SIAM J. Sci. Comput.
Vol. 33, No. 6, pp. 3241–3260

A FAST ALGORITHM FOR FOURIER CONTINUATION[*]

MARK LYON[†]

Abstract. A new algorithm is presented which provides a fast method for the computation of recently developed Fourier continuations (a particular type of Fourier extension method) that yield superalgebraically convergent Fourier series approximations of nonperiodic functions. Previously, the coefficients of an approximating Fourier series have been obtained by means of a regularized singular value decomposition (SVD)-based least-squares solution to an overdetermined linear system of equations. These SVD methods are effective when the size of the system does not become too large, but they quickly become unwieldy as the number of unknowns in the system grows. We demonstrate a novel decoupling of the least-squares problem which results in two systems of equations, one of which may be solved quickly by means of fast Fourier transforms (FFTs) and another that is demonstrated to be well approximated by a low-rank system. Utilizing randomized algorithms, the low-rank system is reduced to a significantly smaller system of equations. This new system is then efficiently solved with drastically reduced computational cost and memory requirements while still benefiting from the advantages of using a regularized SVD. The computational cost of the new algorithm in on the order of the cost of a single FFT multiplied by a slowly increasing factor that grows only logarithmically with the size of the system.

Key words. Fourier series, continuation method, Fourier extension, randomized algorithm

AMS subject classifications. 42A10, 65T40, 65T50

DOI. 10.1137/11082436X

1. Introduction.

We present a new and efficient algorithm for the computation of certain highly accurate approximations to nonperiodic functions by Fourier series. There has been a significant amount of research on this topic, in particular on the resolution of the well-known Gibbs phenomenon that results when standard Fourier series methods are applied to nonperiodic functions. A jump discontinuity is imposed in the approximation at the domain boundaries which gives rise to an adverse oscillation in the accuracy of the Fourier series near this artificial singularity. A variety of approaches have been introduced to mitigate the ill effects of the Gibbs phenomenon. Noteworthy early results include [22, 28, 29], which used smoothing, filtering, or alternate quadrature rules to obtain greater accuracy in the Fourier series approximations away from the singularity created by the jump discontinuity. More recent results have provided approximations with increased accuracy throughout the entire domain of interest. We mention the well-known methods based on Gegenbauer polynomials [20, 21, 32] and those utilizing Padé approximations [10, 19] in this latter category.

Our focus here is on a specific approach aimed at creating an approximating Fourier series with a period that is larger than the domain of function or data we wish to approximate. Broadly speaking, our objective is a special case of Whitney's extension problem [24, 35], which, in a significantly more general setting, has motivated significant advances in approximation theory (see, e.g., [14, 15] and the included references). We will deal with a specific extension technique termed "Fourier exten-

[*] Submitted to the journal's Methods and Algorithms for Scientific Computing section February 14, 2011; accepted for publication (in revised form) August 25, 2011; published electronically November 10, 2011. This work was supported by the National Science Foundation.
http://www.siam.org/journals/sisc/33-6/82436.html

[†] Department of Mathematics and Statistics, University of New Hampshire, Durham, NH 03861 (mark.lyon@unh.edu).

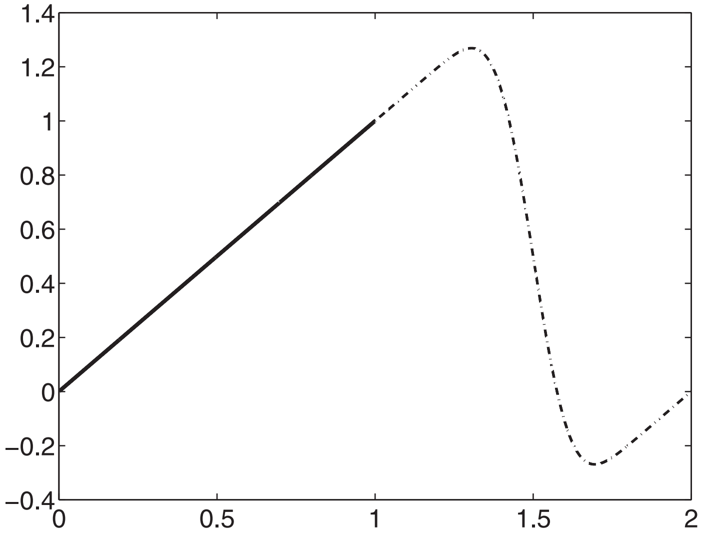

FIG. 1.1. *The function $f(x) = x$ (solid) on the interval* $[0, 1]$ *and a periodic Fourier continuation approximation to $f(x) = x$ (dashed) on the interval* $[0, 2]$.

sion of the third kind" in [3], "Fourier extension" in [4, 25], and "Fourier continuation" in [1, 5, 6, 7, 27]. The effect of Fourier continuation is demonstrated in Figure 1.1 for the function $f(x) = x$. The figure shows the function $f(x) = x$ on the interval $[0, 1]$ and a periodic continuation of the function on the interval $[0, 2]$ obtained by the method of [3, 4, 5, 6]. These continuations have been shown to produce super-algebraically convergent ([25] additionally gives conditions for an exponential rate of convergence) Fourier series approximations throughout the domain of original function, provided the function is sufficiently smooth. Fourier continuations have been successful in a variety of contexts, including multidimensional applications with unevenly spaced points (see, e.g., [6]). From Figure 1.1, it is clear that the these Fourier continuations are not analytic continuations and no connection to analyticity is meant to be implied by our terminology.

In [3, 4, 6], the Fourier coefficients of the continuation are calculated by solving a severely ill-conditioned least-squares system (the source of the ill-conditioning is discussed in section 2.1). In order to achieve accurate solutions, a regularized singular value decomposition (SVD) is used. Singular values below a cut-off, ϵ, which is typically near machine precision (cut-offs from 10^{-12} to 10^{-14} were studied in [3]), are neglected, and the resulting truncated system is solved to obtain the Fourier coefficients. Typically the number of Fourier modes (denoted here by M) that are computed is held to no more than half the number of points (denoted here by N) where the values of the function to be approximated are known. The function is typically split into even and odd symmetric parts, and the resulting decoupling of the equations for the sine and cosine series coefficients reduces the overall computational cost by a factor of four. Ultimately though, the $\mathcal{O}(NM^2)$ cost of the SVD for M Fourier modes at N spatial points is prohibitive for computations with large values of N and M.

It has been observed (see, e.g., [6, 25]) that due to the ill-conditioning of the system, the calculated coefficient can vary greatly with only insignificant changes in the accuracy of the approximation and that the Fourier continuation procedure may result in very large coefficients relative to the size of the data. As established in [3, 4, 5, 6, 25], this ill-conditioning does not adversely affect the accuracy of the Fourier approximations. In [6], a Chebyshev reprojection method is introduced where the Fourier continuation is calculated and then subsequently projected onto a Chebyshev basis. This is shown to be an effective means of stabilizing standard Fourier continuations at a relatively insignificant additional cost.

Two notable variants of Fourier continuation methods for calculating Fourier approximations to nonperiodic functions are also noted here. In [25], it is proposed to use a quadrature-based calculation of certain coefficients of unbounded orthogonal functions rather than calculate the Fourier coefficients. Due to the orthogonality of the alternate basis, the computational cost is reduced to $\mathcal{O}(NM)$, which significantly reduces the cost of the full SVD methods above, yet is still asymptotically slower than the approach we present here.

In [7, 27] an alternate continuation method, FC(Gram), is presented that relies upon a limited number of precomputed continuations to certain orthogonal polynomials. The computational cost of this approach is essentially equivalent to the cost of a single fast Fourier transform (FFT), $\mathcal{O}(N\log(N))$, and is shown to be efficient and effective in the computation of solutions to partial differential equations (PDEs) on complex domains. The approach was extended in [1] and applied to the compressible Navier–Stokes equations in general spatial domains. In this technique, the order of convergence corresponds to the degree of the polynomials continued, and therefore the resulting algorithm is not superalgebraically convergent, as the Fourier continuations we are concerned with here are. Further, the accelerated methods in [7, 27] actually require that, with N spatial points, slightly more than N Fourier modes be used in the approximation. Significant error can be introduced if Fourier coefficients calculated by the FC(Gram) algorithm are arbitrarily removed or modified. Thus, while the method of [7, 27] is efficient for solving of PDEs and in other applications, it is not as effective for applications such as data compression or the Fourier filtering of noisy data (see section 3.1).

In section 2, a new accelerated algorithm is introduced for the computation of the superalgebraically convergent SVD-based Fourier continuations referenced above that can be used in compression and filtering applications. At a computational cost of $\mathcal{O}(N\log(N)\log(M))$, the algorithm's computational cost is effectively equivalent to computing a relatively small number of FFTs. The new algorithm is based on a novel decoupling of the Fourier continuation problem, presented below, into separate systems of equations for the sine and cosine coefficients. This new decoupling results in a matrix whose singular values decay extremely fast, with only a very small number of the singular values greater than a given cut-off. The matrix can therefore be well approximated by a low-rank matrix, and special randomized algorithms have recently been introduced to compute decompositions for such low-rank systems (see, e.g., [8, 9, 17, 26, 36]). As stated in the literature, randomized methods are typically most effective when the underlying matrix is sparse and well approximated by a low-rank matrix. The sparsity allows the matrix-vector products required by the randomized algorithms to be efficiently computed. In the present case, the matrix is not sparse, but the matrix-vector products are nonetheless efficient, as they can be computed with FFTs. Section 3 provides a variety of numerical results demonstrating the robustness, speed, and accuracy of our algorithm.

