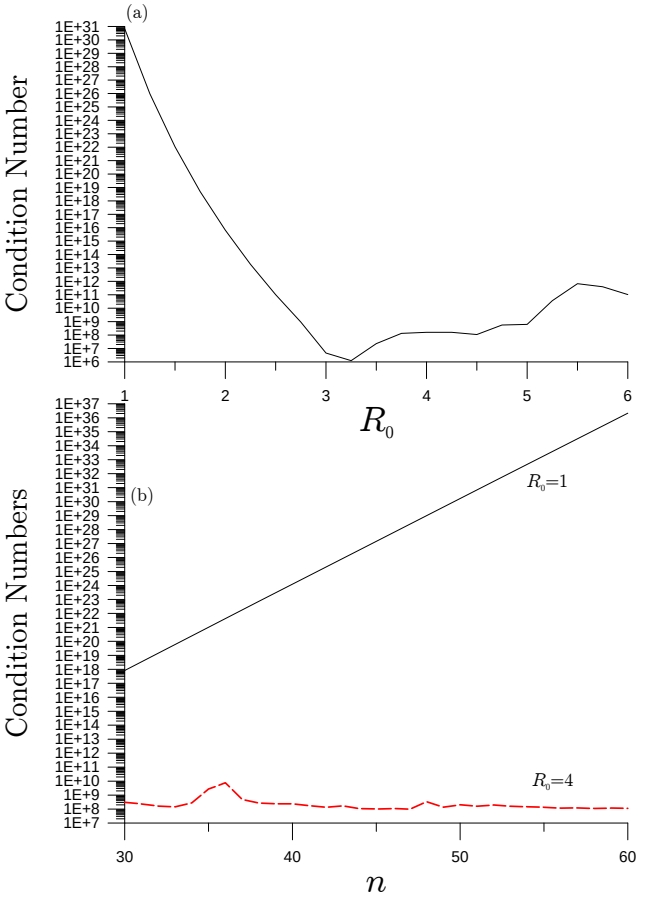

Figure 1: Comparing the condition numbers in (a) for different R_0, and (b) for different n with $R_0 = 1$ and $R_0 = 4$.

4.2 The integer nodes

We can assess the accuracy of the inversion of $\mathbf{V}$ by

$$e_1 = |\|\mathbf{V}^{-1}\mathbf{V}\| - \sqrt{n}|, \tag{20}$$

where n is the dimension of $\mathbf{V}$.

Let $x_i = i,\ i = 1,\ldots,n$ be the integer nodes of $\mathbf{V}$. We consider two cases with $R_0 = 1$ and $R_0 = n$, and apply the algorithm in Section 3 to calculate $\mathbf{V}^{-1}$. From Fig. 2 it can be seen that both the computations with $R_0 = 1$ and $R_0 = n$ have the same accuracy as $n \leq 14$. However, when $n > 14$ the case which does not consider

the characteristic length with $R_0 = 1$ leads to a large error, growing exponentially with n.

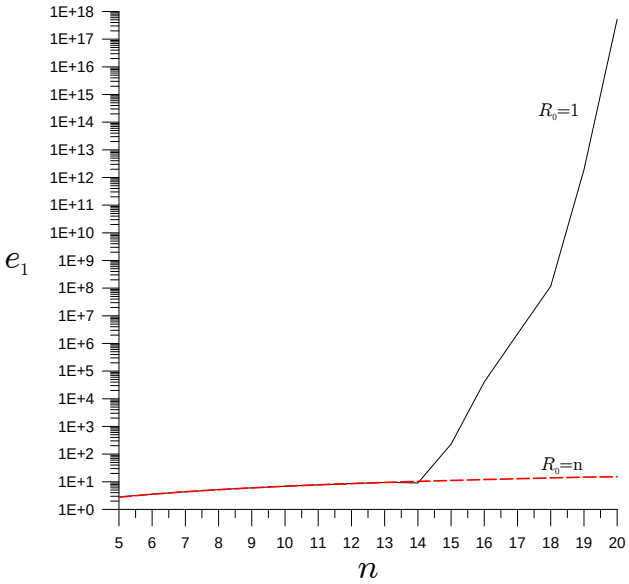

Figure 2: Plotting the error of e_1 with respect to n for $R_0 = 1$ and $R_0 = n$.

4.3 Equidistant nodes in the interval (0,1)

The nodes used in this test case are given by $x_i = i/(n+1),\ i = 1,\ldots,n$. Gohberg and Olshevsky (1997) have demonstrated the ill-condition of this case that $\text{Cond}(\mathbf{V}) = 6 \times 10^7$ when $n = 10$, and $\text{Cond}(\mathbf{V}) = 4 \times 10^{18}$ when $n = 30$. We investigate the condition numbers of $\mathbf{B}$ by using $R_i = x_{i+1} + R_0$, where $R_0 = 0.01$ and $R_0 = 0.1$ are used. In Fig. 3(a) we plot the condition numbers of $\mathbf{B}$ versus n, where we use the CGM to calculate $\mathbf{B}^{-1}$ under a convergence criterion 10^{-9}. The condition numbers for both cases are smaller than 10^9, which are much smaller than the condition number of $\mathbf{V}$ given above.

