 CMES, vol.43, no.3, pp.253-276, 2009

A Highly Accurate Technique for Interpolations Using Very High-Order Polynomials, and Its Applications to Some Ill-Posed Linear Problems

Chein-Shan Liu[1] **and Satya N. Atluri**[2]

Abstract: Since the works of Newton and Lagrange, interpolation had been a mature technique in the numerical mathematics. Among the many interpolation methods, global or piecewise, the polynomial interpolation $p(x) = a_0 + a_1 x + \ldots + a_n x^n$ expanded by the monomials is the simplest one, which is easy to handle mathematically. For higher accuracy, one always attempts to use a higher-order polynomial as an interpolant. But, Runge gave a counterexample, demonstrating that the polynomial interpolation problem may be ill-posed. Very high-order polynomial interpolation is very hard to realize by numerical computations. In this paper we propose a new polynomial interpolation by $p(x) = \bar{a}_0 + \bar{a}_1 x/R_0 + \ldots + \bar{a}_n x^n/R_0^n$, where R_0 is a characteristic length used as a parameter, and chosen by the user. The resulting linear equations system to solve the coefficients $\bar{a}_\alpha$ is well-conditioned, if a suitable R_0 is chosen. We define a non-dimensional parameter, $R_0^* = R_0/(b-a)$ [where a and b are the end-points of the interval for x]. The range of values for R_0^* for numerical stability is identified, and one can overcome the difficulty due to Runge, as well as increase the accuracy and stability in the numerical interpolation by very-high-order polynomials, for these values of R_0^*. Numerical results which validate the current theory are presented for (i) the first and higher-order derivatives of noisy numerical data [such as determining da/dN in fatigue mechanics], (ii) the solution of the Abel integral equation under noisy data, and (iii) the numerical determination of an inverse Laplace transform under noisy data. These results confirm the validity of the present approach for very high-order polynomial interpolation.

Keywords: Very High-Order Polynomial Interpolation, Characteristic length, Numerical derivatives of noisy data, Abel integral equation, Inverse Laplace transform, Vandermonde matrices

[1] Department of Civil Engineering, National Taiwan University, Taipei, Taiwan. E-mail: liuchei.nshan@msa.hinet.net

[2] Center for Aerospace Research & Education, University of California, Irvine

1 Introduction

Interpolation is the process of using known data at discrete locations, to estimate data everywhere in the domain. Various interpolation techniques are often used in the engineering sciences. One of the simplest methods, linear interpolation, requires the knowledge of data at two discrete locations and the constant rate of change of the data between these two locations. With this information, one may interpolate values anywhere between those two locations. More sophisticated interpolations are discussed in many textbooks and papers.

Computer modeling in engineering and the sciences has spurred an enormous interest in the methods of interpolating data, or of approximating 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 over a finite interval can be approximated uniformly, within any preassigned error, by polynomials, in practical computations there appeared a counterexample due to Runge. On the other hand, finding an nth-order polynomial function $p(x) = a_0 + a_1 x + \ldots + a_n x^n$ to best match a continuous function $f(x)$ in the interval of $x \in [0,1]$, for large values of n, say $n \geq 10$, leads to a highly ill-conditioned system of linear equations to determine the coefficients a_α, where the system matrix is a Hilbert matrix. This makes the interpolation by very high-order polynomials not easy for numerical implementation. In this paper we propose a new technique of interpolation by very high-order polynomials, which can overcome the abovementioned ill-conditioned behavior.

There are a lot of applications in numerical mathematics, which use approximations by very high-order polynomials. In this paper we will apply the new interpolation method to solve the problem of numerical differentiation of noisy data, and the problem of an inverse Laplace transform. In many applications one is required to calculate the derivative of a function which is measured experimentally, i.e., to differentiate noisy data. The problem of numerical differentiation of noisy data is ill-posed: small changes of the data may result in large changes of the derivative [Ramm and Smirnova (2001); Ahn et al. (2006)]. In this paper, we present an example from fatigue mechanics, namely, that of determining (da/dN) [a is the crack-length, and N is the number of fatigue cycles], as a function of a.

A possible use of the presently proposed method of approximation, using very high-order polynomials, is to solve the following Abel integral equation under noise:

$$\int_0^s \frac{\phi(t)}{(s-t)^\eta} dt = h(s),\ s > 0,\ \ \eta \in (0,1). \tag{1}$$

It is known that Eq. (1) has the exact solution given by

$$\phi(s) = \frac{\sin(\eta\pi)}{\pi} \frac{d}{ds} \int_0^s \frac{h(t)}{(s-t)^{1-\eta}} dt. \tag{2}$$

Nevertheless, the exact solution fails in a practical application, when the input function $h(t)$ is given with a random error, because the differential operator involved in Eq. (2) is ill-posed and unbounded.

There were many approaches for determining the numerical solution of the Abel integral equation [Gorenflo and Vessella (1991)]. Fettis (1964) has proposed a numerical solution of the Abel equation by using the Gauss-Jacobi quadrature rule. Kosarev (1973) proposed a numerical solution of the Abel equation by using the orthogonal polynomials expansion. Piessens and Verbaeten (1973) and Piessens (2000) developed an approximate solution of the Abel equation by means of the Chebyshev polynomials of the first kind. When the input data are with noisy error, Hao (1985,1986) used the Tikhonov regularization technique, and Murio et al. (1992) suggested a stable numerical solution. Furthermore, Garza et al. (2001) and Hall et al. (2003) used the wavelet method, and Huang et al. (2008) used the Taylor expansion method to derive the inversion of noisy Abel equation. Recently, Liu and Atluri (2009) have developed a novel technique of a fictitious time integration method to resolve the problem of numerical differentiation of noisy data and applied it to the inversion of the above Abel integral equation under noise. The results in Liu and Atluri (2009) were very good. In this paper we show that the presently proposed method of very-high-order polynomial interpolation leads to a better way to solve the ill-posed problem of numerical differentiation of noisy data, than that presented in Liu and Atluri (2009), even when higher-order derivatives of noisy data are required.

2 A novel technique for improving the conditioning of the coefficient-matrix, for interpolation by very high-order polynomials

Polynomial interpolation is the interpolation of a given data at a number of discrete spatial locations, by a polynomial. In other words, given some data points, such as obtained by the sampling of a measurement, the aim is to find a polynomial which goes exactly through these points.

Given a set of $n+1$ spatial locations x_i $(i = 0,\ldots,n)$ where the respective data y_i $(i = 0,\ldots,n)$ are given [where no two x_i are the same], one is looking for a polynomial $p(x)$ of order at most n with the following property:

$$p(x_i) = y_i, \;\; i = 0,1,\ldots,n, \tag{3}$$

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

Here, we are interested in cases where n is very large, say $n \geq 10$. The unisolvence theorem states that such a polynomial $p(x)$ exists and is unique. This can be proved by using the Vandermonde matrix. Suppose that the interpolation polynomial is in the form of

$$p(x) = a_0 + a_1 x + \ldots + a_n x^n = \sum_{\alpha=0}^{n} a_\alpha x^\alpha, \tag{4}$$

where x^α constitute a monomial basis. The statement that $p(x)$ interpolates the data points means that Eq. (3) must hold.