2. Accelerated algorithm.

The continuation problem, as developed in [3, 5, 6], is a "least-squares trigonometric interpolation." Following the notation and definitions of [6], let $f(x)$ be a smooth and sufficiently differentiable function over a bounded domain. For concreteness of the presentation, we take our domain here to be the unit interval. Let x_j be a sequence of N points in that interval and define $y_j = f(x_j)$ for j from 0 to $N-1$. With the exception of the remarks in section 2.7, we restrict ourselves to the evenly spaced discretization points

$$x_j = \frac{j}{N-1}, \quad j = 0, \dots, N-1. \tag{2.1}$$

The Fourier continuation is defined to be the Fourier series

$$f^c(x) = f^c_{N,M}(x) = \sum_{k \in t(M)} a_k e^{\frac{2\pi i}{b} kx} \tag{2.2}$$

with period b, where $t(M) = \{j \in \mathbb{N} : -(M-1)/2 \leq j \leq (M-1)/2\}$ for M odd and $t(M) = \{j \in \mathbb{N} : -M/2 \leq j \leq M/2\}$ for M even. The coefficients a_k in (2.2) are determined as the solution to the least-squares minimization problem

$$\min_{(a_k)} \sum_{j=1}^{N} \left| \sum_{k \in t(M)} a_k e^{\frac{2\pi i}{b} kx_j} - y_j \right|^2 . \tag{2.3}$$

After study in [6], it was concluded that setting $b = 2$ provided what appeared to be the best balance between accurate and robust results, and this value has been used in [3, 4, 5, 25] as well. Here we will use $b = 2$ and develop our algorithm specifically for that case. We note, however, that, if required, a domain mapping could be applied to the results of our algorithm to generate continuations for arbitrary values of $b > 1$.

Typically these Fourier continuation least-squares systems are solved through the application of an SVD (see [33] for a thorough description of the SVD) due to the need to regularize the inherently ill-conditioned least-squares system. The regularization is accomplished by neglecting SVD modes corresponding to singular values less than a given cut-off, ϵ. Various values for ϵ have been used in the literature with optimal values typically ranging from 10^{-12} to just above 10^{-15} (see [3]). The computational cost of the SVD is $\mathcal{O}(NM^2)$ with a memory requirement of $\mathcal{O}(NM)$. As noted in [3, 4, 6], utilizing odd and even projections of the function to be continued can result in a certain decoupling of sine and cosines modes, which then reduces the computational cost by a factor of four.

In section 2.2, we show that an alternate decoupling (which we motivate in section 2.1) of the problem into sine and cosine modes yields two systems. One of these system can be solved directly using FFTs. The other system is well approximated by a low-rank system in that only $\mathcal{O}(\log(M))$ singular values are greater than a given ϵ near machine precision as shown in section 2.4. A reduced SVD (with computational cost of $\mathcal{O}(N \log(M)^2)$) can accurately generate an accurate least-squares solution to this system through a randomized algorithm, markedly decreasing the computational cost for large M as shown in section 2.5. Further, ideas allowing for the extension of the algorithm to multidimensional problems and unevenly spaced data are discussed in section 2.7. As the focus here is on a specific algorithm for Fourier continuation, we refer the reader to [3, 4, 5, 6, 25] for further theoretical and numerical results relating to the development of this technique.

2.1. Ill-conditioning and the full Fourier continuation. In Figure 2.1, normalized singular values are shown that result as part of the solution of (2.3) (without

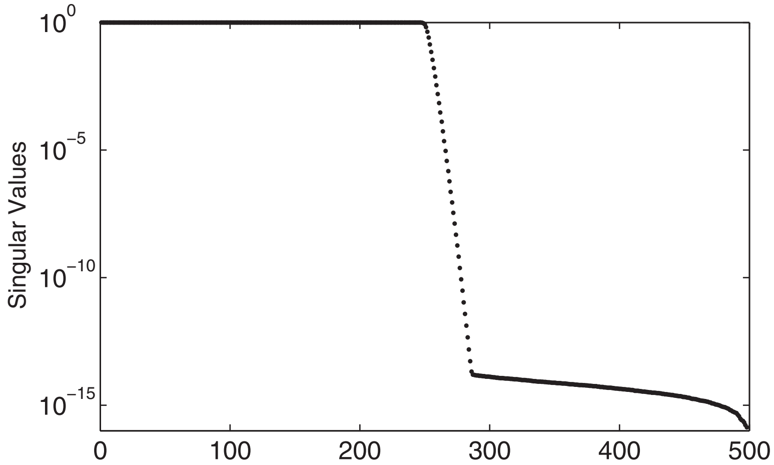

FIG. 2.1. *Normalized singular values that result as part of the solution of* (2.3) *using SVD techniques for $M = 500$, $b = 2$, and $N = 1000$.*

any decoupling) using SVD techniques for $M = 500$, $b = 2$, and $N = 1000$ in MATLAB. Similar results are obtained for other combinations of M and N: the first half of the singular values are uniform until the halfway point, where the singular values rapidly decay until the true singular values are masked by machine precision errors. The singular values demonstrate the ill-conditioning of the least-squares system. One way to view this ill-conditioning, as was done in [25], is to consider that the complex exponential system (2.3) actually contains elements of two different bases. Indeed the function $e^{\frac{-0.25}{x(1-x)}}$ (or this function multiplied by any $\rfloor^\infty$ function) could be very accurately approximated by either just the sine modes or just the cosine modes that are embedded in the complex exponentials of (2.3). It follows that the coefficients a_k in (2.3) used to approximate the function $e^{\frac{-0.25}{x(1-x)}}$ could then be altered drastically without a measurable change in accuracy of the approximation which is inherently the cause of the ill-conditioning.

If just the cosine modes were used in our approximation, the resulting singular values would be essentially the first half of the singular values in Figure 2.1. If just the sine modes were used, again the resulting singular values would be the first half of the singular values of the figure. When the sine and cosine modes are used together, it adds only a small number of additional singular values greater than any ϵ (that is sufficiently larger than machine precision) to what we would have with either the sines or the cosines alone. In what follows, instead of using symmetries to split the sine and cosine modes, we employ a decoupling such that one system captures the first half of the singular values of Figure 2.1 and can be solved by FFT methods. The second system will capture only the rapidly decaying and masked singular values. This latter system can then be well approximated by a low-rank matrix, which allows randomized algorithms to be effective in massively reducing the computational cost of solving this system.

2.2. Decoupling into cosine and sine modes. In order to give a precise description of the new algorithm, we introduce some further useful notation. For the clarity of the presentation, we will assume that the value of M is even so that the same number of cosine and sine modes is used in the Fourier continuation. The method for odd values of M is analogous and straightforward to develop from the presentation below, as it requires only using one less sine mode or cosine mode in the approximation.

We will denote $\vec{y} = (y_0, y_1, \ldots, y_{N-1})^T$ as the discrete values of the function we wish to approximate ($y_j = f(x_j)$) at the points x_j as defined in (2.1). The discrete cosine transform (DCT) matrix (i.e., DCT-1; see [31]) $\boldsymbol{C}$ is given by

$$\boldsymbol{C}_{jk} = \left(\sqrt{\frac{2}{N-1}}\right) w_j w_k \cos(\pi k x_j), \qquad \begin{array}{l} j = 0, 1, \ldots, N-1, \\ k = 0, 1, \ldots, N-1, \end{array} \tag{2.4}$$

with

$$w_j = \begin{cases} 1/\sqrt{2}, & j = 0, N-1, \\ 1, & j = 1, 2, \ldots, N-2. \end{cases} \tag{2.5}$$

According to this definition of the matrix, $\boldsymbol{C}$ is orthogonal and $\boldsymbol{C}^{-1} = \boldsymbol{C}^T$, where $\boldsymbol{C}^{-1}$ denotes the inverse of $\boldsymbol{C}$ and $\boldsymbol{C}^T$ denotes the transpose of $\boldsymbol{C}$. Further, let $\boldsymbol{C}^m$ be the first $M/2$ columns of $\boldsymbol{C}$. For a partial discrete sine transform (DST), define the matrix $\boldsymbol{S}^m$ by

$$\boldsymbol{S}^m_{jk} = \left(\sqrt{\frac{2}{N-1}}\right) \sin(\pi(k+1)x_j), \qquad \begin{array}{l} j = 0, 1, \ldots, N-1, \\ k = 0, 1, \ldots, M/2-1. \end{array} \tag{2.6}$$

As defined above, the column vectors of $\boldsymbol{S}^m$ are orthogonal to each other but not to the columns of $\boldsymbol{C}$ or $\boldsymbol{C}^m$.