Corresponding to Eq. (20), we can also assess the accuracy of the inversion of $\mathbf{V}$ by

$$e_2 = |\|\mathbf{V}\mathbf{V}^{-1}\| - \sqrt{n}|. \tag{21}$$

Theoretically, $\mathbf{V}$ and $\mathbf{V}^{-1}$ are commutative, i.e., $\mathbf{V}\mathbf{V}^{-1} = \mathbf{V}^{-1}\mathbf{V} = \mathbf{I}_n$, and $e_1 = e_2 = 0$. But numerically, it is usually not true. For ill-conditioned matrices, in

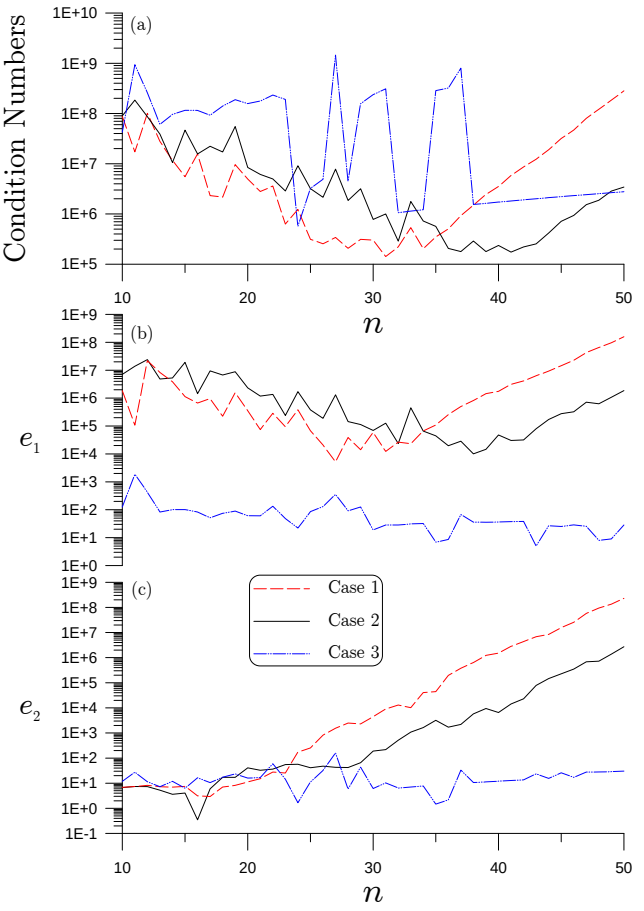

Figure 3: Under different cases: (a) plotting the condition numbers, (b) plotting the error of e_1, and (c) plotting the error of e_2.

the inversions by numerical method the rounding errors will be enlarged greatly. Jog (2004) has developed a new technique to suppress e_2 for lower-dimensional Vandermonde matrices.

How to effectively select the numbers R_i to suppress the errors of e_1 and e_2 is interesting. In Figs. 3(b) and 3(c) we consider three cases: case 1 with $R_i = x_{i+1} + 0.01$, case 2 with $R_i = x_{i+1} + 0.1$, and case 3 with $R_i = |R_0 + (i-1)[1-(n-1)R_0]/(n-2)|$ where $R_0 = 6$.

The solid line shows case 2 by considering the reduction technique in Section 2.2 with $R_i = x_{i+1} + 0.1$, while the dashed-dotted line for case 1 with $R_i = x_{i+1} + 0.01$.

They show that the numerical errors grow with n, and the numerical errors may be large up to 10^9 for e_1 and 10^8 for e_2 when $n = 50$. Conversely, for case 3, the numerical errors do not grow with n, and the numerical errors can be much smaller than the above two cases. It can be seen that by a suitable choice of R_i the errors can be greatly reduced when n is large.

4.4 The Runge phenomenon

The Runge phenomenon illustrates that a visible error can occur when we numerically construct a higher-order polynomial interpolant [Quarteroni, Sacco and Saleri (2000)]. The function to be interpolated is

$$f(x) = \frac{1}{1+25x^2}, \quad x \in [-1,1]. \tag{22}$$

The nodes used in this test case are given by $x_i = -1+2(i-1)/(n-1)$, $i = 1,\ldots,n$. Gohberg and Olshevsky (1997) have demonstrated the ill-condition of this case that $\text{Cond}(\mathbf{V}) = 7 \times 10^{18}$ when $n = 50$. We investigate the condition number of $\mathbf{B}$ by using $R_i = |x_{i+1}| + R_0$, where $R_0 \geq 0$. Here we fix $R_0 = 1.5$. In Fig. 4(a) we plot the condition number of $\mathbf{B}$ versus n, where we use the CGM to calculate $\mathbf{B}^{-1}$ under a convergence criterion 10^{-6}. The error of e_2 as defined above for $\mathbf{B}$ are also plotted in Fig. 4(b).

Skrzipek (2004) has computed the relative left residual $rlr = \|\mathbf{V}^{-1}\mathbf{V} - \mathbf{I}_n\|/\text{Cond}(\mathbf{V})$, and the relative right residual $rrr = \|\mathbf{V}\mathbf{V}^{-1} - \mathbf{I}_n\|/\text{Cond}(\mathbf{V})$. We only compare the case with $n = 50$. The results of $rlr = 5.35 \times 10^{-17}$ and $rrr = 2.77 \times 10^{-12}$ were obtained by Skrzipek (2004), while our results are $rlr = 1.20 \times 10^{-20}$ and $rrr = 8.96 \times 10^{-22}$. It can be seen that our results are much better than that calculated by Skrzipek (2004).

We apply the reduction technique in Section 2.2 by solving Eq. (10) to obtain $\bar{a}_i$, which are then inserted into the new interpolant in Eq. (7) to solve this problem. R_i is given as above with $R_0 = 1.6$. In Fig. 5(a) we compare the exact function with the interpolated polynomial, and even n is large up to 101, no oscillation is observed in the interpolant, where the interpolated error as shown in Fig. 5(b) is smaller than 0.0137. Even under a stringent convergence criterion 2×10^{-14} the CGM is convergent within 5562 iterations. We also apply the reduction technique in Section 2.1 to solve this problem by using $R_0 = 1.5$. The numerical error as shown in Fig. 5(b) by the dashed line is slightly larger than the above one, and by applying the CGM to solve this problem, it is convergent slowly under a stringent convergence criterion as given above, and instead of we employ 10^{-10} to be the convergence criterion in the use of CGM to solve the linear equations.