If we substitute Eq. (4) into Eq. (3), we obtain a system of linear equations to determine the coefficients a_α. The system in a matrix-vector form reads as

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^{n-1} & x_0^n \\ 1 & x_1 & x_1^2 & \ldots & x_1^{n-1} & x_1^n \\ \vdots & \vdots & \vdots & \ldots & \vdots & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \ldots & x_{n-1}^{n-1} & x_{n-1}^n \\ 1 & x_n & x_n^2 & \ldots & x_n^{n-1} & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \\ a_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix}. \tag{5}$$

We have to solve the above system for a_α in order to construct the interpolant $p(x)$. The matrix on the left is commonly referred to as a Vandermonde matrix denoted by $\mathbf{V}$. Its determinant is nonzero; which proves the unisolvence theorem: there exists a unique interpolating polynomial.

In a practical application of Eq. (5) in the engineering problems, the data $\mathbf{y} = [y_0, y_1, \ldots, y_n]$ are rarely given exactly; instead, noises in y_i are unavoidable, due to measurement and modeling errors. Therefore, we may encounter the problem that the numerical solution of Eq. (5) may deviate from the exact one to a great extent, when $\mathbf{V}$ is severely ill-conditioned and $\mathbf{y}$ is perturbed by noise. Indeed, the condition number of the Vandermonde matrix may be very large [Gautschi (1975)], causing large errors when computing the coefficients a_α, if the system of equations is solved using a numerical method. Several authors have therefore proposed algorithms which exploit the structure of the Vandermonde matrix to compute numerically stable solutions, instead of using the Gaussian elimination [Higham (1987,1988); Björck and Pereyra (1970); Calvetti and Reichel (1993)]. These methods rely on constructing first a Newton interpolation of the polynomial and then converting it to the monomial form above.

Our strategy to solve this ill-conditioned problem of polynomial interpolation, for very high-order interpolation, say $n \geq 10$, is to consider a set of re-defined undeter-

mined coefficients:

$$\bar{a}_\alpha = R_0^\alpha a_\alpha, \tag{6}$$

where R_0 is a characteristic length of the problem domain with $[a,b] \subset [-R_0, R_0]$. Such that, from Eq. (4), we have a new polynomial interpolant:

$$p(x) = \sum_{\alpha=0}^{n} \bar{a}_\alpha \left(\frac{x}{R_0}\right)^\alpha. \tag{7}$$

If we substitute Eq. (7) into Eq. (3), we obtain a system of linear equations, to determine the coefficients $\bar{a}_\alpha$:

$$\begin{bmatrix} 1 & \frac{x_0}{R_0} & \left(\frac{x_0}{R_0}\right)^2 & \cdots & \left(\frac{x_0}{R_0}\right)^{n-1} & \left(\frac{x_0}{R_0}\right)^n \\ 1 & \frac{x_1}{R_0} & \left(\frac{x_1}{R_0}\right)^2 & \cdots & \left(\frac{x_1}{R_0}\right)^{n-1} & \left(\frac{x_1}{R_0}\right)^n \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ 1 & \frac{x_{n-1}}{R_0} & \left(\frac{x_{n-1}}{R_0}\right)^2 & \cdots & \left(\frac{x_{n-1}}{R_0}\right)^{n-1} & \left(\frac{x_{n-1}}{R_0}\right)^n \\ 1 & \frac{x_n}{R_0} & \left(\frac{x_n}{R_0}\right)^2 & \cdots & \left(\frac{x_n}{R_0}\right)^{n-1} & \left(\frac{x_n}{R_0}\right)^n \end{bmatrix} \begin{bmatrix} \bar{a}_0 \\ \bar{a}_1 \\ \vdots \\ \bar{a}_{n-1} \\ \bar{a}_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix}. \tag{8}$$

In order to overcome the difficulties which appear in the conventional collocation Trefftz method to solve the Laplace equation, Liu (2007a,2007b,2007c,2008) has proposed a modified Trefftz method, and 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 overcome the ill-conditioning of the original Vandermonde matrix by including a characteristic length R_0 into the coefficient-matrix.

3 Comparing the condition numbers of the coefficient-matrices

Here we compare the condition numbers of the system matrices in Eqs. (5) and (8) for different n and R_0. The condition number of a system matrix $\mathbf{A}$ is defined as:

$$\mathrm{Cond}(\mathbf{A}) = \|\mathbf{A}\|\|\mathbf{A}^{-1}\|. \tag{9}$$

The norm used for $\mathbf{A}$ is the Frobenius norm defined by $\|\mathbf{A}\| := \sqrt{\sum_{i,j=1}^{n} A_{ij}^2}$, where A_{ij} is the ij-th component of $\mathbf{A}$. The Frobenius norm of a matrix is a direct extension of the Euclidean norm for a vector.

We define a dimensionless measure of R_0 by

$$R_0^* := \frac{R_0}{b-a}. \tag{10}$$

For the general interpolated interval $[a,b]$, three ranges of R_0^* need to be identified. If $b > a \geq 0$ (non-negative interval) or $a < b \leq 0$ (non-positive interval), we require $R_0^* \geq 1$ for a numerically stable solution of Eq. (8). The other cases are when $ab < 0$. If $b = \eta|a|$, where $\eta \geq 1$, then we need

$$R_0^* = \frac{R_0}{b-a} \geq \frac{b}{b-a} = \frac{-\eta a}{-(1+\eta)a} = \frac{\eta}{1+\eta}. \tag{11}$$

Similarly, if $-a = \eta b$, where $\eta \geq 1$, then we need

$$R_0^* = \frac{R_0}{b-a} \geq \frac{-a}{b-a} = \frac{\eta b}{(1+\eta)b} = \frac{\eta}{1+\eta}. \tag{12}$$

Thus, for the symmetric interval, i.e., $b = -a$, we need $R_0^* \geq 0.5$. Because of the fact that $\eta/(1+\eta) < 1$, if we take $R_0^* \geq 1$, the above all requirements are satisfied automatically. However, for the accuracy of interpolation, and also taking the numerical stability into account, it is better to choose a suitable R_0^* in a range not violating the above inequality for each type interval. The quantity $\eta/(1+\eta)$ is denoted by $R_c := \eta/(1+\eta)$. There are two limiting cases: $R_c = 1$ when $\eta \to \infty$, and $R_c = 0.5$ when $\eta = 1$. Therefore, we conclude that when $R_0^* \geq R_c$, the numerical interpolation is stable for very-high-order polynomials.

We give an example to test the condition number of Eq. (8). The interval of $[a,b] = [0,4]$ is fixed and is non-negative, and we divide it into $n+1$ discrete spatial locations $x_i = i\Delta x = i(b-a)/n$. Taking the polynomial-order $n = 50$ and $n = 100$, respectively, we apply the congugate gradient method (CGM) to find the inverse matrix with a convergence criterion of 10^{-8}. The condition numbers are calculated by Eq. (9), and their variations with respect to $R_0^* = R_0/(b-a)$ are plotted in Fig. 1, with the solid line for $n = 50$ and the dashed line for $n = 100$. It can be seen that when R_0^* is smaller than 0.75 the condition numbers increase very fast for both the case of $n = 50$ and $n = 100$, respectively, to a value larger than 10^{28} when $R_0^* = 0.25$ for the case of $n = 50$. More seriously, the condition numbers blow up when $R_0^* < 0.6$ for the case of $n = 100$. Conversely, when $R_0^* \geq 1$ (which is located in the stable region for a non-negative interval) the condition numbers of both cases tend to stable values which are smaller than 10^9.