A Fourier series, with a period twice as large as the original domain of the function, will serve as our Fourier continuation approximation. This series can be written as

$$f^c(x) = \left(\sqrt{\frac{2}{N-1}}\right) \left(\sum_{k=0}^{M/2-1} w_j w_k a_k \cos(\pi k x) + \sum_{k=0}^{M/2-1} b_k \sin(\pi(k+1)x) \right), \tag{2.7}$$

and the coefficients a_k and b_k for $k = 0, 1, \ldots, M/2-1$ can be determined as the least-squares solution to a system of N equations and M unknowns,

$$\boldsymbol{C}^m \vec{a} + \boldsymbol{S}^m \vec{b} \approx \vec{y}, \tag{2.8}$$

where the M unknowns are the $M/2$ cosine coefficients, $\vec{a} = (a_0, a_1, \ldots, a_{M/2-1})^T$, and the $M/2$ sine coefficients, $\vec{b} = (b_0, b_1, \ldots, b_{M/2-1})^T$. The coefficients $\vec{a}$ and $\vec{b}$ are then solutions to the minimization

$$\min_{\vec{a}, \vec{b}} \left\| \boldsymbol{C}^m \vec{a} + \boldsymbol{S}^m \vec{b} - \vec{y} \right\|_{\ell^2} \tag{2.9}$$

with the ℓ^2-norm of an N-length vector defined as

$$\|\vec{v}\|_{\ell^2} = \sqrt{\sum_{j=0}^{N-1} v_j^2}. \tag{2.10}$$

To develop the decoupled problem, we introduce a discrete cosine low-pass filter, $\boldsymbol{F}^\ell$, by

$$\boldsymbol{F}^\ell = \boldsymbol{C} \boldsymbol{D} \boldsymbol{C}^T = \boldsymbol{C}^m \left(\boldsymbol{C}^m\right)^T, \tag{2.11}$$

where $\boldsymbol{D}$ is a diagonal matrix with all values equal to zero except for the values $\boldsymbol{D}_{kk}$ for $k = 0, \ldots, M/2-1$ where $\boldsymbol{D}_{kk} = 1$. The action of the matrix $\boldsymbol{F}^\ell$ on a vector is then to project it into the vector space that is spanned by the discrete sampling of the first $M/2$ cosine modes at the points x_j. A related high-pass filter, $\boldsymbol{F}^h$, is defined by

$$\boldsymbol{F}^h = \boldsymbol{I} - \boldsymbol{F}^\ell, \tag{2.12}$$

where $\boldsymbol{I}$ is the $N \times N$ identity matrix. The filter $\boldsymbol{F}^h$ then projects into the vector space that is orthogonal to the first $M/2$ cosine modes. Both $\boldsymbol{F}^\ell$ and $\boldsymbol{F}^h$ as defined above are orthogonal projectors (e.g., $\boldsymbol{F}^\ell\boldsymbol{F}^\ell = \boldsymbol{F}^\ell$ and $\boldsymbol{F}^h\boldsymbol{F}^h = \boldsymbol{F}^h$). By the definition in (2.11), it follows that

$$\boldsymbol{F}^\ell \boldsymbol{C}^m \vec{a} = \boldsymbol{C}^m \vec{a}, \tag{2.13}$$

since $\boldsymbol{C}^m\vec{a}$ is necessarily in the space spanned by the first $M/2$ cosine modes. Similarly, by the definition in (2.12), we have that

$$\boldsymbol{F}^h \boldsymbol{C}^m \vec{a} = \vec{0} \tag{2.14}$$

for any $M/2$-length vector $\vec{a}$ where $\vec{0}$ is the N-length vector of all zeroes.

Remark 2.1. The action of the matrices $\boldsymbol{C}^m$, $\boldsymbol{C}$, $\boldsymbol{S}^m$, their transposes, and consequently $\boldsymbol{F}^\ell$ and $\boldsymbol{F}^h$ on a given vector can be computed directly using FFTs without ever explicitly forming the matrices. The computational cost of a single FFT application is $\mathcal{O}(N\log(N))$ regardless of the value of N (including prime values of N; see [18]). Our algorithm is slightly more efficient when DCT and DST methods are used in place of full FFTs.

Multiplying (2.9) by $\boldsymbol{F}^\ell + \boldsymbol{F}^h$ (note that according to (2.12), $\boldsymbol{F}^\ell + \boldsymbol{F}^h = \boldsymbol{I}$) yields

$$\min_{\vec{a},\vec{b}} \left\| \boldsymbol{F}^\ell\boldsymbol{C}^m\vec{a} + \boldsymbol{F}^h\boldsymbol{C}^m\vec{a} + \boldsymbol{F}^\ell\boldsymbol{S}^m\vec{b} + \boldsymbol{F}^h\boldsymbol{S}^m\vec{b} - \boldsymbol{F}^\ell\vec{y} - \boldsymbol{F}^h\vec{y} \right\|_{\ell^2}, \tag{2.15}$$

which, by (2.13) and (2.14), reduces to

$$\min_{\vec{a},\vec{b}} \left\| \boldsymbol{C}^m\vec{a} + \boldsymbol{F}^\ell\boldsymbol{S}^m\vec{b} + \boldsymbol{F}^h\boldsymbol{S}^m\vec{b} - \boldsymbol{F}^\ell\vec{y} - \boldsymbol{F}^h\vec{y} \right\|_{\ell^2}. \tag{2.16}$$

Further, since the spaces spanned by the columns of the $\boldsymbol{F}^\ell$ and $\boldsymbol{C}^m$ are orthogonal to the columns of $\boldsymbol{F}^h$, (2.16) is equivalent to

$$\min_{\vec{a},\vec{b}} \left(\left\| \boldsymbol{C}^m\vec{a} + \boldsymbol{F}^\ell\boldsymbol{S}^m\vec{b} - \boldsymbol{F}^\ell\vec{y} \right\|_{\ell^2}^2 + \left\| \boldsymbol{F}^h\boldsymbol{S}^m\vec{b} - \boldsymbol{F}^h\vec{y} \right\|_{\ell^2}^2 \right), \tag{2.17}$$

where only the first norm in (2.17) depends on the cosine coefficients, $\vec{a}$. By direct substitution, it can be seen that setting

$$\vec{a} = (\boldsymbol{C}^m)^T \left(\vec{y} - \boldsymbol{S}^m\vec{b} \right) \tag{2.18}$$

minimizes the first norm by providing an exact and unique solution to

$$\boldsymbol{C}^m\vec{a} + \boldsymbol{F}^\ell\boldsymbol{S}^m\vec{b} = \boldsymbol{F}^\ell\vec{y} \tag{2.19}$$

for any vector $\vec{b}$. The minimization problem is then reduced to

$$\min_{\vec{b}} \left\| \boldsymbol{F}^h\boldsymbol{S}^m\vec{b} - \boldsymbol{F}^h\vec{y} \right\|_{\ell^2} \tag{2.20}$$

with the only unknowns being the sine series coefficients, $\vec{b}$.

2.3. Solving for the sine coefficients. We now let $\boldsymbol{U}\boldsymbol{S}\overline{\boldsymbol{V}}^T$ denote the reduced SVD (see [33]) of the matrix $\boldsymbol{F}^h\boldsymbol{S}^m$, where the matrix $\boldsymbol{V}$ is unitary, $\overline{\boldsymbol{V}}^T$ denotes the complex conjugate of the transpose of $\boldsymbol{V}$, the columns of $\boldsymbol{U}$ are orthogonal, and $\boldsymbol{S}$ is a nonnegative diagonal matrix with the singular values of the matrix $\boldsymbol{F}^h\boldsymbol{S}^m$ in decreasing order on the diagonal. Let $\boldsymbol{S}^\dagger$ be an $M/2 \times M/2$ regularized inverse defined to be zero everywhere except for all j such that $\boldsymbol{S}_{jj} > \epsilon$, where it has the value $\boldsymbol{S}^\dagger_{jj} = \frac{1}{\boldsymbol{S}_{jj}}$. The regularized SVD least-squares solution to the system

$$\boldsymbol{F}^h\boldsymbol{S}^m\vec{b} = \boldsymbol{U}\boldsymbol{S}\overline{\boldsymbol{V}}^T\vec{b} \approx \boldsymbol{F}^h\vec{y} \tag{2.21}$$

is then given by

$$\vec{b} = \boldsymbol{V}\tilde{\boldsymbol{S}}^{-1}\overline{\boldsymbol{U}}^T\boldsymbol{F}^h\vec{y}. \tag{2.22}$$

If we let $\tilde{R}$ denote the number of singular values of $\boldsymbol{S}$ that are greater than ϵ and define $\boldsymbol{\Lambda}$ be the $N \times \tilde{R}$ matrix composed of the first $\tilde{R}$ columns of $\boldsymbol{U}$, it follows by the orthogonality of the columns of $\boldsymbol{U}$ that the $\vec{b}$ obtained in (2.22) is also an exact solution to the reduced system

$$\overline{\boldsymbol{\Lambda}}^T\boldsymbol{F}^h\boldsymbol{S}^m\vec{b} = \overline{\boldsymbol{\Lambda}}^T\boldsymbol{F}^h\vec{y}. \tag{2.23}$$

If $\tilde{R}$ is small and $\boldsymbol{\Lambda}$ is known, the matrix $\overline{\boldsymbol{\Lambda}}^T\boldsymbol{F}^h\boldsymbol{S}^m$ can be efficiently computed by computing $\left(\boldsymbol{F}^h\boldsymbol{S}^m\right)^T\boldsymbol{\Lambda}$ at a computational cost of $\mathcal{O}(\tilde{R}N\log(N))$. The reduced system in (2.23) can then be solved by computing the SVD of $\overline{\boldsymbol{\Lambda}}^T\boldsymbol{F}^h\boldsymbol{S}^m$ at a cost of only $\mathcal{O}(M\tilde{R}^2)$ compared with the $\mathcal{O}(NM^2)$ cost of obtaining the SVD of $\boldsymbol{F}^h\boldsymbol{S}^m$. Thus if $\boldsymbol{\Lambda}$ is known and $\tilde{R}$ is small, the computational cost can be significantly reduced.

