UCLA
COMPUTATIONAL AND APPLIED MATHEMATICS

Rapid Computation of the Discrete Fourier Transform

Chris Anderson
Marie Dillon Dahleh

August 1994

CAM Report 94-25

Department of Mathematics
University of California, Los Angeles
Los Angeles, CA. 90024-1555

Rapid Computation of the Discrete Fourier Transform

Chris Anderson * Marie Dillon Dahleh †

August 5, 1994

Abstract

We present algorithms for the rapid computation of the forward and inverse discrete Fourier transform for points which are non-equispaced or whose number is unrestricted. The computational procedure is based on approximation using a local Taylor series expansion and the Fast Fourier Transform. The forward transform for non-equispaced points is computed as the solution of a linear system involving the inverse Fourier transform. This latter system is solved using the iterative method GMRES with pre-conditioning. Numerical results are given to confirm the efficiency of the algorithms.

1 Introduction

The Fast Fourier Transform (FFT) [1] is a powerful tool which is used in numerous applications ranging from signal processing to rock mechanics [4]. In order to use the FFT one must have uniformly spaced data and numbers of points which are restricted to have certain values (e.g. a power of 2 or

*IBM T.J. Watson Research Center, Yorktown Heights, NY, 10598
†Department of Mathematics, UCLA, Los Angeles, California, 90024

a product of primes). The goal of this paper is to describe a method for computing rapidly a forward and inverse discrete Fourier transform on sets of data points which are either non-equispaced and/or are unrestricted in number. Our method is similar in spirit to the papers of Dutt and Rohklin [2], [3], in that the method relies on approximation of results obtained with the standard FFT. The major differences between the two approaches is that Dutt and Rohklin utilize a fast multipole method to perform the approximation whereas we use a local Taylor series expansions. Our method also differs in the construction of the forward transform in that we employ a preconditioner and use an iterative method which does not require the transpose of the inverse transform matrix. We found our approach easier to implement because it is based on conventional computational techniques which are easy to program - namely the standard FFT and Taylor series expansions whose coefficients are computed spectrally.

In the next section we describe the specific computational problem associated with the discrete forward and inverse Fourier transform. In section 3 we give a detailed description of our numerical algorithm. The key idea behind the inverse transform is to compute transformed values at a set of equispaced points using a standard FFT and then approximate the required values at the non-equispaced points (or arbitrary number) using local Taylor series expansions. The derivatives necessary for these expansions are computed spectrally. The use of local Taylor series expansions to represent the trigonometric sum is entirely appropriate since the sum is an analytic function. The computational procedure for the forward transform of a set of equispaced points of arbitrary number is the same as for the inverse transform (one just interchanges the notion of coefficients and function values). For non-equispaced points the forward transform is computed as the solution of a linear system of equations defined via the inverse transform. This system is solved iteratively using the generalized minimum residual method (GMRES) with preconditioning [5]. For each iteration an inverse Fourier transform must be computed, but since the number of iterations is quite small the method is efficient.

2 The Problem

In this paper we consider the forward Fourier transform as solving the trigonometric interpolation problem : Given the N distinct data points $\{x_j\}$ (not necessarily uniformly spaced) on the interval $[0, 2\pi]$, and associated function values $\{y_j\}$, find the N coefficients $\{\alpha_k\}$ so that the trigonometric sum formed with these coefficients interpolates the given data - i.e.

$$y_j = \frac{1}{N} \sum_{k=0}^{N-1} \alpha_k e^{i\,k\,x_j} \tag{1}$$

for $j = 0 \ldots N-1$.

The set of values $\{\alpha_k\}$ are thus the coefficients of the data $\{y_j\}$ when expressed in the discrete basis obtained by evaluating the Fourier basis elements at the nodes $\{x_j\}$, i.e. the discrete forward Fourier transform of $\{y_j\}$.

If the points are equispaced, then the orthogonality of the discrete Fourier basis functions implies that

$$\alpha_k = \sum_{j=0}^{N-1} y_j e^{-i\,k\,x_j} \tag{2}$$

Alternately, the inverse discrete Fourier transform consists of evaluating the trigonometric interpolant; i.e. given a set of Fourier coefficients $\{\beta_k\}$ and a set of points $\{x_j\}$ determine the values $\{y_j\}$ by evaluating the sums

$$y_j = \frac{1}{N} \sum_{k=0}^{N-1} \beta_k e^{i\,k\,x_j} \tag{3}$$

for $j = 0 \ldots N-1$.