In Fig. 2 we plot the condition number with respect to the order n of the polynomial, in the range of $10 \leq n \leq 40$, for $R_0^* = 0.25$ and $R_0^* = 2.5$. It can be seen that for $R_0^* = 0.25$ the condition number of the Vandermonde matrix increases exponentially

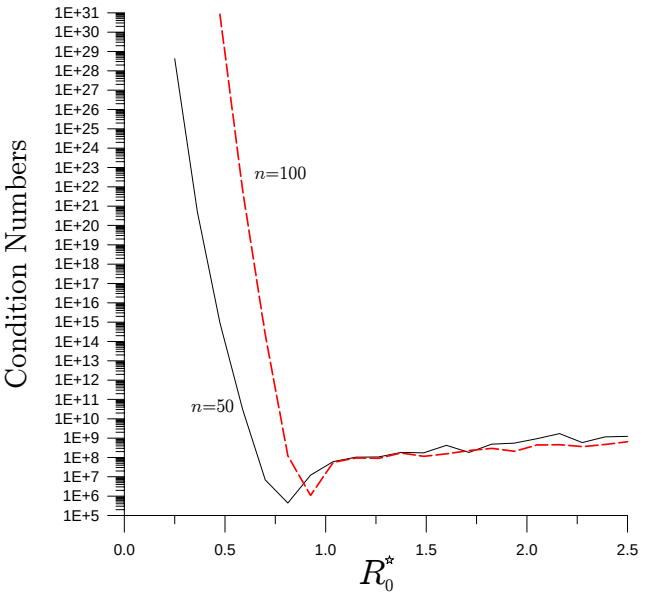

Figure 1: Plotting the condition numbers of the coefficient-matrices, with respect to R_0^*, in the cases when $n = 50$ and $n = 100$.

with respect to n [Gautschi and Inglese (1988); Skrzipek (2004)], and with a huge value up to 10^{38} when $n = 35$. Conversely, when $R_0^* = 2.5$, the condition number can be controlled almost to within an order of 10^9.

In order to further demonstrate the ill-conditioned behavior of polynomial interpolation, we consider a high-order interpolation in the intervals $[0,1]$ and $[-1,1]$, respectively. In Table 1 we list some condition numbers of the above mentioned Vandermonde matrices [Gohberg and Olshevsky (1997)]. It can be seen that the condition numbers grow fast when n increases. For the purpose of comparison we also calculate the condition numbers of the new matrix $\mathbf{A}$ with $R_0^* = 1.5$. The result is plotted in Fig. 2 by a dashed-dotted blue line, of which the condition numbers are greatly reduced to the order of 10^9. Table 1 shows that the ill-conditioned behavior for a very high-order polynomial interpolation in the interval $[0,1]$ may happen. The introduction of the scaling factor of R_0^* into the polynomial interpolation can reduce its ill-conditioned behavior.

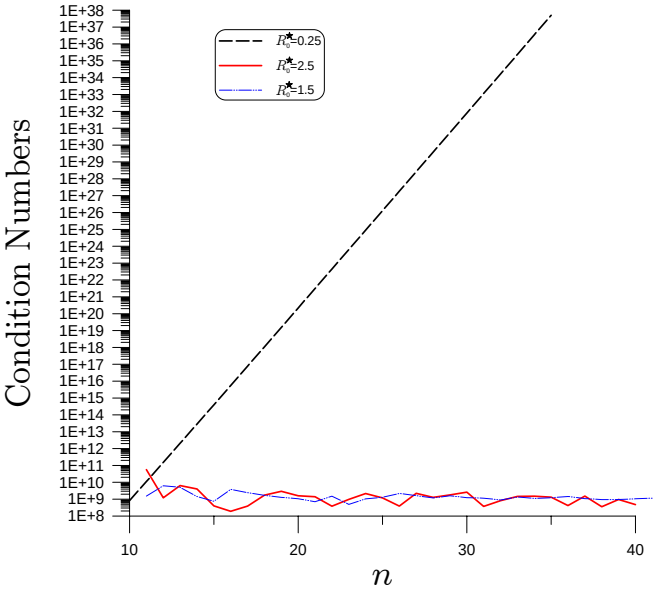

Figure 2: Plotting the condition number of the coefficient-matrix with respect to the number of monomials, for $R_0^* = 0.25$, $R_0^* = 1.5$ and $R_0^* = 2.5$.

Table 1: The condition numbers of the coefficient-matrix, with equidistant nodes in $(0,1)$ and $(-1,1)$

n	Cond($\mathbf{V}$)	Equidistant in $(0,1)$
5	2×10^{3}	
10	6×10^{7}	
20	4×10^{16}	
30	4×10^{18}	
40	8×10^{18}	
50	6×10^{18}	
		Equidistant in $(-1,1)$
5	2×10^{1}	
10	5×10^{3}	
20	3×10^{8}	
30	2×10^{13}	
40	1×10^{18}	
50	7×10^{18}	

4 Applications

4.1 The Runge phenomenon

Runge's phenomenon illustrates that significant errors can occur when constructing a polynomial interpolant of higher order [Quarteroni et al. (2000)]. An example function to be interpolated is

$$f(x) = \frac{1}{1+x^2}, \quad x \in [-5,5]. \tag{13}$$

We apply the new interpolation technique to solve this problem by taking $R_0^* = 1$ [for the symmetric interval $R_c = 0.5$]. In Fig. 3(a) we compare the exact function with the interpolated polynomial, and even when n is large, up to 100, no oscillation is observed in the interpolant. In Fig. 3(b) we show the coefficients $\bar{a}_\alpha, \ \alpha = 1, \ldots, 100$, and in Fig. 3(c) the coefficients $a_\alpha, \ \alpha = 1, \ldots, 100$. Thus, it is seen that the usual Runge's phenomenon can be overcome by the present interpolation technique.

For comparison purpose we also plot the numerical result in Fig. 3(a), by the dashed-dotted line, for the case when $R_0^* = 0.1$ and $n = 10$. It is obvious that when $R_0^* = 0.1$ (i.e., $R_0 = 1$), the Runge phenomenon occurs. Runge's phenomenon shows that for the original interpolation technique with high values of n, the interpolation polynomial may oscillate wildly between the data points. This problem is commonly resolved by the use of spline interpolation. However, in such a case, the interpolant is not a globally defined polynomial, but an assemblage of a chain of several local polynomials of a lower order.

In order to appreciate the accuracy of the present method, we compare our numerical results under $R_0^* = 0.8$ and $R_0^* = 1$, with the numerical results obtained from the tenth-degree Lagrange interpolation, $L_{10}(x)$, as well as the third-degree spline interpolation, $S(x)$, in Table 2 for some spatial location points. It can be seen that the accuracy of the present method is better than that of the spline interpolation, and is much better than that of the Lagrange interpolation. Overall, by using $R_0^* = 0.8$ the accuracy of the present method for the Runge example is in the order of 10^{-3}. The numerical results by using $R_0^* = 0.8$ are slightly better than that by using $R_0^* = 1$.

In Fig. 4 we compare the numerical errors for $R_0^* = 0.8, 1, 1.2$ and 2 with a fixed $n = 100$. All the erros are smaller than 0.05. The errors increase when R_0^* increases. But there is a lower bound of R_0^*, and the value of R_0^* smaller than 0.5 may induce numerical instability when n increases to a large value.