The matrix $\boldsymbol{\Lambda}$ in (2.23) could be replaced by any matrix with orthogonal columns that has a column space which contains the space spanned by the columns of $\boldsymbol{\Lambda}$ and still yield the same solution for the coefficients $\vec{b}$. The exact $\boldsymbol{\Lambda}$ cannot be known without calculating the full SVD, but an alternate matrix, whose columns span a space that essentially contains column space of $\boldsymbol{\Lambda}$, can be efficiently calculated using FFTs and randomized algorithms as detailed below. This will be efficient if $\tilde{R}$ is small, and in the next section, we show that $\tilde{R}$ grows very slowly as M increases (approximately as $4\log_2(M)$).

2.4. Singular values of the matrix $\boldsymbol{F}^h\boldsymbol{S}^m$. Extensive computational results, some of which are detailed here, clearly indicate that the number of singular values of the matrix $\boldsymbol{F}^h\boldsymbol{S}^m$ above a given cut-off, ϵ, grows only logarithmically as M increases. In Figures 2.2 and 2.3 the singular values of the matrix $\boldsymbol{F}^h\boldsymbol{S}^m$ were calculated in MATLAB for various values of M and N. The largest 75 calculated singular values are shown for the data pairs $\{M = 100, N = 200\}$, $\{M = 200, N = 400\}$, and $\{M = 400, N = 800\}$ in Figure 2.2 and the nearly identical results for the pairs $\{M = 100, N = 1200\}$, $\{M = 200, N = 1200\}$, and $\{M = 400, N = 1200\}$ in Figure 2.3. The singular values rapidly decay until a plateau is reached. This plateau is the result of the accumulation of round-off error in the SVD computation. For any ϵ above this artificial plateau, the number of singular values larger than ϵ is small and increases very slowly as M increases.

Figure 2.4 further shows the value of $\tilde{R}$, the number of singular values of $\boldsymbol{F}^h\boldsymbol{S}^m$ above a cut-off (the cut-off is set to be $\epsilon = 10^{-13}$ for the figure) as a function of M.

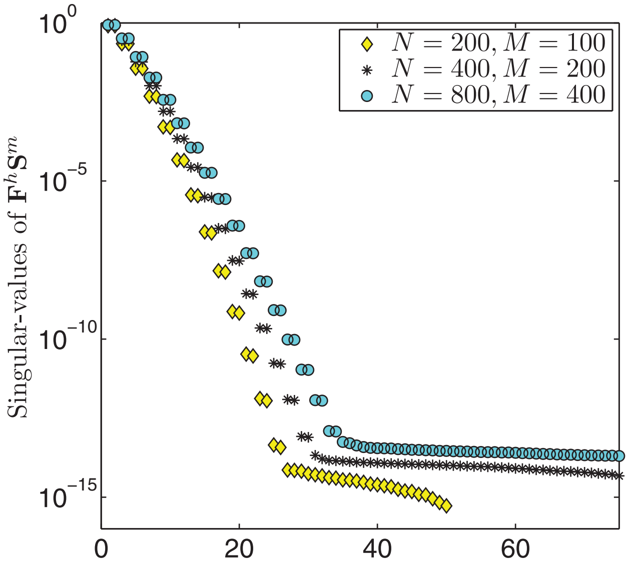

FIG. 2.2. *The largest* 75 *singular values of the matrix* $\boldsymbol{F}^h\boldsymbol{S}^m$ *for the three sets of values* $\{M = 100, N = 200\}$, $\{M = 200, N = 400\}$, *and* $\{M = 400, N = 800\}$. *The singular values of* $\boldsymbol{F}^h\boldsymbol{S}^m$ *are essentially in pairs and rapidly decay until they plateau due to the accumulation of round-off error in the SVD calculation. The number of singular values above a given cut-off, larger than the artificial round-off error plateau, grows only modestly as* M *increases.*

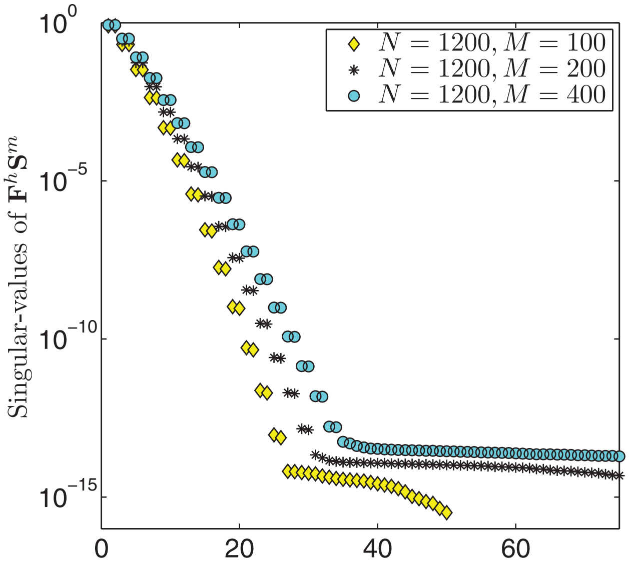

FIG. 2.3. *The largest* 75 *singular values of the matrix* $\boldsymbol{F}^h\boldsymbol{S}^m$ *for the three sets of values* $\{M = 100, N = 1200\}$, $\{M = 200, N = 1200\}$, *and* $\{M = 400, N = 1200\}$. *Results are very similar to those in Figure* 2.2 *for different values of* N.

$\tilde{R}$ was calculated at 12 logarithmically spaced values of M, with N fixed at 1500. It is easily seen in the figure that the growth of $\tilde{R}$ follows the function

$$\tilde{R} \approx 4\log_2(M) - 1. \tag{2.24}$$

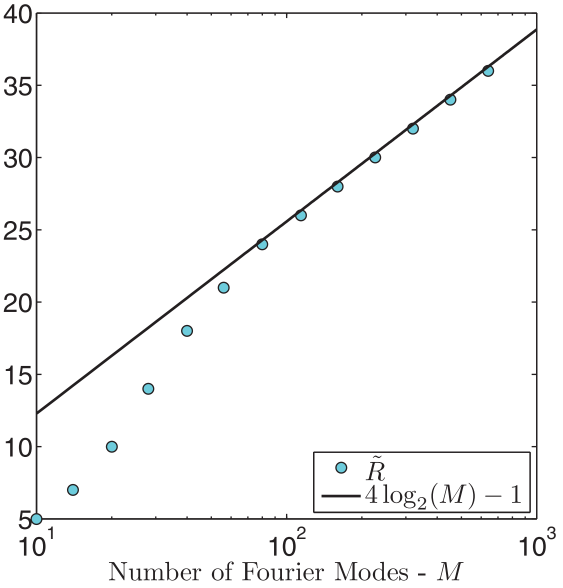

Fig. 2.4. *Calculated values of $\tilde{R}$, taken to be the number of singular values of $\boldsymbol{F}^h\boldsymbol{S}^m$ greater than a cut-off (here 10^{-13}), denoted by the gray circles and calculated for $N = 1500$ over a range of approximately logarithmically spaced values of M. A black line is also shown depicting the function $4\log_2(M) - 1$ which accurately captures the growth $\tilde{R}$ with M.*

Table 2.1

Calculated number, $\tilde{R}$, of singular values larger than 10^{-13} for various values of M and N. $\tilde{R}$ does not appear to change at all with N and depends entirely on the value of M whenever $M \leq N/2$.

	$N = 500$	$N = 750$	$N = 1000$	$N = 1500$
$M = 20$	10	10	10	10
$M = 40$	18	18	18	18
$M = 80$	24	24	24	24
$M = 160$	28	28	28	28
$M = 320$	32	32	32	32
$M = 640$	32	35	36	36

Many other computational results further demonstrate that the above approximation for $\tilde{R}$ holds for all systems for which the SVD could be calculated in reasonable time. Table 2.1 further demonstrates that the value of $\tilde{R}$ does not appear to depend on N whenever the condition $M \leq N/2$ is satisfied. When this condition, which is needed for the accuracy of the Fourier continuation (see [6]), is not satisfied, the calculated value of $\tilde{R}$ only decreases.

2.5. Reduced problem through randomization. Recently, there has been significant progress made in using randomized algorithms in order to quickly calculate matrix decompositions of low-rank approximations to matrices (see, e.g., [8, 9, 17, 26, 36]). Our work is motivated primarily by [26, 36], where it is shown that although randomized methods have a positive probability of failure, this probability can be made negligible (i.e., 10^{-17} in [26]). Our interest is in obtaining a solution to the ill-conditioned and effectively low-rank system (2.20) rather than in actually obtaining any given decomposition as is done in the citations, but our method develops similarly.