3 The Numerical Method

3.1 Inverse Transform

Our procedure for the forward discrete Fourier Transform uses our construction of the inverse discrete Fourier Transform, and so we describe the inverse

transform first. The procedure for the inverse transform consists of using the standard fast inverse transform on a set of $M = 2^q$ equispaced points to determine a set of local Taylor series expansions centered at these points. The Taylor series expansions are then evaluated at the desired set of non-equispaced points or equispaced points of arbitrary number.

Assume we have the N distinct points $\{x_j\}$ in an interval $[0, 2\pi]$ and we are given a set of N Fourier coefficients $\{\beta_k\}$. We first choose a value of M which is a power of 2, and for which $2\pi(N-1) \leq M$. Let $\{\tilde{x}_m\}$ be the set of M equispaced points in the interval $[0, 2\pi]$. Then

- Construct a new set of coefficients $\{\tilde{\beta}_k\}$ by padding and scaling the coefficients $\{\beta_k\}$,

$$\tilde{\beta}_k = \begin{cases} \left(\frac{M}{N}\right) \beta_k & k \leq N \\ 0 & N < k \leq M \end{cases}$$

  The scaling factor $\frac{M}{N}$ is introduced so that the inverse transform of the padded coefficients $\{\tilde{\beta}_k\}$ yields a function which is equivalent to that of the original coefficients at the points $\{x_j\}$.

- Using the discrete inverse FFT compute the values and the first p derivatives of the function

$$\tilde{y}(x) = \frac{1}{M} \sum_{k=0}^{M-1} \tilde{\beta}_k e^{i\,k\,\tilde{x}} \tag{4}$$

  at the points $\{\tilde{x}_m\}$. The derivatives are evaluated spectrally, i.e. the p^{th} derivative is obtained by computing the inverse transform of a set of coefficients of the form $(i\,k\,)^p\,\tilde{\beta}_k$.

- For a given point x_j, determine the closest of the equispaced points, say $\tilde{x}^*$ to it. Using the values and derivatives of (4) at $\tilde{x}^*$ one approximates the value at the point x_j by using a p^{th} order Taylor series expansion about $\tilde{x}^*$.

The order p of the Taylor series expansion used is determined by the desired precision required in the function values. If these values are required to a

precision ϵ (which may be unit round-off) then the maximal order of the approximation required to do this is the value of p so that

$$\frac{\gamma}{(p+1)!} \leq \epsilon \tag{5}$$

where $\gamma = \max_{0 \leq k \leq N-1} |\beta_k|$. This follows from the fact that the error in an order p Taylor expansion of an individual term in (4) with wavenumber k is bounded by

$$\frac{1}{M} \frac{|\tilde{\beta}_k| h^{p+1} k^{p+1}}{(p+1)!} = \frac{1}{N} \frac{|\beta_k| h^{p+1} k^{p+1}}{(p+1)!}.$$

Since $k \leq N-1$ and $h = \frac{2\pi}{M}$ we have,

$$\frac{1}{N} \frac{|\beta_k| h^{p+1} k^{p+1}}{(p+1)!} \leq \frac{1}{N} \left(\frac{|\beta_k|}{(p+1)!} \right) \left(\frac{2\pi(N-1)}{M} \right)^{p+1} \leq \frac{1}{N} \frac{|\beta_k|}{(p+1)!}.$$

The error in approximating the sum of N Fourier components is thus bounded by N times the maximum of this value, i.e. (5). The estimate (5) is only a bound, and in the implementation of the approximation procedure one adaptively determines the size of p that one should use - i.e. one accumulates the Taylor series approximation term by term and stops when the error is within the desired precision. Our computational experiments indicate that far fewer terms than the number suggested by the error bound need be used.