The ill-posedness of the interpolation problem is not fully due to the size of interpolated range. In order to demonstrate this phenomenon, we consider a normalized interval of the above Runge problem by interpolating the following function in the

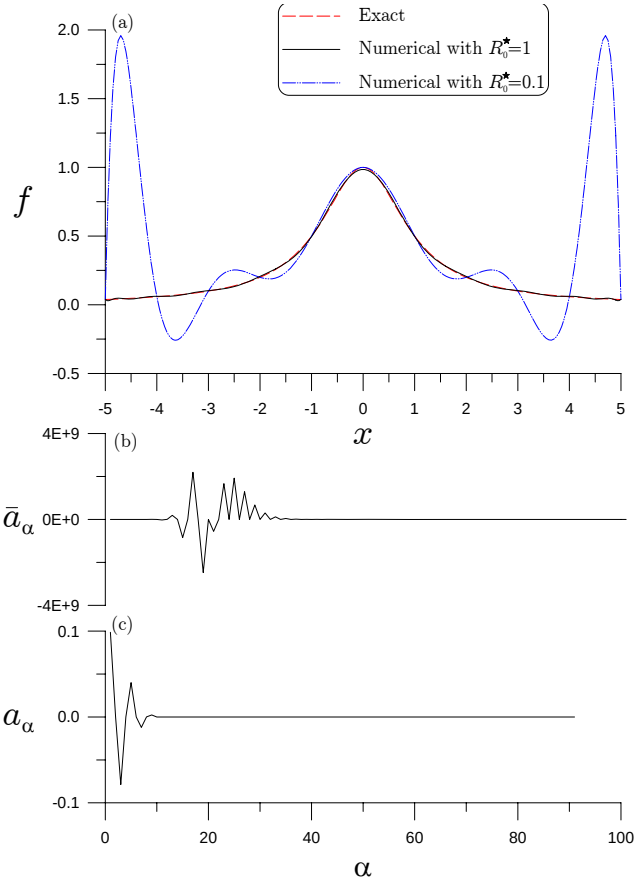

Figure 3: (a) Comparing the numerical and exact solutions of a numerical example due to Runge, and plotting the coefficients of (b) $\bar{a}_\alpha$ and (c) a_α.

interval of $[-1,1]$:

$$f(x) = \frac{1}{1+25x^2}, \quad x \in [-1,1], \tag{14}$$

where the number 25 appears due to a rescaling of the intervals from $[-5,5]$ to $[-1,1]$.

As shown in Table 1, the condition number of the corresponding Vandermonde matrix under an assumption of equidistantly interpolated locations in $[-1,1]$ also increases rapidly with n. We apply the new interpolation technique to solve this problem by fixing $R_0^* = 0.5, 0.55, 0.6, 0.75$. In Fig. 5 we compare the exact function with the interpolated polynomials, by using $R_0^* = 0.55, 0.6, 0.75$, and even n is

Table 2: Comparing numerical results by different methods for the Runge example

x	$\frac{1}{1+x^2}$	present $(R_0^* = 0.8)$	present $(R_0^* = 1)$	$S(x)$	$L_{10}(x)$
0.3	0.91743	0.91750	0.91726	0.92754	0.94090
0.8	0.60796	0.61270	0.61660	0.62420	0.64316
1.3	0.37175	0.36700	0.36284	0.36133	0.31650
1.8	0.23585	0.24111	0.24229	0.23154	0.18878
2.3	0.15898	0.15434	0.15683	0.16115	0.24145
2.8	0.11312	0.11666	0.11182	0.11366	0.19837
3.3	0.08410	0.08121	0.08709	0.08426	-0.10832
3.8	0.06477	0.06828	0.06173	0.06556	-0.20130
4.3	0.05131	0.04630	0.05176	0.04842	0.88808
4.8	0.04160	0.03769	0.04652	0.03758	1.80438

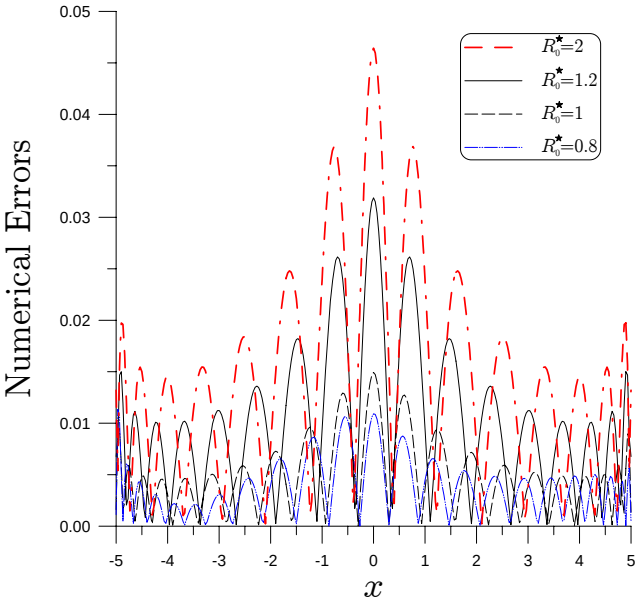

Figure 4: Comparing the numerical errors by using different R_0^* for the Runge example.

large up to 100, no oscillation is observed in the interpolants. However, the results using $R_0^* = 0.55$ and 0.6 have a little discrepancy at the two ends of the interval. For the case $R_0^* = 0.5$ as shown in Fig. 5 by the dashed line the resulting interpolant, as before, is oscillatory at two ends of the interval, and a large discrepancy from the

exact solution results. Thus, we can conclude that even for the interpolation problem in a normalized range $[-1,1]$, the choice of a suitable R_0^* in the interpolant can avoid an incorrect interpolated result, and increase the accuracy by using higher-order polynomials.

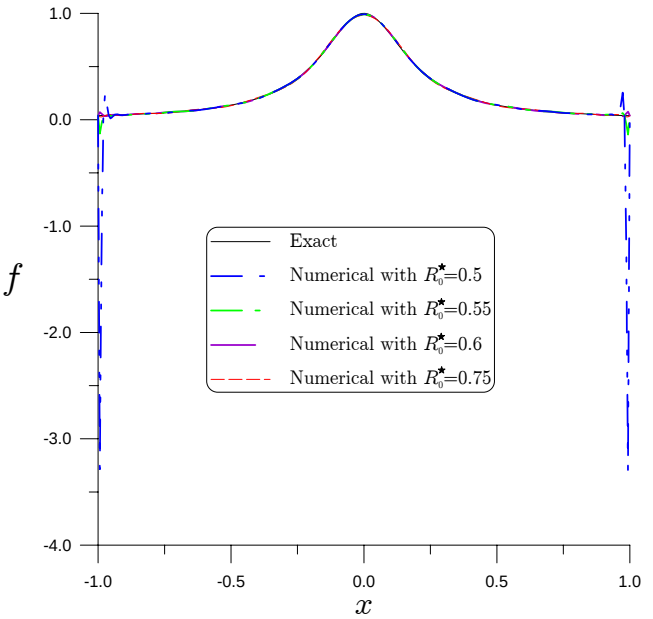

Figure 5: Comparing the numerical solutions under different R_0^* with the exact solution for the Runge problem in a normalized range.

For the above two cases the intervals are symmetric, and thus we only require that $R_0^* \geq 0.5$ for a stable numerical solution. In order to further investigate the influence of R_0^* on the accuracy of interpolation, we display the maximum errors with respect to δ, where $R_0^* = R_c + \delta$ in Fig. 6(a) for the function in Eq. (14), where we fix $n = 100$. It can be seen that at $\delta = 0.2$, i.e., $R_0^* = 0.7$, the maximum error obtained is minimal.