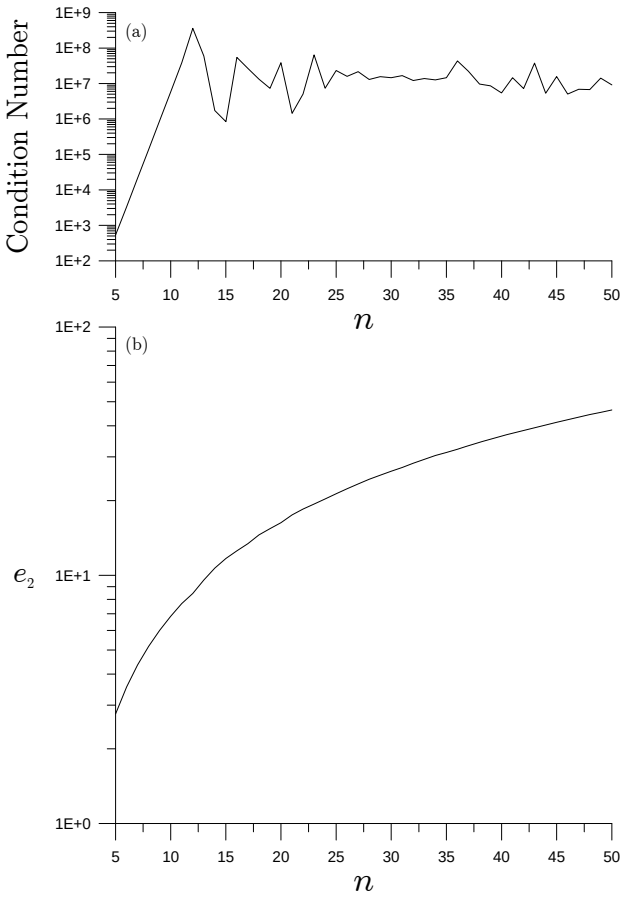

Figure 4: (a) Plotting the condition number, and (b) plotting the error of e_2.

This example demonstrates that even in the interval of $[-1,1]$ the introduction of an extra constant R_0 into the polynomial interpolation has the effect to stabilize the numerical solution for finding the coefficients in the polynomial interpolant. When one does not consider R_0 by letting $R_0 = 1$ as in the original interpolation equation (2), it is easy to induce numerical instability for finding the coefficients in the polynomial interpolant.

4.5 Differential quadrature

Bellman and Casti (1971), and Bellman, Kashef and Casti (1972) first proposed the Differential Quadrature (DQ), which is an approximation of the real differential

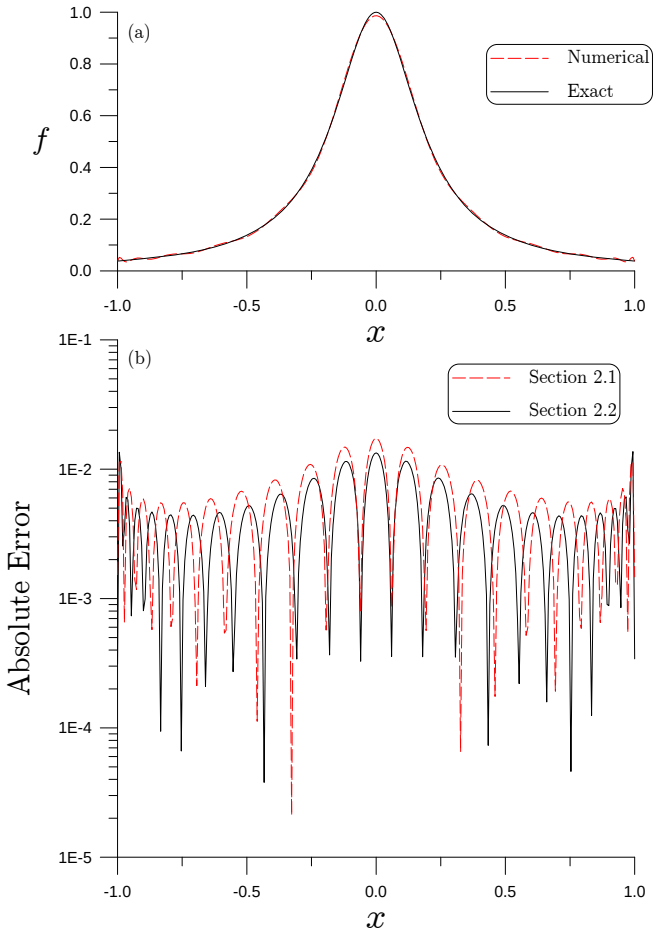

Figure 5: Comparing the numerical and exact solutions of a numerical example.