From the SVD of $\boldsymbol{F}^h\boldsymbol{S}^m$ and the definition of $\boldsymbol{\Lambda}$ above (2.23) it follows that for any vector $\vec{x}$,

$$\boldsymbol{F}^h\boldsymbol{S}^m\vec{x} = \vec{\omega} + \vec{z}, \tag{2.25}$$

where $\vec{\omega}$ is a linear combination of the columns of $\boldsymbol{\Lambda}$ and where the magnitude of $\vec{z}$ is small (i.e., $\|\vec{z}\|_{\ell^2} < \epsilon$). Let $\boldsymbol{W}$ be an $M/2 \times L$ matrix, with each entry of the matrix being an independent uniformly distributed random value from the the interval $[-1, 1]$. Each of the L columns of the product of $\boldsymbol{F}^h\boldsymbol{S}^m\boldsymbol{W}$ can be computed in $\mathcal{O}(N\log(N))$ operations through FFTs, and by (2.25) each column of $\boldsymbol{W}$ gives an opportunity to sample the space spanned by the columns of $\boldsymbol{\Lambda}$. Thus as L increases, the probability increases that the columns of $\boldsymbol{F}^h\boldsymbol{S}^m\boldsymbol{W}$ span a space containing the column space of $\boldsymbol{\Lambda}$.

Using an "economy" or "reduced" QR decomposition (again see [33]), matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ can be computed so that $\boldsymbol{QR} = \boldsymbol{F}^h\boldsymbol{S}^m\boldsymbol{W}$, where $\boldsymbol{Q}$ is an $N \times L$ matrix with orthonormal columns that span the same space as the columns of the product $\boldsymbol{F}^h\boldsymbol{S}^m\boldsymbol{W}$. For L large enough, the column span of $\boldsymbol{Q}$ will essentially contain the space spanned by the columns of $\boldsymbol{\Lambda}$. The Fourier continuation problem is then solved in a subspace span by the columns of $\boldsymbol{Q}$ which has been designed to sample the column space of $\boldsymbol{\Lambda}$. The least-squares problem is then projected into the column space of $\boldsymbol{Q}$ to drastically reduce the cost of the required computations. The value we use for L is based on the calculated value of $\tilde{R}$ in the previous section, the analysis of [26] and the cited references, and our computational results presented in section 3. Setting $L = [4\log_2(M)] + 20$ provides solid and robust numerical results, as we demonstrate in our computational results, where $[4\log_2(M)]$ is used to denote the closest integer to $4\log_2(M)$.

2.6. Algorithm summary. A concise statement of the algorithm follows for quick implementation of all of the steps. It is assumed that the discrete grid has N points. The order of the computational cost is given for each step in the algorithm. References to FFTs may be replaced by DCTs or DSTs as appropriate.

1. Generate $\boldsymbol{W}$, an $M/2 \times L$ matrix with each entry being an independent uniformly distributed random value from the the interval $[-1, 1]$, where M is number of Fourier modes to be used in the approximation and L is a parameter for which we recommend the value $[4\log_2(M)] + 20$. $\mathcal{O}(ML)$
2. The matrix product $\boldsymbol{F}^h\boldsymbol{S}^m\boldsymbol{W}$ is calculated by applying the matrices $\boldsymbol{S}^m$ and $\boldsymbol{F}^h$ (see (2.6), (2.11), and (2.12)) to the columns of $\boldsymbol{W}$ to generate the corresponding column of the matrix $\boldsymbol{F}^h\boldsymbol{S}^m\boldsymbol{W}$ using FFTs. $\mathcal{O}(LN\log(N))$
3. A reduced $\boldsymbol{QR}$ decomposition of the matrix $\boldsymbol{F}^h\boldsymbol{S}^m\boldsymbol{W}$ is computed. $\mathcal{O}(NL^2)$
4. The matrix $\boldsymbol{Q}^T\boldsymbol{F}^h\boldsymbol{S}^m$ is computed by applying the operators $(\boldsymbol{F}^h)^T$ and $(\boldsymbol{S}^m)^T$ to the columns of $\boldsymbol{Q}$ to generate the corresponding rows of $\boldsymbol{Q}^T\boldsymbol{F}^h\boldsymbol{S}^m$ using FFTs. $\mathcal{O}(LN\log(N))$
5. The reduced SVD of $\boldsymbol{Q}^T\boldsymbol{F}^h\boldsymbol{S}^m = \boldsymbol{U}\boldsymbol{S}\overline{\boldsymbol{V}}^T$ is then computed where the matrix $\boldsymbol{U}$ is $L \times L$, $\boldsymbol{S}$ is $L \times L$, and $\boldsymbol{V}$ is $M \times L$. $\mathcal{O}(ML^2)$
6. The $L \times L$ matrix $\boldsymbol{S}^\dagger$ is formed by setting all values to zero except for all j such that $\boldsymbol{S}_{jj} > \epsilon$, for which we set $\boldsymbol{S}^\dagger_{jj} = \frac{1}{\boldsymbol{S}_{jj}}$.
7. The sine coefficients are then given by $\vec{b} = \boldsymbol{V}(\boldsymbol{S}^\dagger(\overline{\boldsymbol{U}}^T(\boldsymbol{Q}^T\boldsymbol{F}^h\vec{y})))$. $\mathcal{O}(NL + N\log(N))$
8. The cosine coefficients are computed by $\vec{a} = (\boldsymbol{C}^m)^T(\vec{y} - \boldsymbol{S}^m\vec{b})$ (see (2.18)), which can be done with FFTs. $\mathcal{O}(N\log(N))$

Noting that M will always be less than N and if we set $L = [4\log_2(M)] + 20$, the total computational cost is $\mathcal{O}(N\log(N)\log(M))$, which is equivalent to performing $\mathcal{O}(\log(M))$ FFTs of length N.

Remark 2.2. We note that in the algorithm presented above, the cosine and sine modes were handled differently from one another. An alternate algorithm can easily be produced by switching sine and cosine modes in preceding steps.

2.7. Extensions to unevenly spaced data and higher dimensional problems. The method presented above has been developed for evenly spaced data but is not inherently limited to this case. Significant work has been accomplished in extending the regular evenly spaced FFT to unevenly spaced data including [2, 11, 12, 13, 16, 23, 30, 34]. These methods preserve the $\mathcal{O}(N\log(N))$ complexity of the FFT but with a larger constant that depends on the desired accuracy of the solution. Implementing the present algorithms on unevenly spaced data in one spatial dimension amounts to replacing the FFTs with the appropriate unevenly spaced FFTs (USFFTs).

In order to extend these results to multiple dimensions, one must account for the sensitivity of the coefficients $\vec{a}$ and $\vec{b}$ (see Remark 3.2) that results from the ill-conditioning of the system. This prohibits a naive extension of the Fourier continuation to higher dimensions to be performed following the standard approach of multidimensional FFTs for periodic functions. As mentioned in the introduction, a Chebyshev reprojection can be performed on top of the Fourier continuation as detailed in [6]. Once the coefficients $\vec{a}$ and $\vec{b}$ are determined, the approximation can be oversampled onto a Chebyshev grid (using an USFFT) at an insignificant additional cost. The Chebyshev coefficients, unlike the Fourier continuation coefficients, are stable with respect to the data as shown in [6]. Applying the Fourier continuation in multiple dimensions then follows in a manner similar to a multidimensional Chebyshev approximation, once the reprojection is incorporated into the continuation procedure. Extending these methods to the multidimensional approximation of unevenly distributed data would then involve a combination of the Chebyshev reprojection and the relevant algorithms for USFFTs and is left for future work.

3. Numerical results. The preceding randomized algorithm was implemented along with a comparison implementation that takes the full regularized SVD of $\boldsymbol{F}^h\boldsymbol{S}^m$ to solve the system (2.20) exactly as in (2.22). Figure 3.1 compares the computational cost of the two different algorithms where the full SVD cost is equivalent to the cost for the continuation algorithms in [3, 4, 5, 6]. All computations were performed in MATLAB on a desktop computer with an AMD Phenom-II processor running at 3.0 Ghz with 8.0 Gigabyts of RAM. In Figure 3.1, N is shown and M was set to be $N/2$ and we set $L = [4\log_2(M)] + 20$. The implementation involving only the regularized SVD is faster for small values of N to about 200–300, where the algorithms have similar computational costs. For larger values of N, the speed of the present approach quickly becomes apparent. For 10,000 points, the randomized algorithm is more than three orders of magnitude faster. For N of about 3.2 million, which is far too large to be realistically computed using the full SVD, the randomized algorithm is complete in less than two minutes. The results shown in the figure were implemented with full complex FFTs, and the overall computational time of the randomized algorithm could be reduced by a small factor using DCT and/or DST algorithms.