4.2 The derivatives of noisy data

In many applications it is necessary to calculate the derivatives of a function measured experimentally, i.e., to differentiate noisy data. The problem of numerical differentiation of noisy data is ill-posed: small changes of the data may result in a large change of the derivatives.

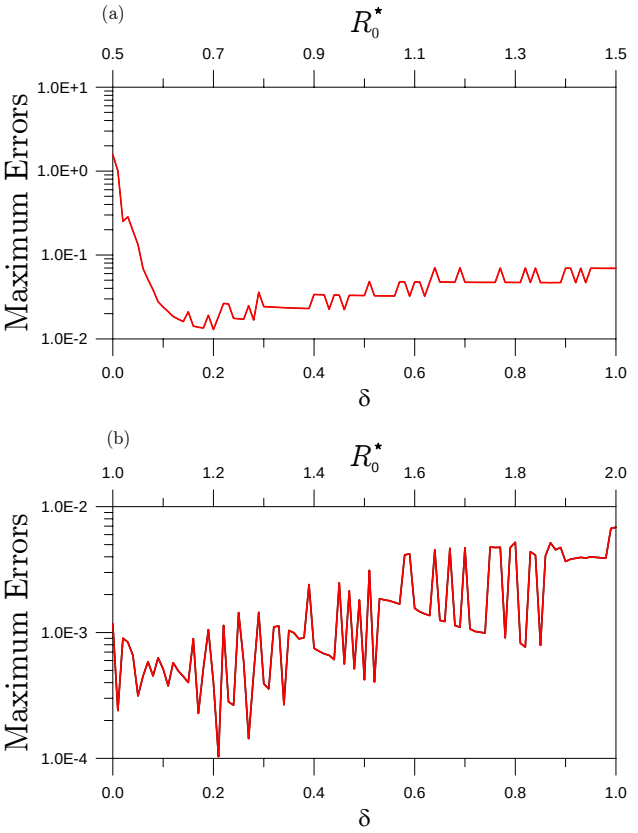

Figure 6: The variations of maximum errors with respect to δ and R_0^* for (a) the $f(x)$ in Eq. (14), and (b) the $f(x)$ in Eq. (15).

We consider the following functions and their derivatives:

$$\text{(a)}\ \ f(x)=\sin x,\ f'(x)=\cos x,\ f''(x)=-\sin x,\ \ 0\le x\le 4\pi, \tag{15}$$

$$\text{(b)}\ \ f(x)=\frac{1}{1+x},\ f'(x)=\frac{-1}{(1+x)^2},\ f''(x)=\frac{2}{(1+x)^3},\ \ 0\le x\le 10. \tag{16}$$

Under a noise of $\hat{f}_i=f_i+\sigma R(i)$ with $\sigma=0.01$, we apply the new interpolation technique to find the interpolant by

$$\hat{f}(x)=\sum_{\alpha=0}^{n}\bar{a}_\alpha\left(\frac{x}{R_0}\right)^\alpha, \tag{17}$$

and the derivatives by

$$\hat{f}'(x) = \sum_{\alpha=0}^{n} \frac{\alpha \bar{a}_\alpha}{R_0} \left(\frac{x}{R_0} \right)^{\alpha-1}, \tag{18}$$

$$\hat{f}''(x) = \sum_{\alpha=0}^{n} \frac{\alpha(\alpha-1) \bar{a}_\alpha}{R_0^2} \left(\frac{x}{R_0} \right)^{\alpha-2}. \tag{19}$$

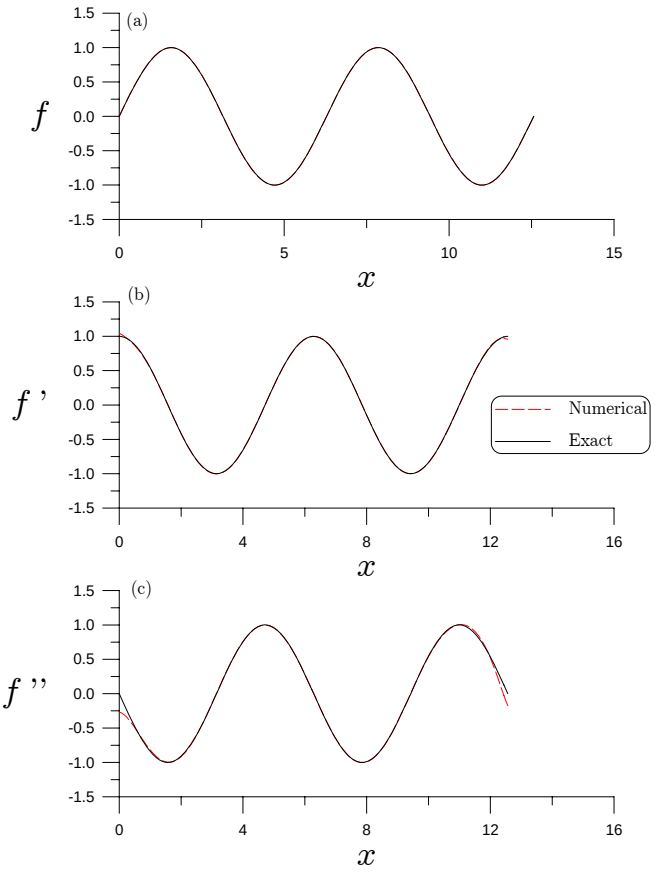

Figure 7: Comparing the numerical and exact solutions of a numerical example of case a for numerical derivatives of noisy data of $f(x) = \sin x$. [$n = 105$, $R_0^* = 19/(4\pi)$]

In Fig. 7 [for the problem in Eq. (15)] and Fig. 8 [for the problem in Eq. (16)] we compare the recovered functions of $\hat{f}(x)$, $\hat{f}'(x)$ and $\hat{f}''(x)$ with the exact ones. For

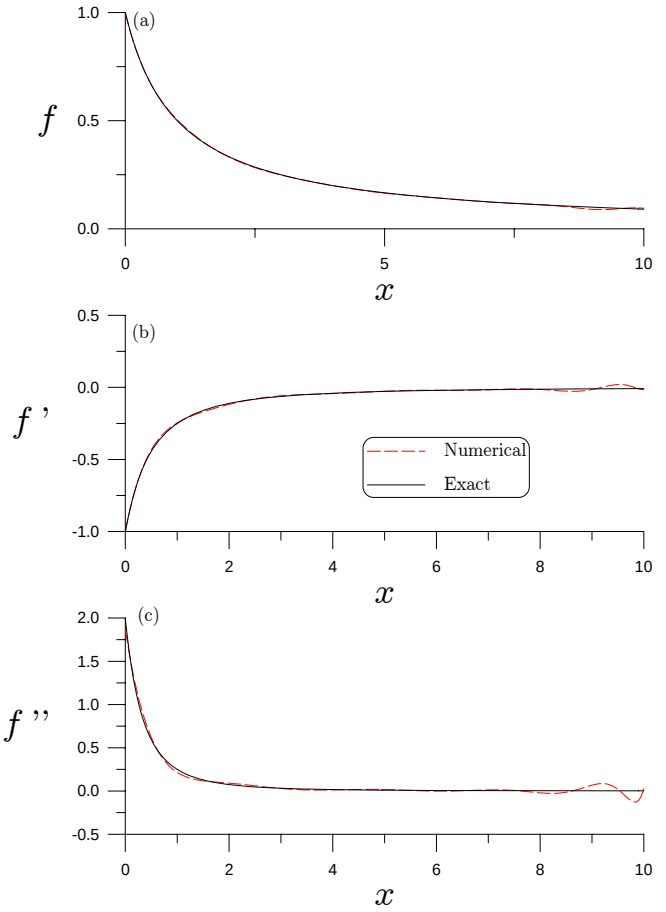

Figure 8: Comparing the numerical and exact solutions of a numerical example of case b for numerical derivatives of noisy data of $f(x) = 1/(1+x)$. [$n = 105$, $R_0^* = 1.5$]

case a (Fig. 7), we use $n = 105$ and $R_0^* = 19/(4\pi)$; and for case b (Fig. 8), we use $n = 105$ and $R_0^* = 1.5$. It can be seen that the present method is robust against the noise, and the numerical derivatives of noisy data can be calculated very accurately, even up to the second-order.