of a differentiable function to mimid the integral quadrature. Here, we consider a function $f(x)$ defined in a closed interval $x \in [a,b]$. It is supposed that there are n grid points with coordinates $x_1 = a, x_2, \ldots, x_n = b$. The function $f(x)$ is assumed to be differentiable at any grid point, so that its first-order derivative $f'(x)$ at any grid point x_i can be approximated by

$$f'(x_i) = \sum_{j=1}^{n} a_{ij} f(x_j). \tag{23}$$

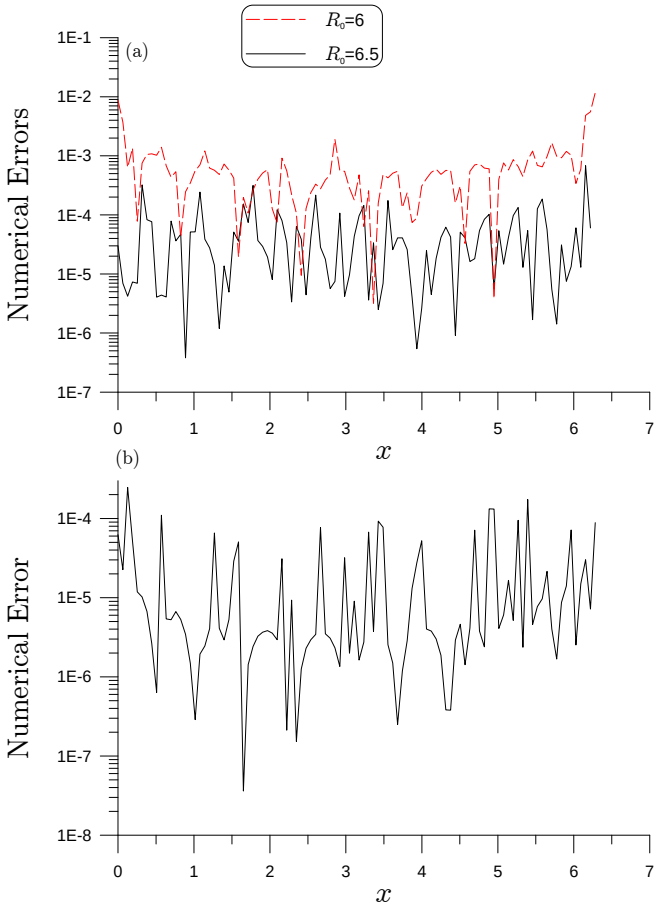

Figure 6: The numerical errors of differential quadrature obtained by the reducing method.

In the first approach of Bellman, Kashef and Casti (1972), the test functions are chosen as

$$g_k(x) = x^k, \quad k = 0, 1, \ldots, n-1, \tag{24}$$

such that we have the following algebraic equations to determine the weighting coefficients a_{ij}:

$$\begin{cases} \sum_{j=1}^{n} a_{ij} = 0, \\ \sum_{j=1}^{n} a_{ij} x_j = 1, \\ \sum_{j=1}^{n} a_{ij} x_j^k = k x_i^{k-1}, \quad k = 2, \ldots, n-1. \end{cases} \tag{25}$$

By inspection, we can see that for each fixed i, the above system is with the Vandermonde matrix given in Eq. (4) as the coefficient matrix. Therefore, we can apply the technique described in Section 2.1 to solve the above system. To demonstrate the efficiency of our technique to reduce the condition number of $\mathbf{V}$, we use the following example: $f(x)=\sin x,\ f'(x)=\cos x,\ \ x\in[0,2\pi]$ to test the accuracy. By fixing $n=100$ we apply the CGM to solve the reduced system with the system matrix $\mathbf{A}$ given in Eq. (6), where the convergence criterion of the CGM is fixed to be 10^{-12}. In Fig. 6(a) we show the numerical errors of $f'(x)=\cos x$ at different grid points under the given $R_0=6$ and $R_0=6.5$. It can be seen that the numerical errors are smaller than 5×10^{-4} when the value $R_0=6.5$ is larger than 2π. Conversely, when $R_0=6$ is smaller than 2π, the numerical errors increase to the second-order. Similarly, in Fig. 6(b) we show the numerical errors of $f'(x)=\cos x$ at different grid points by using the technique described in Section 2.2 under the given $R_i=x_i+R_0$ with $R_0=5.5$. It can be seen that the numerical errors are smaller than 10^{-4}, which is slightly better than that shown in Fig. 6(a).

As mentioned by Shu (2000), n is usually chosen to be less than 13, due to the severe ill-condition of $\mathbf{V}$. Here, we can take n very large, up to $n=100$. As an application we apply the DQ and the fictitious time integration method [Liu and Atluri (2008a] to solve the following boundary value problem [Liu (2006)]:

$$u''=\frac{3}{2}u^2, \tag{26}$$

$$u(0)=4,\ \ u(1)=1. \tag{27}$$

The exact solution is

$$u(x)=\frac{4}{(1+x)^2}. \tag{28}$$

After the work by Liu and Atluri (2008a), the author and his coworkers have applied the fictitious time integration method (FTIM) to solve many engineering problems [Liu and Atluri (2008b, 2008c, 2009b); Liu (2008b, 2008c, 2009a, 2009b, 2009c, 2009d, 2010); Chi, Yeih and Liu (2009); Ku, Yeih, Liu and Chi (2009); Chang and Liu (2009); Tsai, Liu and Yeih (2010)].

By introducing the DQ of u' and u'' at the grid points we can obtain

$$u'_i=\sum_{j=1}^{n}a_{ij}u_j,\ \ u''_i=\sum_{j=1}^{n}b_{ij}u_j, \tag{29}$$

where $u'_i=u'(x_i)$, $u''_i=u''(x_i)$, $x_i=(i-1)\Delta x$ with $\Delta x=1/(n-1)$ the grid length, and $b_{ij}=a_{ik}a_{kj}$. Thus we have to solve the following nonlinear algebraic equations