Computations were performed to compare the accuracy of the decomposition in the present approach with previously published results in [6], and results are shown in Table 3.1 for the computation of the Fourier continuation of $f(x) = x$ which is depicted in Figure 1.1. The differences in the accuracies are primarily due to the

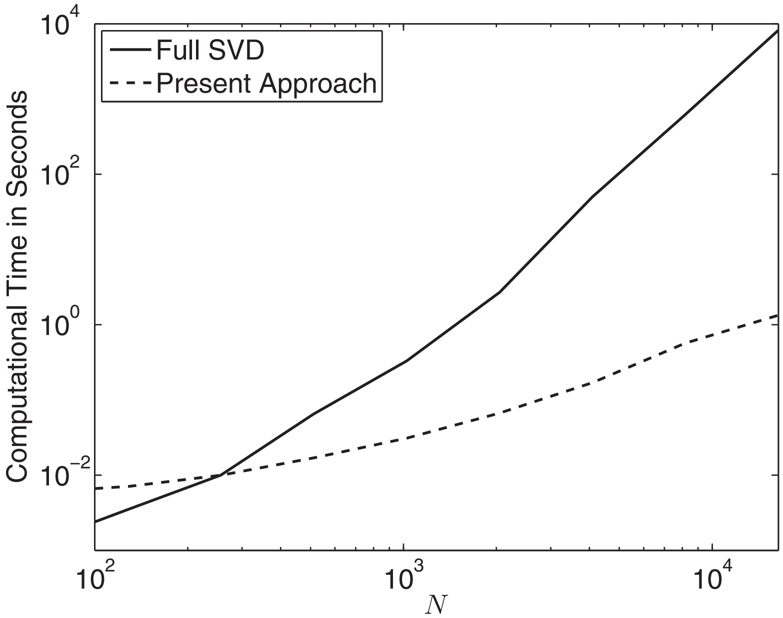

FIG. 3.1. *Comparison of the computational costs of the randomized algorithm presented here to using a full SVD for computing Fourier continuations. For the computations M was set at $N/2$ and $L = [4\log_2(M)] + 20$. The effects of the resulting $\mathcal{O}(N^3)$ computational cost of the full SVD and the $\mathcal{O}(N\log(N)^2)$ computational cost of the randomized algorithm are clearly demonstrated.*

TABLE 3.1

Approximation error evaluated as the maximum over 25,000 *points of the difference between the function $f(x) = x$ and its Fourier continuation approximation of the function with $\epsilon = 5\times 10^{-15}$ (see Remark* 3.1*). For comparison, the previously published Fourier continuation results presented in* [6] *are shown in parentheses below the present result.*

	$\lvert f(x) - f^c(x)\rvert$	$\left\lvert \frac{df(x)}{dx} - \frac{df^c(x)}{dx}\right\rvert$	$\left\lvert \frac{d^2f(x)}{dx^2} - \frac{d^2f^c(x)}{dx^2}\right\rvert$
$N = 8$, $M = 4$	1.03×10^{-2} (1.4×10^{-2})	3.52×10^{-1} (4.6×10^{-1})	4.89×10^{0} (4.8×10^{0})
$N = 16$, $M = 8$	3.20×10^{-4} (4.9×10^{-4})	2.64×10^{-2} (3.7×10^{-2})	1.18×10^{0} (1.4×10^{0})
$N = 32$, $M = 16$	4.35×10^{-7} (6.2×10^{-7})	9.41×10^{-5} (1.3×10^{-4})	1.31×10^{-2} (1.7×10^{-2})
$N = 64$, $M = 32$	1.86×10^{-12} (2.4×10^{-12})	1.05×10^{-9} (1.4×10^{-9})	3.98×10^{-7} (5.1×10^{-7})
$N = 128$, $M = 64$	2.67×10^{-15} (1.5×10^{-14})	2.91×10^{-12} (1.1×10^{-11})	2.41×10^{-9} (8.6×10^{-9})

choice of which specific sine and cosine modes are used in the continuation ([6] uses more cosine modes than sine modes for M even). The approximation errors for the function and its first and second derivatives are shown in the table along with the published results from [6] in parentheses. The errors for the present algorithm were calculated using 25,000 points and clearly demonstrate that the current decoupling preserves the superalgebraic convergence of the Fourier continuation. The results in the last row of Table 3.1 depend heavily on ϵ (see Remark 3.1), as do the results presented in [6] and those in [3, 4]. Other comparisons yielded results similar to those

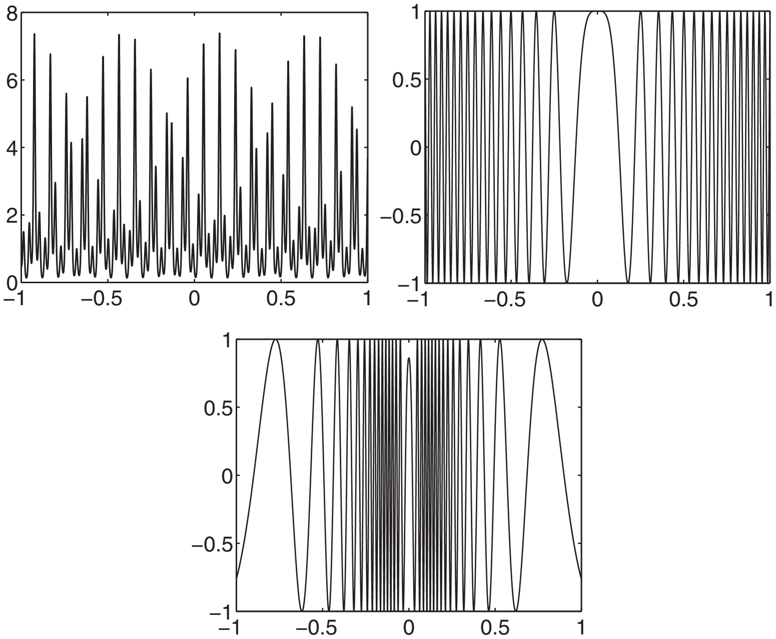

FIG. 3.2. *Depictions of the test functions for which continuation approximations will be computed in section* 3*. The top left function is* $e^{\sin(65.5\pi x-27\pi)-\cos(20.6\pi x)}$ *over the domain* $[0,1]$*. The figure on the top right is* $\cos(100x^2)$ *and the figure on the bottom is* $\cos(\frac{100}{1+25x^2})$*, both over the domain* $[-1,1]$*.*

shown here. For M sufficiently small (i.e., $M < 4\log_2(M) + 20$), there is no appreciable difference between the randomized algorithm and the full SVD solution to (2.20). Therefore, due to the relatively low number of points and modes required for an accurate approximation, Table 3.1 indicates the full SVD solution of (2.20) with $\epsilon = 5 \times 10^{-15}$. The differences in the accuracies between the full SVD solution to (2.20) and the randomized algorithm solution are detailed next.

Remark 3.1. Changing the cut-off can have a significant effect on the accuracy. Note with reference to Table 3.1 that if the value of ϵ is set at 10^{-13}, then the resulting accuracies are 1.17×10^{-13}, 1.39×10^{-10}, 1.12×10^{-7}, which, while very accurate, are significantly larger than the results for $\epsilon = 5 \times 10^{-15}$. The other lines of the Table 3.1 are not altered by this change in the cut-off.

In order to be able to accurately assess the performance of the randomized algorithm versus the full SVD, test functions are required that necessitate hundreds or thousands of Fourier modes for accurate approximations. Three such functions are depicted in Figure 3.2. Table 3.2 shows the results of calculating Fourier continuations on the first of these functions by inverting the full regularized SVD to solve (2.20) as well as the results of the randomized algorithm for various values of L. The results for the randomized algorithm were repeated 1000 times each, and the mean error is reported along the with standard deviation of the error. The error was again evaluated as the maximum error evaluated at 25,000 evenly spaced points. The number of Fourier coefficients M was set to be $N/2$ for all result in the table. As Table 3.2 shows, even with L set equal to $[4\log_2(M)]$, the results are reasonably close to the results of the full SVD, although with some variability. As L increases modestly, the variability in the results is reduced exceedingly fast. With $L = [4\log_2(M)] + 20$, the results

Table 3.2

Demonstration of the robustness of the randomized Fourier continuation algorithm. Continuations of the function $e^{\sin(65.5\pi x-27\pi)-\cos(20.6\pi x)}$ *depicted on the top right of Figure* 3.2 *were computed using the full SVD and the randomized algorithm for various values of the parameter* L*. In both cases the cut-off* ϵ *was set at* 10^{-14}*. The randomized results were repeated* 1000 *times each, and the mean error and associated standard deviation are reported for each set of parameters. The number of Fourier modes* M *was set to be* $N/2$*.*

L-$4\log_2(M)$	$N = 2048$	$N = 4096$	$N = 8192$
0	6.26×10^{-1} $\pm 5.13 \times 10^{-1}$	2.24×10^{-5} $\pm 1.17 \times 10^{-5}$	4.73×10^{-13} $\pm 3.03 \times 10^{-13}$
5	8.16×10^{-1} $\pm 1.48 \times 10^{-1}$	1.94×10^{-5} $\pm 1.84 \times 10^{-6}$	2.65×10^{-13} $\pm 1.01 \times 10^{-13}$
10	7.85×10^{-1} $\pm 7.50 \times 10^{-2}$	1.94×10^{-5} $\pm 9.52 \times 10^{-7}$	2.33×10^{-13} $\pm 3.38 \times 10^{-14}$
20	7.61×10^{-1} $\pm 3.67 \times 10^{-2}$	1.95×10^{-5} $\pm 5.40 \times 10^{-7}$	2.25×10^{-13} $\pm 1.24 \times 10^{-14}$
40	7.53×10^{-1} $\pm 1.99 \times 10^{-2}$	1.95×10^{-5} $\pm 3.43 \times 10^{-7}$	2.22×10^{-13} $\pm 3.30 \times 10^{-15}$
Full-SVD	7.59×10^{-1}	1.95×10^{-5}	2.22×10^{-13}

are fairly uniform and the resulting mean error is approximately 20 times larger than the standard deviation of the calculated error. The superalgebraic convergence of the technique is also clearly displayed in the table *for all values of* L *shown.* Doubling the number of Fourier coefficients from 2048 to 4096, which results from sampling the function at twice as many points, decreases the error by almost eight orders of magnitude (a factor of between 2^{26} to 2^{27}).