To test the effect of R_0^*, we perform the same calculations for the above two problems, with $R_0^* = 0.5$; however, n is restricted to be $n = 10$, because when n is larger, the numerical results are unsatisfactory due to numerical instability [for the avove two problems defined in the non-negative intervals, we require that $R_0^* \geq 1$, for

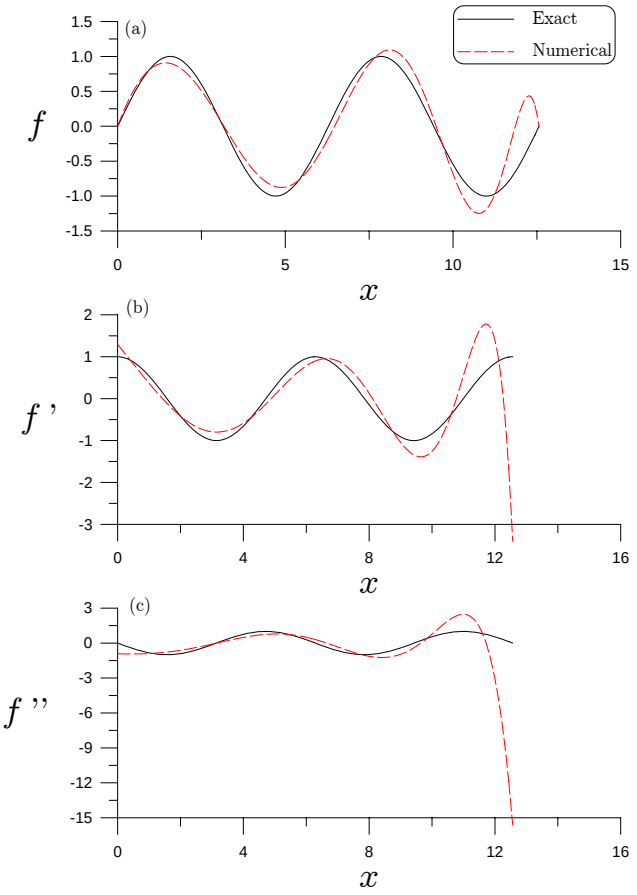

Figure 9: Comparing the numerical and exact solutions of case a with $R_0^* = 0.5$.

numerical stability]. In Fig. 9 [for the problem in Eq. (15)] and Fig. 10 [for the problem in Eq. (16)] we compare the recovered functions of $\hat{f}(x)$, $\hat{f}'(x)$ and $\hat{f}''(x)$ with the exact ones. Obviously, the numerical results are not accurate.

In order to investigate the influence of R_0^* on the accuracy of interpolation in a non-negative interval, we display the maximum errors with respect to δ and $R_0^* = R_c + \delta$ in Fig. 6(b) for the function in Eq. (15), where we fix $n = 120$. It can be seen that near to the value of $\delta = 0.23$, i.e., $R_0^* = R_C + 0.23$, the maximum error obtained is minimal. High accuracy in the order of 10^{-4} can be gained, if one uses $R_0^* = 1.23$.

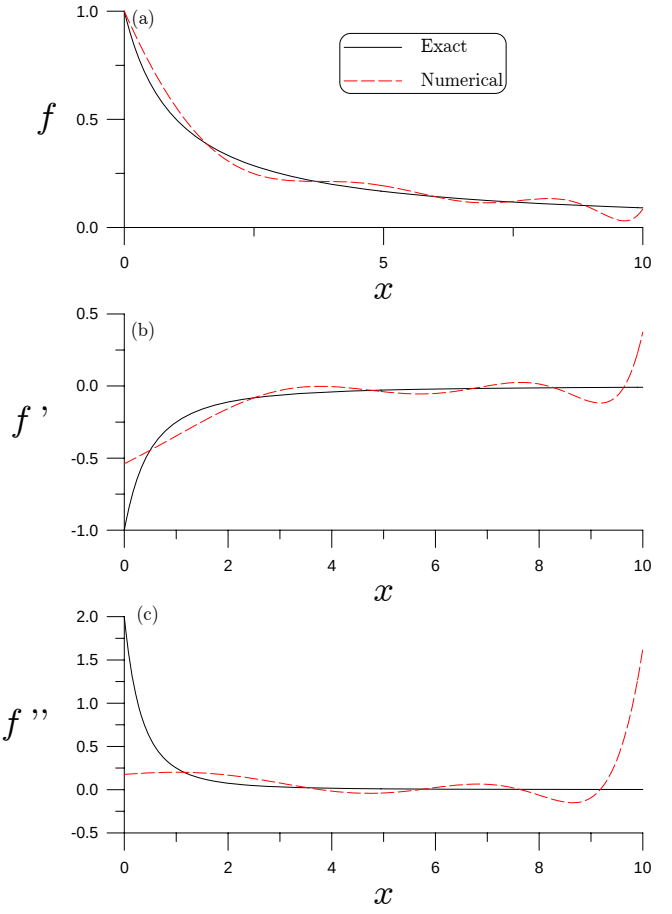

Figure 10: Comparing the numerical and exact solutions of case b with $R_0^* = 0.5$.

4.3 Crack propagation

As an application of the present numerical differentiation of noisy data, we consider the problem of crack propagation in order to estimate the crack propagation rate da/dN, where a is the measured crack-length and N is the number of load cycles [Broek (1982); Sih (1991)]. Theoretically, the crack propagation rate da/dN versus a has a power law relation: $da/dN = ca^{\beta}$. We suppose that the measured data of a are scattered along an unknown curve as shown in Fig. 11(a) with