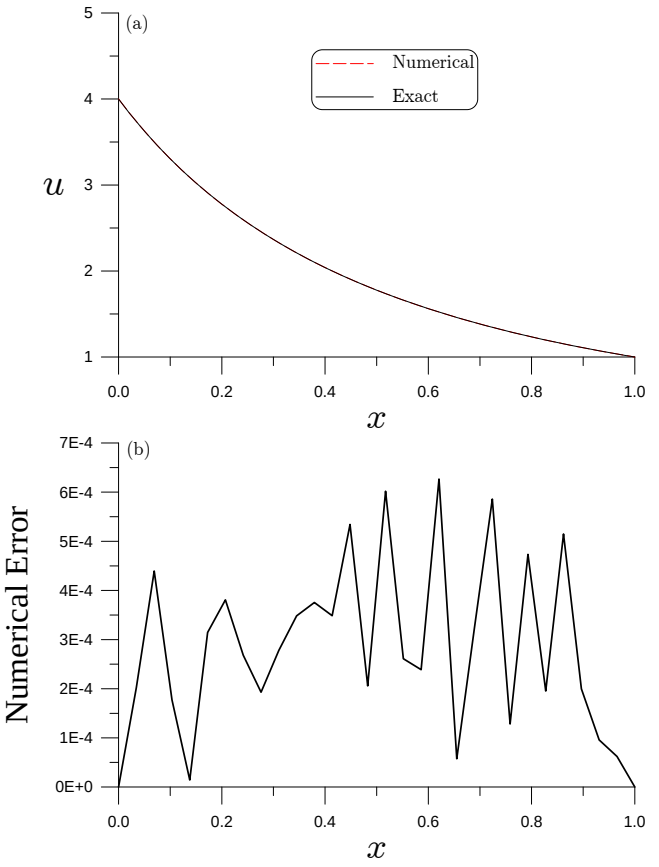

Figure 7: Applying the DQ and FTIM to a boundary value problem: (a) comparing numerical and exact solutions, and (b) displaying the numerical error.

for u_i:

$$F_i = \sum_{j=1}^{n} b_{ij} u_j - \frac{3}{2} u_i^2 = 0, \tag{30}$$

$$u_1 = 4, \quad u_n = 1. \tag{31}$$

Under the following parameters $R_0 = 1.5$, $n = 30$ we compute the solution of the above nonlinear system by the FTIM with $\Delta t = 0.01$ and $\nu = -0.12$, and compare them with the exact solution in Fig. 7(a), which can be seen that the error as shown in Fig. 7(b) is very small in the order of 10^{-4}.

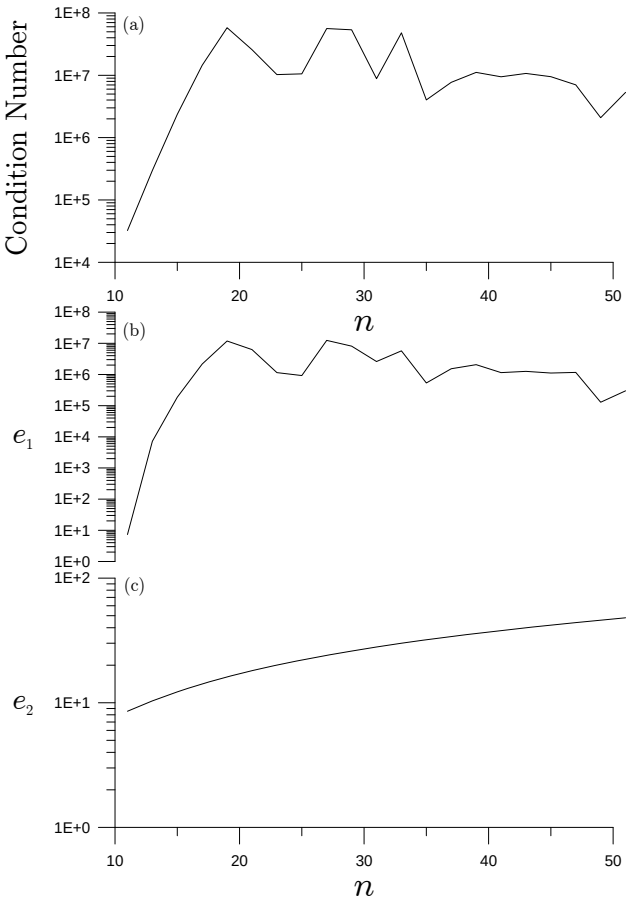

Figure 8: (a) Plotting the condition number, (b) plotting the error of e_1, and (c) plotting the error of e_2.

5 A highly accurate interpolation

In many applications one wants to interpolate the data as accurate as possible. But this is limited by the interpolation of n data with $(n-1)$-order polynomials, where the resulting Vandermonde matrices are highly ill-conditioned as measured by the Lebesgue constant $2^n/[e(n-1)\ln n]$. The results of Beckermann (2000) and Li (2006) show that in the best possible cases, the condition numbers of the Vandermonde matrices still grow exponentially with the order of the interpolant polynomial. Because of this, these days no one is interpolating a function by high-order polynomials in the usual bases $1, x, x^2, \ldots$, but rather in the Chebyshev polynomials.