Many other tests have been run with other functions at different values for N and M. All results indicate the stability and reliability of the method independent of randomization. By choosing $L = [4\log_2(M)] + 20$, the difference between subsequent computations of an approximation to a given function is insignificant when compared to the difference between the function and any of the individual approximations. The algorithm is inherently random, and it follows that there does exist a finite (but very small) probability that it could produce significantly inaccurate results, although this has not been observed. In [26] it was shown, under certain conditions including choosing an appropriate parameter analogously to our choice of L, that the probability of failing to obtain an accurate SVD of a low-rank matrix would be less than 10^{-17}. The probability of failure can be controlled by the choice of L, and a modest increase in the value of L relative to $[4\log_2(M)]$ makes a failure significantly less likely.

Remark 3.2. The robustness of the error of the Fourier continuation does not extend to the actual Fourier coefficients that are calculated. Due to the ill-conditioning of the problem, the coefficients are inherently very sensitive to small changes in the system, including those introduced through the randomization procedure. Indeed, even though the error of the approximation changes very little, the computed Fourier coefficients can deviate by very large amounts if they are recomputed with a different matrix W. They can also change significantly with only minor changes in the value of ϵ for the full SVD continuation. If all that is required for a particular application is an accurate approximation of the function, or its derivatives, this is not generally a problem. Applying this algorithm in multiple dimensions and other applications may require calculations with greater stability. This is easily achieved in the present case through the Chebyshev reprojection that was first developed in [6] at a cost equivalent to a single USFFT. The Chebyshev representation is stable with respect to the data, and thus the resulting Chebyshev coefficients will not vary in amounts

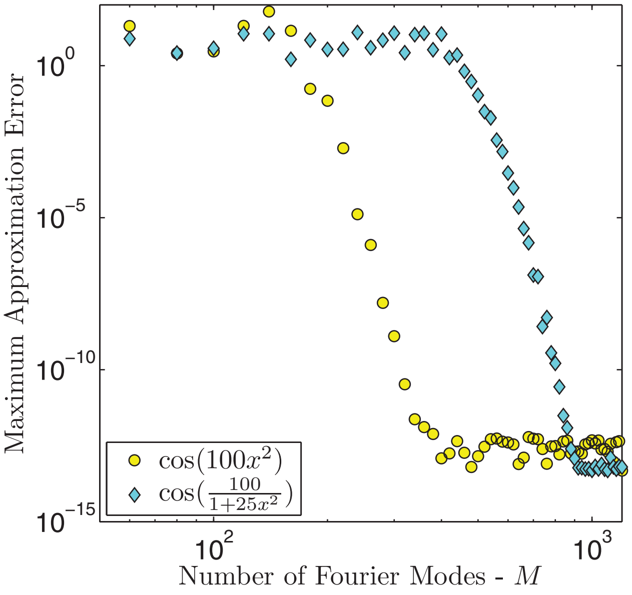

FIG. 3.3. *Convergence of the Fourier continuations, computed by the randomized algorithm of section* 2.6*, to the functions* $\cos(100x^2)$ *and* $\cos(\frac{100}{1+25x^2})$*. The errors in the continuations, shown in the figure for both functions, were evaluated as the maximum errors over* 25,000 *points in the domain. The results demonstrate the superalgebraic convergence of the randomized Fourier continuation.*

greater than the variability of the error of the Fourier continuation, which goes to zero as the Fourier continuation converges to the function.

As a further demonstration of the accuracy of the randomized Fourier continuation, it was applied to the functions $\cos(100x^2)$ and $\cos(\frac{100}{1+25x^2})$ that are depicted in the upper left and bottom, respectively, of Figure 3.2. The number of Fourier modes M was again set at $N/2$, and values of $\epsilon = 10^{-14}$ and $L = [4\log_2(M)] + 20$ were used. The approximation error was calculated as the maximum error over 25,000 points and is depicted in Figure 3.3 for both functions. The corresponding errors calculated by comparing the derivative of the Fourier continuation and the analytic derivative of the function are depicted in Figure 3.4. As can be seen in the figure, as soon as sufficiently large numbers of Fourier modes are used in the continuation, the convergence of the method is rapid to both the function and its derivative.

3.1. Approximation of noisy data and compression. We present a further numerical example in order to contrast the Fourier continuation used here with the FC(Gram) continuation that was introduced and detailed in [7, 27] for use in high efficiency PDE solvers. For a given N points, the FC(Gram) approach provides a highly accurate approximation at an asymptotic computational cost equivalent to a single FFT but provides only fixed order accuracy (here a 10th order approximation will be used). Our algorithm here is slower than the FC(Gram) by a factor of $\mathcal{O}(\log(M))$ but is superalgebraically convergent. In addition, the present algorithm can be particularly advantageous when approximating noisy data or trying to compress a smooth signal as we demonstrate here.

The function e^{-10x} was sampled at 10,000 evenly spaced points on the interval $[0, 1]$ and then approximated with both the present algorithm and the FC(Gram) algorithm. The thin dashed line in Figure 3.5 shows the accuracy of the FC(Gram)

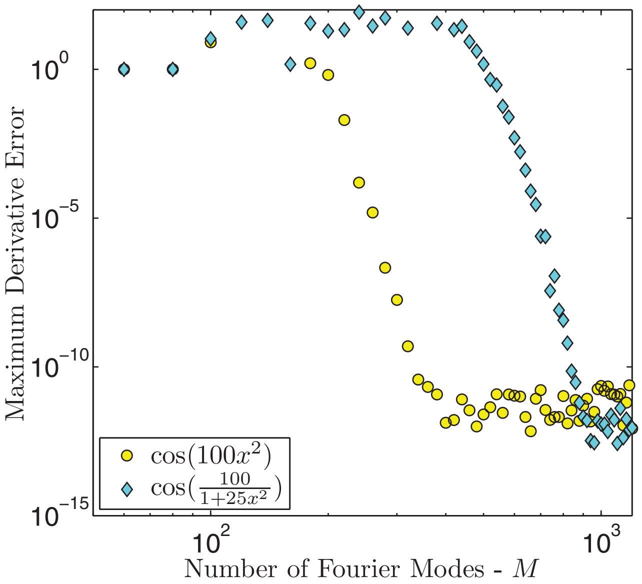

FIG. 3.4. *Convergence of the Fourier continuations to the derivatives of the functions, computed by applying the randomized algorithm of section* 2.6 *to the functions* $\cos(100x^2)$ *and* $\cos(\frac{100}{1+25x^2})$ *and then computing the difference between the derivative of the continuation and the analytic derivative of the function. The errors, shown in the figure for both functions, were calculated as the maximum errors over* 25,000 *points in the domain. The results demonstrate the superalgebraic convergence of the randomized Fourier continuation.*

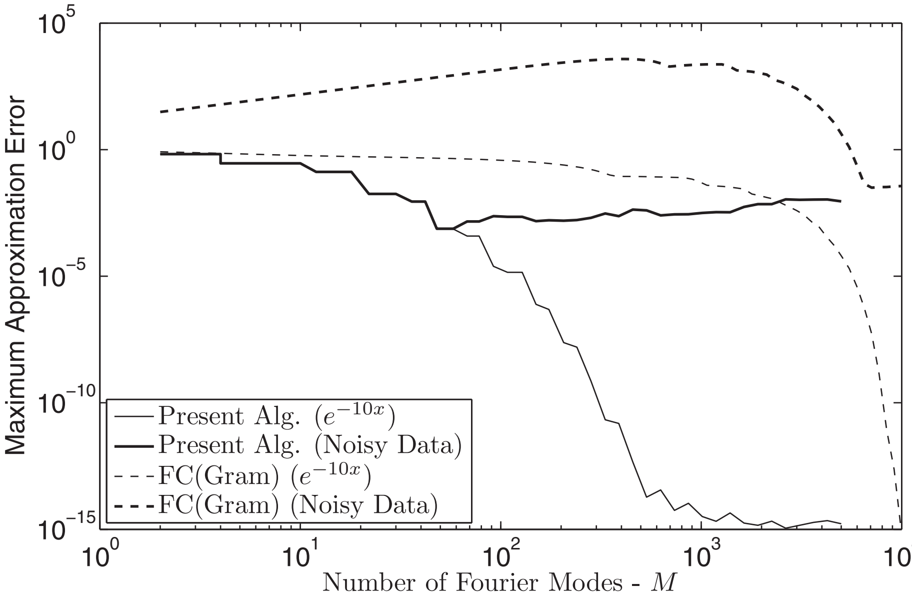

FIG. 3.5. *Comparison of the FC(Gram) method with the present Fourier continuation algorithm for smooth and noisy data.* $N = 10{,}000$ *points were used to sample the function* e^{-10x}, *and the errors were measured at these same* 10,000 *points.*

where only the lowest M frequency modes of the FC(Gram) approximation are used. The results for the FC(Gram) are due to the fixed value of N. With a smaller number of sampled points, the FC(Gram) could easily produce a very accurate approximation to this function with very few modes, but due to the design of the FC(Gram) algorithm, given N sampling points, $\mathcal{O}(N)$ Fourier modes are required to produce an accurate approximation. The present algorithm when applied to the same problem with various values of M, denoted by the thin solid line, clearly shows that the Fourier continuation algorithm developed here can provide a representation of the function with relatively few modes, without neglecting any of the data.