$$\hat{a}_i = a(N_i) + \sigma R(i), \tag{20}$$

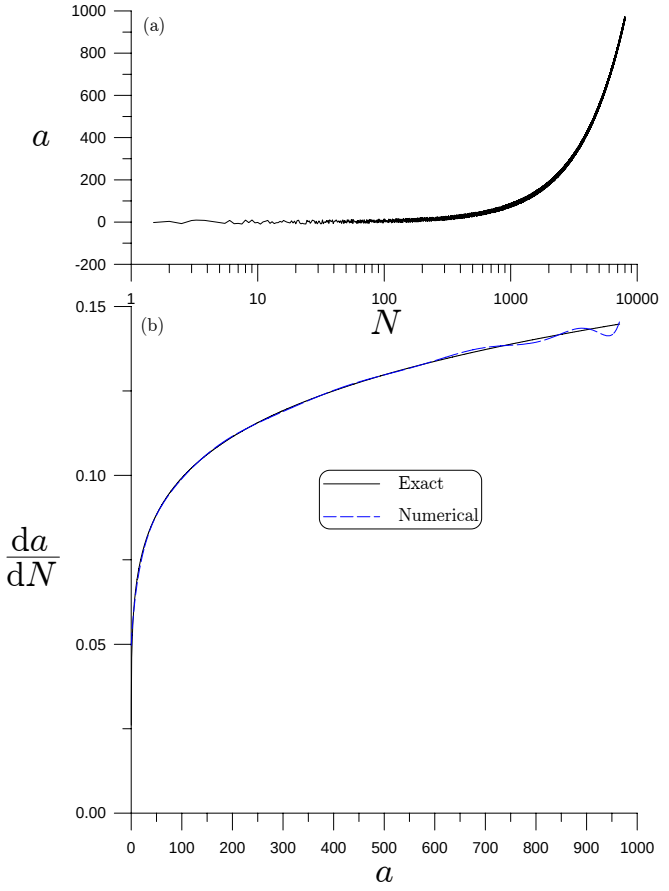

Figure 11: Comparison of the numerical and exact solutions for obtaining da/dN versus a from the measured noisy data for crack-length a versus the number of cycles, N.

where, for definiteness, and for the purpose of comparison, we take $a(N) = 0.02N^{1.2}$ and $\sigma = 1$. Usually, it is very difficult to estimate the rate da/dN versus a by using the scattered noisy data. However, we apply the new interpolation technique to solve this problem by using $n = 100$ and $R_0^* = 1.5$. We can calculate da/dN versus a at the measured points, and the results are plotted in Fig. 11(b) by the dashed line. It can be seen that the estimated rate is very close to the exact one of $da/dN = ca^\beta$, where $\beta = 1/6$ and $c = 0.024/0.02^\beta$.

4.4 Abel integral equation

Next, we consider the Abel integral equation (1), for the following case: $\eta = 1/3$, $\phi(s) = 10s/9$, $h(s) = s^{5/3}$, $0 < s \le 1$. Let

$$f(s_i) = \int_0^{s_i} \frac{\hat{h}(t)}{(s_i - t)^{1-\eta}} dt \tag{21}$$

be the discretized data of $f(s)$, and $\hat{h}(t_i) = h(t_i) + \sigma R(i)$ be the discretized data of $\hat{h}(t)$; such that, through some calculations, we have $f(s_i) = 5\pi s_i^2/[9\sin(\eta\pi)] + s_i^{\eta}\sigma R(i)/\eta$.

The numerical solution of the Abel equation is obtained by

$$\phi(s_i) = \frac{\sin(\eta\pi)}{\pi} f'(s_i), \tag{22}$$

where we apply the new interpolation method to calculate $f'(s_i)$. We use $n = 25$ and $R_0^* = 1.5$ in this calculation, and the convergence criterion used in the CGM is 10^{-5}. Even under a large noise with $\sigma = 0.01$, the numerical result as shown in Fig. 12 by the dashed line is very close to the exact solution. For the purpose of comparisons we also used $R_0^* = 0.5$ and $R_0^* = 2.5$ in the calculations. The accuracy obtained by using $R_0^* = 2.5$ is the same as that using $R_0^* = 1.5$. For the case with $R_0^* = 0.5$ we were forced to reduce $n = 25$ to $n = 10$; otherwise, the numerical instability appears. It can be seen that the error induced by $R_0^* = 0.5$ is much larger than the above two numerical solutions.

4.5 Inverse Laplace transform

For a given function $f(t)$ the Laplace transform is given by

$$\int_0^{\infty} e^{-st} f(t) dt = F(s). \tag{23}$$

The problem of inverse Laplace transform is that of finding $f(t)$ for a given function $F(s)$. This problem is known to be highly ill-posed.

By applying the new interpolation technique to this problem, we let

$$f(t) = \sum_{j=1}^{n} c_j \left(\frac{t}{T_0} \right)^{j-1} \tag{24}$$

to be a new polynomial interpolant of $f(t)$, where the coefficients c_i are to be determined from the given data of $F(s)$, and T_0 is a characteristic time length.

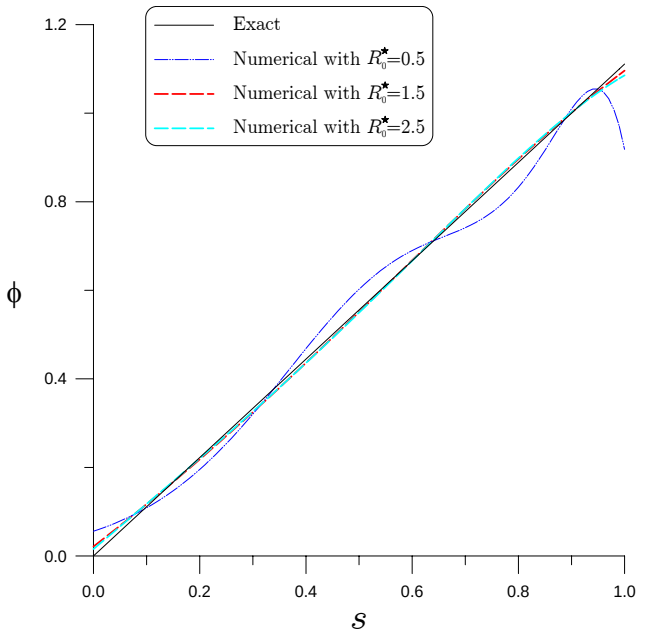

Figure 12: Comparing the numerical and exact solutions of an Abel integral equation under noise for different R_0^*.

Applying the Laplace transform to the above equation and using

$$\int_0^\infty e^{-st} t^{j-1} dt = \frac{(j-1)!}{s^j},$$

we can derive from Eq. (24) that:

$$c_1 + \sum_{j=2}^{n} \frac{(j-1)! c_j}{(T_0 s)^{j-1}} = sF(s). \tag{25}$$

For the given data of $F(s)$, when we take $s_i \in [a,b],\ i = 1,\ldots,n$ to be the sampling points, it leads to

$$\begin{bmatrix} 1 & \frac{1}{T_0 s_1} & \frac{2}{(T_0 s_1)^2} & \cdots & \frac{(n-2)!}{(T_0 s_1)^{n-2}} & \frac{(n-1)!}{(T_0 s_1)^{n-1}} \\ 1 & \frac{1}{T_0 s_2} & \frac{2}{(T_0 s_2)^2} & \cdots & \frac{(n-2)!}{(T_0 s_2)^{n-2}} & \frac{(n-1)!}{(T_0 s_2)^{n-1}} \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ 1 & \frac{1}{T_0 s_{n-1}} & \frac{2}{(T_0 s_{n-1})^2} & \cdots & \frac{(n-2)!}{(T_0 s_{n-1})^{n-2}} & \frac{(n-1)!}{(T_0 s_{n-1})^{n-1}} \\ 1 & \frac{1}{T_0 s_n} & \frac{2}{(T_0 s_n)^2} & \cdots & \frac{(n-2)!}{(T_0 s_n)^{n-2}} & \frac{(n-1)!}{(T_0 s_n)^{n-1}} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n-1} \\ c_n \end{bmatrix} = \begin{bmatrix} s_1 F(s_1) \\ s_2 F(s_2) \\ \vdots \\ s_{n-1} F(s_{n-1}) \\ s_n F(s_n) \end{bmatrix}.$$