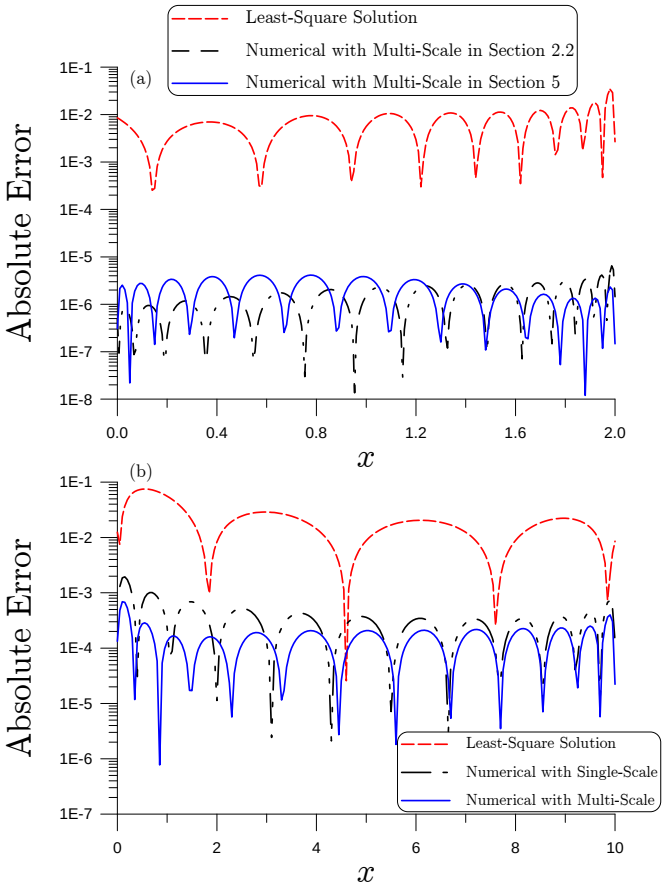

Figure 9: For cases (a) and (b), comparing the absolute errors due to the least-square method and the present methods.

In this section we propose a new interpolation by an m-order polynomial to interpolate the $n = 2m+1$ data points, where n is supposed to be an odd integer. We begin with

$$p_m(x) = c_0 + \sum_{k=1}^{m} c_k x^k. \tag{32}$$

Now, suppose that

$$c_k = \frac{a_k \cos(k\theta_k)}{R_k^k} + \frac{b_k \sin(k\theta_k)}{R_k^k}, \tag{33}$$

where

$$R_k = |x_{2k}| + R_0, \quad \theta_k = \frac{2k\pi}{m}, \quad k = 1, \ldots, m \tag{34}$$

are the sequences of numbers with a constant number R_0 determined by the user, and

$$a = x_0 < x_1 < x_2 < \ldots < x_{2m-1} < x_{2m} = b \tag{35}$$

are the interpolated points in the problem domain $[a, b]$.

Inserting Eq. (33) into Eq. (32) we can obtain

$$p_m(x) = a_0 + \sum_{k=1}^{m} \left[a_k \left(\frac{x}{R_k} \right)^k \cos(k\theta_k) + b_k \left(\frac{x}{R_k} \right)^k \sin(k\theta_k) \right], \tag{36}$$

where we let $c_0 = a_0$. Here, a_k and b_k are unknown constants. In order to obtain them, we impose the following n interpolated conditions:

$$p_m(x_i) = y_i, \quad i = 0, \ldots, n-1. \tag{37}$$

Thus, we obtain a linear equations system to determine a_k and b_k:

$$\begin{bmatrix} 1 & \frac{x_0\cos\theta_1}{R_1} & \frac{x_0\sin\theta_1}{R_1} & \ldots & \left(\frac{x_0}{R_m}\right)^m \cos m\theta_m & \left(\frac{x_0}{R_m}\right)^m \sin m\theta_m \\ 1 & \frac{x_1\cos\theta_1}{R_1} & \frac{x_1\sin\theta_1}{R_1} & \ldots & \left(\frac{x_1}{R_m}\right)^m \cos m\theta_m & \left(\frac{x_1}{R_m}\right)^m \sin m\theta_m \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \frac{x_{2m-1}\cos\theta_1}{R_1} & \frac{x_{2m-1}\sin\theta_1}{R_1} & \ldots & \left(\frac{x_{2m-1}}{R_m}\right)^m \cos m\theta_m & \left(\frac{x_{2m-1}}{R_m}\right)^m \sin m\theta_m \\ 1 & \frac{x_{2m}\cos\theta_1}{R_1} & \frac{x_{2m}\sin\theta_1}{R_1} & \ldots & \left(\frac{x_{2m}}{R_m}\right)^m \cos m\theta_m & \left(\frac{x_{2m}}{R_m}\right)^m \sin m\theta_m \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ b_1 \\ \vdots \\ a_m \\ b_m \end{bmatrix}$$

$$= \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{2m-1} \\ y_{2m} \end{bmatrix}. \tag{38}$$

First we apply the CGM to calculate the condition numbers of the above system with equidistant nodes in [0, 2], whose results together with e_1 and e_2 are plotted in Fig. 8 with respect to n. It can be seen that the condition numbers are reduced to the order of 10^7.

In order to show the accuracy of the new interpolation technique we consider the following functions:

$$\begin{aligned} &\text{(a)}\ \ f(x)=\frac{1}{1+(x-1)^2},\ \ 0\le x\le 2,\\ &\text{(b)}\ \ f(x)=\frac{1}{1+x},\ \ 0\le x\le 10. \end{aligned} \tag{39}$$

For case (a) we take $m=25$ and $R_0=1.23$, while $m=21$ and $R_0=1$ for case (b). The absolute errors are plotted in Fig. 9. Very accurate results are obtained, where the maximum error for case (a) is about 4.1×10^{-6}, and 6.9×10^{-4} for case (b). Even the errors e_1 and e_2 as shown in Fig. 8 are large, we have checked the residual errors in solving Eq. (38) by the CGM, where the residual error for case (a) is about 2.53×10^{-10}, and 1.1×10^{-6} for case (b).