Normally distributed random noise with mean zero and a standard deviation of 0.01 generated by MATLAB was also added to the discrete sampled data of e^{-10x}. Since this noise is not limited to any frequency band, it cannot be completely eliminated by any projection. It can, however, be significantly attenuated as is shown with the thick solid line in the figure which indicates the difference between the Fourier continuation of the noisy data and the function e^{-10x}. Indeed with approximately 50 Fourier modes, the Fourier continuation-based approximation of the noisy data deviates from the base function by an amount approximately 30 times smaller than the noise that was introduced into the data. In contrast, the FC(Gram) algorithm produces large errors until nearly 10,000 Fourier modes are used, where the error for both the FC(Gram) and the Fourier continuation converges to a level nearly equivalent with the noise introduced. The results for the FC(Gram) approximation of the noisy data are denoted by the thick dashed line in Figure 3.5.

In the context of PDE solution, the foregoing results pose no difficulties for the FC(Gram) algorithm, noting that there is minimal additional cost in using $\mathcal{O}(N)$ Fourier modes for N data points within a PDE solver. Further, when using $\mathcal{O}(N)$ Fourier modes, the noise is not amplified, as shown in Figure 3.5, which is necessary for stability in the PDE solver. The FC(Gram) method is not, however, designed to eliminate or reduce this noise. In some other situations, including finding a smooth approximation of noisy data, the present Fourier continuation technique has significant advantages.

4. Conclusions.

A new randomized Fourier continuation algorithm has been presented for the computation of superalgebraically convergent Fourier series approximations to nonperiodic functions. The algorithm has a computational cost of $\mathcal{O}(N\log(N)\log(M))$, equivalent to the computational cost of performing a number of FFTs that is proportional to a logarithm of the number of Fourier modes sought. The accuracy and robustness of the method has been demonstrated through numerical examples. The new algorithm is an apt tool for signal processing and data analysis and is an extremely efficient computational method for overcoming a classical numerical challenge, namely the Gibbs phenomenon.

REFERENCES

[1] N. Albin and O. P. Bruno, *A spectral FC solver for the compressible Navier-Stokes equations in general domains* I: *Explicit time-stepping*, J. Comput. Phys., 230 (2011), pp. 6248–6270.

[2] J. P. Boyd, *A fast algorithm for Chebyshev, Fourier, and sinc interpolation onto an irregular grid*, J. Comput. Phys., 103 (1992), pp. 243–257.

[3] J. P. Boyd, *A comparison of numerical algorithms for Fourier extension of the first, second, and third kinds*, J. Comput. Phys., 178 (2002), pp. 118–160.

[4] J. P. Boyd and J. R. Ong, *Exponentially-convergent strategies for defeating the Runge phenomenon for the approximation of non-periodic functions, part* I: *Single-interval schemes*, Commun. Comput. Phys., 5 (2009), pp. 484–497.

[5] O. P. Bruno, *Fast, high-order, high-frequency integral methods for computational acoustics and electromagnetics*, in Topics in Computational Wave Propagation Direct and Inverse Problems Series, Lect. Notes Comput. Sci. Eng. 31, M. Ainsworth, P. Davies, D. Duncan, P. Martin, and B. Rynne, eds., springer, Berlin, 2003, pp. 43–82.

[6] O. P. Bruno, Y. Han, and M. M. Pohlman, *Accurate, high-order representation of complex three-dimensional surfaces via Fourier continuation analysis*, J. Comput. Phys., 227 (2007), pp. 1094–1125.

[7] O. P. Bruno and M. Lyon, *High-order unconditionally stable FC-AD solvers for general smooth domains* I. *Basic elements*, J. Comput. Phys., 229 (2010), pp. 2009–2033.

[8] P. Drineas, R. Kannan, and M. W. Mahoney, *Fast Monte Carlo algorithms for matrices* II: *Computing a low-rank approximation to a matrix*, SIAM J. Comput., 36 (2006), pp. 158–183.

[9] P. Drineas, R. Kannan, and M. W. Mahoney, *Fast Monte Carlo algorithms for matrices* III: *Computing a compressed approximate matrix decomposition*, SIAM J. Comput., 36 (2006), pp. 184–206.

[10] T. A. Driscoll and B. Fornberg, *A Padé-based algorithm for overcoming the Gibbs phenomenon*, Numer. Algorithms, 26 (2001), pp. 77–92.

[11] A. J. W. Duijndam and M. A. Schonewille, *Nonuniform fast Fourier transform*, Geophys., 64 (1999), pp. 539–551.

[12] A. Dutt and V. Rokhlin, *Fast Fourier transforms for nonequispaced data*, SIAM J. Sci. Comput., 14 (1993), pp. 1368–1393.

[13] A. Dutt and V. Rokhlin, *Fast Fourier transforms for nonequispaced data,* II, Appl. Comput. Harmon. Anal., 2 (1995), pp. 85–100.

[14] C. Fefferman, *Fitting a C^m-smooth function to data,* III, Ann. of Math. (2), 170 (2009), pp. 427–441.

[15] C. Fefferman, *Whitney's extension problems and the interpolation of data*, Bull. Amer. Math. Soc. (N.S.), 46 (2009), pp. 207–220.

[16] J. A. Fessler and B. P. Sutton, *Nonuniform fast Fourier transforms using min-max interpolation*, IEEE Trans. Signal Process., 51 (2003), pp. 560–574.

[17] A. Frieze, R. Kannan, and S. Vempala, *Fast Monte-Carlo algorithms for finding low-rank approximations*, J. ACM, 51 (2004), pp. 1025–1041.

[18] M. Frigo and S. G. Johnson, *The design and implementation of FFTW*3, Proc. IEEE, 93 (2005), pp. 216–231.

[19] J. Geer, *Rational trigonometric approximations to piece-wise smooth periodic functions*, J. Sci. Comput., 10 (1995), pp. 325–356.

[20] A. Gelb and J. Tanner, *Robust reprojection methods for the resolution of the Gibbs phenomenon*, Appl. Comput. Harmon. Anal., 20 (2006), pp. 3–25.

[21] D. Gottlieb, C.-W. Shu, A. Solomonoff, and H. Vandeven, *On the Gibbs phenomenon* I: *Recovering exponential accuracy from the Fourier partial sum of a nonperiodic analytic function*, J. Comput. Appl. Math., 43 (1992), pp. 81–98.

[22] D. Gottlieb and E. Tadmor, *Recovering pointwise values of discontinuous data within spectral accuracy*, in Progress and Supercomputing in Computational Fluid Dynamics, E. Murman and S. Abarbanel, eds., Birkhäuser, Boston, 1985, pp. 357–375.

[23] L. Greengard and J.-Y. Lee, *Accelerating the nonuniform fast Fourier transform*, SIAM Rev., 46 (2004), pp. 443–454.

[24] M. R. Hestenes, *Extension of the range of a differentiable function*, Duke Math. J., 8 (1941), pp. 183–192.

[25] D. Huybrechs, *On the Fourier extension of nonperiodic functions*, SIAM J. Numer. Anal., 47 (2010), pp. 4326–4355.

[26] E. Liberty, F. Woolfe, P.-G. Martinsson, V. Rokhlin, and M. Tyger, *Randomized algorithms for the low-rank approximation of matrices*, Proc. Natl. Acad. Sci. USA, 104 (2007), pp. 20167–20172.

[27] M. Lyon and O. P. Bruno, *High-order unconditionally stable FC-AD solvers for general smooth domains* II. *Elliptic, parabolic and hyperbolic PDEs; theoretical considerations*, J. Comput. Phys., 229 (2010), pp. 3358–3381.

[28] A. Majda, J. McDonough, and S. Osher, *The Fourier method for nonsmooth initial data*, Math. Comp., 32 (1978), pp. 1041–1081.

[29] M. S. Mock and P. D. Lax, *The computational of discontinuous solutions of linear hyperbolic equations*, Comm. Pure Appl. Math., 31 (1978), pp. 423–430.

[30] N. Nguyen and Q. H. Liu, *The regular Fourier matrices and nonuniform fast Fourier transforms*, SIAM J. Sci. Comput., 21 (1999), pp. 283–293.

[31] G. Strang, *The discrete cosine transform*, SIAM Rev., 41 (1999), pp. 135–147.

[32] J. Tanner, *Optimal filter and mollifier for piecewise smooth spectral data*, Math. Comp., 75 (2006), pp. 767–790.

[33] L. N. Trefethen and D. Bau III, *Numerical Linear Algebra*, SIAM, Philadelphia, 1997.

[34] A. F. Ware, *Fast approximate Fourier transforms for irregularly spaced data*, SIAM Rev., 40 (1998), pp. 838–856.

[35] H. Whitney, *Analytic extensions of differentiable functions defined in closed sets*, Trans. Amer. Math. Soc., 36 (1934), pp. 63–89.

[36] F. Woolfe, E. Liberty, V. Rokhlin, and M. Tygert, *A fast randomized algorithm for the approximation of matrices*, Appl. Comput. Harmon. Anal., 25 (2008), pp. 335–366.