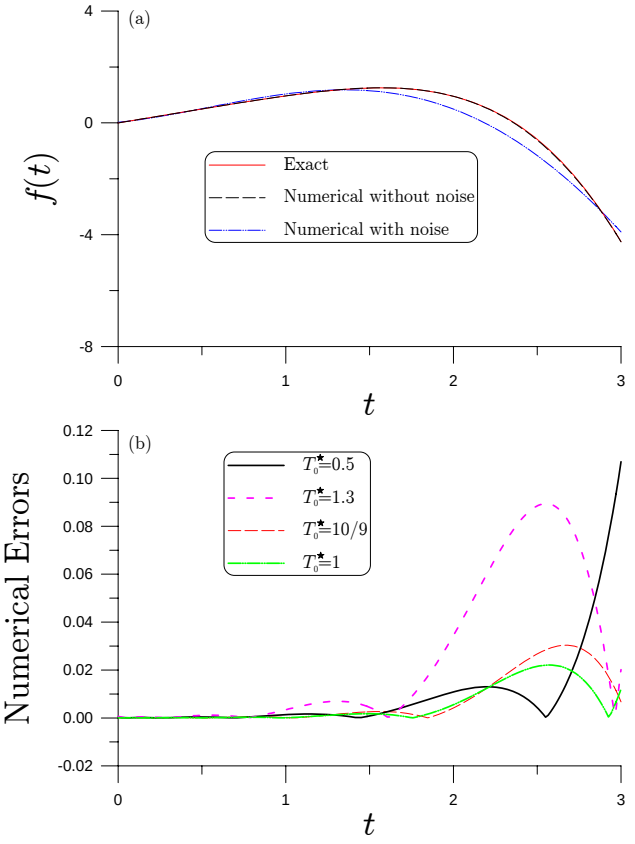

Figure 13: Comparing the results of inverse Laplace transform without noise and with noise to the exact solution for different T_0^*.

(26)

As before we can apply the CGM to solve the above equation obtaining the coefficients c_i.

In the example presented here, the function recovered by the inverse Laplace transfom is $f(t) = (\cosh t \sin t + \sinh t \cos t)/2$, and the data given are $F(s) = s^2/(s^4+4)$. Let $[a,b] = [1,10]$ and $s_i = 1 + 9i/n$. For the case without considering noise, we use $n = 20$ and $T_0^* = T_0/(b-a) = 10/9$ in the calculation, and the convergence criterion used in the CGM is 10^{-12}. The numerical result as shown in Fig. 13 by the dashed line is very close to the exact solution, with error in the order of 10^{-2}. Even under a large noise with $\sigma = 0.001$, the numerical result as shown in Fig. 13 by the dashed-dotted line is near to the exact solution, where we use $n = 25$ and

$T_0^* = 10/9$ in the calculation, and the convergent criterion of the CGM is 10^{-8}. We also considered the cases $T_0^* = 0.5$, $T_0^* = 1$ and $T_0^* = 1.3$ in the calculations. The accuracy obtained by using $T_0^* = 1$ is slightly better than that used $T_0^* = 10/9$. It can be seen that the errors induced by $T_0^* = 0.5$ and $T_0^* = 1.3$ are larger than the above two numerical solutions.

5 Conclusions

In this paper we have proposed a new polynomial interpolation technique in a one-dimensional interval, $a \le x \le b$, by expanding the polynomial in terms of x/R_0 rather than x. We have defined a dimensionless quantity $R_0^* = R_0/(b-a)$ to delineate the range of values for R_0^*, for numerical stability. Critical values R_c are given, such that when $R_0^* \ge R_c$, the numerical interpolation is stable for very-high-order polynomials. Instead of finding the coefficients a_α, we developed a novel method to find the coefficients $\bar{a}_\alpha$. The system of linear equations for a_α is highly ill-conditioned when $R_0^* < R_c$; conversely, when the characteristic length satisfies the criterion $R_0^* \ge R_c$, the condition number of coefficient-matrix is greatly reduced, and the new system is well-conditioned. From two examples investigated in this paper, we found that the use of the values of $R_0^* = R_c + \delta$, where $\delta \in [0.2, 0.23]$, produces the best accuracy for very high-order polynomial interpolation. We applied this new interpolation technique to solve the problem of numerical differentiation of noisy data [such as determining da/dN in fatigue mechanics], the problem of the inversion of the Abel integral equation under noise, and the problem of inverse Laplace transform under noise. Numerical examples showed that the present interpolation theory is robust against the noise. For the types of ill-posed problem considered in the present paper, the present approach is simple and effective in obtaining a reliable numerical solution, without resorting to regularization techniques.

References

Ahn, S.; Choi, U. J.; Ramm, A. G. (2006): A scheme for stable numerical differentiation. *J. Comp. Appl. Math.*, vol. 186, pp. 325-334.

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

Broek, D. (1982): Elementary Engineering Fracture Mechanics. Martinus Nijhoff Publishers, Boston.

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

Fettis, M. E. (1964): On numerical solution of equations of the Abel type. *Math. Comp.*, vol. 18, pp. 491-496.

Garza, J.; Hall, P.; Ruymagaart, F. H. (2001): A new method of solving noisy Abel-type equations. *J. Math. Anal. Appl.*, vol. 257, pp. 403-419.

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

Gautschi, W.; Inglese, G. (1988): Lower bounds for the condition number of Vandermonde matrices. *Numer. 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.

Gorenflo, R.; Vessella, S. (1991): Abel Integral Equations: Analysis and Applications. Springer-Verlag, Berlin.

Hall, P.; Paige, R.; Ruymagaart, F. H. (2003): Using wavelet methods to solve noisy Abel-type equations with discontinuous inputs. *J. Multivariate. Anal.*, vol. 86, pp. 72-96.

Hao, S. (1985): Application of the regularization method to the numerical solution of Abel's integral equation. *J. Comput. Math.*, vol. 3, pp. 27-34.

Hao, S. (1986): Application of the regularization method to the numerical solution of Abel's integral equation (II). *J. Comput. Math.*, vol. 4, pp. 322-328.

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.

Huang, L.; Huang, Y.; Li, X. F. (2008): Approximate solution of Abel integral equation. *Comp. Math. Appl.*, vol. 56, pp. 1748-1757.

Kosarev, E. L. (1973): The numerical solution of Abel's integral equation. *Zh. vychisl. Mat. mat. Fiz.*, vol. 13, pp. 1591-1596.

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. (2008): 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.; Atluri, S. N. (2009): 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.

Murio, D. A.; Hinestroza, D. G.; Mejia, C. W. (1992): New stable numerical inversion of Abel's integral equation. *Comput. Math. Appl.*, vol. 11, pp. 3-11.

Piessens, R. (2000): Computing integral transforms and solving integral equations using Chebyshev polynomial approximations. *J. Comput. Appl. Math.*, vol. 121, pp. 113-124.

Piessens, R.; Verbaeten, P. (1973): Numerical solution of the Abel integral equation. *BIT*, vol. 13, pp. 451-457.

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

Ramm, A. G.; Smirnova, A. B. (2001): On stable numerical differentiation. *Math. Comp.*, vol. 70, pp. 1131-1153.

Sih, G. C. (1991): Mechanics of Fracture Initiation and Propagation: Surface and Volume Energy Density Applied as Failure Criterion. Kluwer Academic Publishers, Netherlands.

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