For the purpose of comparison we also find the least-square solutions of Eq. (37) for the above two interpolated functions. For case (a) $m=25$ is used, and the maximum error of the least-square solution is about 3.4×10^{-2}. When the multi-scale numerical solution in Section 2.2 is applied to case (a), we use $n=2m+1=51$ and the same $R_0=1.3$. It results in a very accurate numerical solution with the maximum error 6.5×10^{-6}. As shown in Fig. 9(a) by the solid-dashed line the half-order and multi-scale numerical solution in Section 5 is more accurate than the multi-scale numerical solution in Section 2.2, and is much accurate than the least-square solution. For case (b), we find that when $m>4$ the numerical solution by the least-square is unstable, and we use $m=4$ for this case, whose maximum error is about 4.4×10^{-2}. As shown in Fig. 9(b) the multi-scale numerical solution is much accurate than the least-square solution. We also apply the single-scale numerical method in Section 2.1 to case (b) with $R_0=15$ and $n=50$, whose numerical error is shown in Fig. 9(b) by the solid-dashed line. It can be seen that the multi-scale numerical solution is more accurate than the single-scale numerical solution.

6 Conclusions

In this paper we have proposed two novel diagonal preconditioners to reduce the condition number of Vandermonde matrices, and a new algorithm is given to calculate the inverse of the resulting coefficient matrix. We applied the new techniques in the data interpolation by using very high-order polynomials, and very accurate results were obtained. For the differential quadratures, the present approach was able to provide very accurate results of the numerical differentials. A numerical example of nonlinear boundary value problem was verified by using the proposed differential quadrature together with the fictitious time integration method to solve

the resulting nonlinear algebraic equations. A new technique by using the half-order polynomials with a multi-scale technique to interpolate the given data was derived. Numerical results confirmed the validity of the present approach for polynomial interpolation with a multi-scaling technique, which were better than that obtained by a single-scale polynomial interpolation, and were much better than that obtained by the least-square method. Moreover, because we only need to specify an extra constant R_0 in the multi-scale expansion, it was easily adopted in the polynomial interpolation for significantly improving the numerical accuracy.

Acknowledgement: Taiwan's National Science Council project NSC-100-2221-E-002-165-MY3 granted to the author is highly appreciated.

References

Bauer, F. L. (1963): Optimally scaled matrices. *Num. Math.*, vol. 5, pp. 73-87.

Beckermann, B. (2000): The condition number of real Vandermonde, Krylov and positive definite Hankel matrices. *Num. Math.*, vol. 85, pp. 553-577.

Bellman, R. E.; Casti, J. (1971): Differential quadrature and long-term integration. *J. Math. Anal. Appl.*, vol. 34, pp. 235-238.

Bellman, R. E.; Kashef, B. G.; Casti, J. (1972): Differential quadrature: a technique for the rapid solution of nonlinear partial differential equations. *J. Comp. Phys.*, vol. 10, pp. 40-52.

Björck, A.; Pereyra, V. (1970): Solution of Vandermonde systems of equations. *Math. Comput.*, vol. 24, pp. 893-903.

Calvetti, D.; Reichel, L. (1993): Fast inversion of Vanderomnde-like matrices involving orthogonal polynomials. *BIT*, vol. 33, pp. 473-484.

Chang, C. W.; Liu, C.-S. (2009): A fictitious time integration method for backward advection-dispersion equation. *CMES: Computer Modeling in Engineering & Sciences*, vol. 51, pp. 261-276.

Chen, Y. W.; Liu, C.-S.; Chang, J. R. (2009): Applications of the modified Trefftz method for the Laplace equation. *Eng. Anal. Bound. Elem.*, vol. 33, pp. 137-146.

Chen, Y. W.; Liu, C.-S.; Chang, C. M.; Chang, J. R. (2010): Applications of the modified Trefftz method to the simulation of sloshing behaviours. *Eng. Anal. Bound. Elem..*, vol. 34, pp. 581-598.

Chen, Y. W.; Yeih, W.; Liu, C.-S.; Chang, J. R. (2012): Numerical simulation of the two-dimensional sloshing problem using a multi-scaling Trefftz method. *Eng.*

Anal. Bound. Elem., vol. 36, pp. 9-29.

Chi, C. C.; Yeih, W.; Liu, C.-S. (2009): A novel method for solving the Cauchy problem of Laplace equation using the fictitious time integration method. *CMES: Computer Modeling in Engineering & Sciences*, vol. 47, pp. 167-190.

Gautschi, W. (1975): Norm estimates for inverses of Vandermonde matrices. *Num. Math.*, vol. 23, pp. 337-347.

Gautschi, W.; Inglese, G. (1988): Lower bounds for the condition number of Vandermonde matrices. *Num. Math.*, vol. 52, pp. 241-250.

Gohberg, I.; Olshevsky, V. (1997): The fast generalized Parker-Traub algorithm for inversion of Vandermonde and related matrices. *J. Complexity*, vol. 13, pp. 208-234.

Higham, N. J. (1987): Error analysis of the Björck-Pereyra algorithms for solving Vandermonde systems. *Num. Math.*, vol. 50, pp. 613-632.

Higham, N. J. (1988): Fast solution of Vandermonde-like systems involving orthogonal polynomials. *IMA J. Num. Anal.*, vol. 8, pp. 473-486.

Jog, C. S. (2004): The accurate inversion of Vandermonde matrices. *Comp. Math. Appl.*, vol. 47, pp. 921-929.

Ku, C. Y.; Yeih, W.; Liu, C.-S.; Chi, C. C. (2009): Applications of the fictitious time integration method using a new time-like function. *CMES: Computer Modeling in Engineering & Sciences*, vol. 43, pp. 173-190.

Li, R. C. (2006): Asymptotically optimal lower bounds for the condition number of a real Vandermonde matrix. *SIAM J. Matrix Anal. Appl.*, vol. 28, pp. 829-844.

Liu, C.-S. (2006): The Lie-group shooting method for nonlinear two-point boundary value problems exhibiting multiple solutions. *CMES: Computer Modeling in Engineering & Sciences*, vol. 13, pp. 149-163.

Liu, C.-S. (2007a): A modified Trefftz method for two-dimensional Laplace equation considering the domain's characteristic length. *CMES: Computer Modeling in Engineering & Sciences*, vol. 21, pp. 53-65.

Liu, C.-S. (2007b): A highly accurate solver for the mixed-boundary potential problem and singular problem in arbitrary plane domain. *CMES: Computer Modeling in Engineering & Sciences*, vol. 20, pp. 111-122.

Liu, C.-S. (2007c): An effectively modified direct Trefftz method for 2D potential problems considering the domain's characteristic length. *Eng. Anal. Bound. Elem.*, vol. 31, pp. 983-993.

Liu, C.-S. (2008a): A highly accurate collocation Trefftz method for solving the Laplace equation in the doubly-connected domains. *Numer. Meth. Partial Diff.*

Eq., vol. 24, pp. 179-192.

Liu, C.-S. (2008b): A time-marching algorithm for solving non-linear obstacle problems with the aid of an NCP-function. *CMC: Computers, Materials & Continua*, vol. 8, pp. 53-65.

Liu, C.-S. (2008c): A fictitious time integration method for two-dimensional quasilinear elliptic boundary value problems. *CMES: Computer Modeling in Engineering & Sciences*, vol. 33, pp. 179-198.

Liu, C.-S. (2009a): A fictitious time integration method for solving *m*-point boundary value problems. *CMES: Computer Modeling in Engineering & Sciences*, vol. 39, pp. 125-154.

Liu, C.-S. (2009b): A fictitious time integration method for the Burgers equation. *CMC: Computers, Materials & Continua*, vol. 9, pp. 229-252.

Liu, C.-S. (2009c): A fictitious time integration method for solving delay ordinary differential equations. *CMC: Computers, Materials & Continua*, vol. 10, pp. 97-116.

Liu, C.-S. (2009d): A fictitious time integration method for a quasilinear elliptic boundary value problem, defined in an arbitrary plane domain. *CMC: Computers, Materials & Continua*, vol. 11, pp. 15-32.

Liu, C.-S. (2010): The fictitious time integration method to solve the space- and time-fractional Burgers equations. *CMC: Computers, Materials & Continua*, vol. 15, pp. 221-240.

Liu, C.-S.; Atluri, S. N. (2008a): A novel time integration method for solving a large system of non-linear algebraic equations. *CMES: Computer Modeling in Engineering & Sciences*, vol. 31, pp. 71-83.

Liu, C.-S.; Atluri, S. N. (2008b): A novel fictitious time integration method for solving the discretized inverse Sturm-Liouville problems, for specified eigenvalues. *CMES: Computer Modeling in Engineering & Sciences*, vol. 36, pp. 261-285.

Liu, C.-S.; Atluri, S. N. (2008c): A fictitious time integration method (FTIM) for solving mixed complementarity problems with applications to non-linear optimization. *CMES: Computer Modeling in Engineering & Sciences*, vol. 34, pp. 155-178.

Liu, C.-S.; Atluri, S. N. (2009a): A highly accurate technique for interpolations using very high-order polynomials, and its applications to some ill-posed linear problems. *CMES: Computer Modeling in Engineering & Sciences*, vol. 43, pp. 253-276.

Liu, C.-S.; Atluri, S. N. (2009b): A Fictitious time integration method for the numerical solution of the Fredholm integral equation and for numerical differentiation

of noisy data, and its relation to the filter theory. *CMES: Computer Modeling in Engineering & Sciences*, vol. 41, pp. 243-261.

Liu, C.-S.; Hong, H. K.; Atluri, S. N. (2010): Novel algorithms based on the conjugate gradient method for inverting ill-conditioned matrices, and a new regularization method to solve ill-posed linear systems. *CMES: Computer Modeling in Engineering & Sciences*, vol. 60, pp. 279-308.

Liu, C.-S.; Yeih, W.; Atluri, S. N. (2009): On solving the ill-conditioned system $\mathbf{Ax} = \mathbf{b}$: general-purpose conditioners obtained from the boundary-collocation solution of the Laplace equation, using Trefftz expansions with multiple length scales. *CMES: Computer Modeling in Engineering & Sciences*, vol. 44, pp. 281-311.

Neagoe, V. E. (1996): Inversion of the Van der Monde matrix. *IEEE Sign. Proce. Lett.*, vol. 3, pp. 119-120.

Quarteroni, A.; Sacco, R.; Saleri, F. (2000): Numerical Mathematics. Springer-Verlag, Berlin.

Shen, Y. H.; Liu, C.-S. (2011): A new insight into the differential quadrature method in solving 2-D elliptic PDEs. *CMES: Computer Modeling in Engineering & Sciences*, vol. 71, pp. 157-178.

Shu, C. (2000): Differential Quadrature and Its Application in Engineering. Springer-Verlag, London.

Skrzipek, M. R. (2004): Inversion of Vandermonde-like matrices. *BIT*, vol. 44, pp. 291-306.

Tsai, C. C.; Liu, C.-S.; Yeih, W. (2010): Fictitious time integration method of fundamental solutions with Chebyshev polynomials for solving Poisson-type nonlinear PDEs. *CMES: Computer Modeling in Engineering & Sciences*, vol. 56, pp. 131-151.

van der Sluis, A. (1969): Condition numbers and equilibration of matrices. *Num. Math.*, vol. 14, pp. 14-23.

Wertz, H. J. (1965): On the numerical inversion of a recurrent problem: the Vandermonde matrix. *IEEE Trans. Auto. Control*, vol. AC-10, p.492.