3.2 Forward Transform

3.2.1 Equispaced Points

If the points $\{x_j\}$ are equispaced then the forward transform of values $\{y_j\}$ is given by

$$\alpha_k = \sum_{j=0}^{N-1} y_j e^{-ikx_j}. \tag{6}$$

Since $x_j = j\left(\frac{2\pi}{N}\right)$, then we can rewrite the sum as

$$\alpha_k = \sum_{j=0}^{N-1} y_j e^{-ikx_j} = \sum_{j=0}^{N-1} y_j e^{-ijk\left(\frac{2\pi}{N}\right)} = \sum_{j=0}^{N-1} y_j e^{-ijz_k}.$$

Thus, the α_k's are the values of the function $\sum_{j=0}^{N-1} y_j e^{-ijz_k}$ evaluated at the N equispaced points $z_k = k\left(\frac{2\pi}{N}\right)$ in the interval $[0, 2\pi]$. The task of evaluating the forward transform for equispaced points thus has the same form as the evaluation of the inverse transform - so one can use the technique for the inverse transform described previously with only slight modifications. In particular, since the factor $\frac{1}{N}$ does not appear in the forward transform sum, one does not scale $\{y_j\}$ by $\left(\frac{M}{N}\right)$ when padding these values to create the M values $\{\tilde{y}_j\}$. Also, the error bound for the Taylor series approximation becomes

$$\frac{N\nu}{(p+1)!} \leq \epsilon \tag{7}$$

where $\nu = \max_{0 \leq j \leq N-1} |y_j|$. While this error bound does depend on N, the dependence is rather weak, and our computational experiments indicate that it is of no great concern.

The problem of the forward transform for non-equispaced points is more challenging because the the coefficients $\{\alpha_k\}$ are not given by the sum (6). Thus, a different procedure must be employed. In our procedure we build upon the fact that we can compute the inverse transform for non-equispaced points rapidly. In particular, the goal of the forward transform is to find coefficients $\{\alpha_k\}$ so that the inverse transform of these coefficients interpolates a given set of function values $\{y_j\}$. Expressed in matrix/vector notation, the forward transform consists in finding the vector of coefficients $\vec{\alpha}$ so that the linear system

$$A\vec{\alpha} = \vec{y}, \tag{8}$$

is satisfied. A is the representation in matrix form of the inverse Fourier transform.

Our procedure for the forward transform is just to solve the linear system (8) for the coefficients $\vec{\alpha}$. We choose to solve this system iteratively, because this will involve operations of the form $A\vec{z}$ for vectors $\vec{z}$ - i.e. applications of the inverse transform which can be computed efficiently using the technique described above.

The iterative method we choose to use was the generalized minimum residual method (GMRES) with preconditioning [5]. The GMRES method is a Krylov subspace iterative method for solving nonsymmetric linear systems.

There are several advantages to this method. First, each iteration can be performed quickly because the bulk of the work is contained in the matrix multiplication step $A\vec{\alpha}^{(i)}$ which is just the evaluation of the inverse transform for each iterate. Second, it is optimal in the sense that it minimizes the residual in a given Krylov subspace. Third, it does not require the knowledge of A^T. The major disadvantage of this method is that it may require several iterations to converge. At each iteration, one computes the next basis element for the Krylov subspace. All previous basis elements are needed. If one needs a large number of iterations to converge, then one may need significant storage for this procedure. However, with the preconditioner described below the method converges in a few iterations, and the storage requirements of GMRES does not pose a problem.

A preconditioner for the system (8) consists of an operator which takes a set of values $\vec{y}$ and returns another set of values $\vec{\alpha}$. After some experimentation we found that a satisfactory preconditioner was simply the forward Fourier transform of the values $\{y_j\}$ assuming that they are associated with equispaced points. (Thus the preconditioner is the exact inverse of the matrix A in the case of equispaced points.)

4 Numerical Experiments

We have implemented our method in Fortran using double precision arithmetic. All the calculations were run on a Sparc 10. For the inverse transform we report two types of error. The first is the relative ∞-norm which is defined to be

$$E_\infty = \frac{\max_{1\le i\le N} |y_i^a - y_i|}{\max_{1\le i\le N} |y_i|},$$

and the second is the relative 2-norm error defined by

$$E_2 = \left(\frac{\sum_{i=1}^N |y_i^a - y_i|^2}{\sum_{i=1}^N |y_i|^2}\right)^{\frac{1}{2}}$$

where y_i^a represents the approximation to the y_i. For the forward transform we report the final residual error.

4.1 Inverse Transform Experiments

The first set of experiments are concerned with the inverse transform. We consider two cases. In Tables 1-3, we present results for points which are not equispaced but whose number is a power of two. We examine how the degree of non uniformity of the points affects the number of terms in the Taylor series, p, needed to obtain a desired accuracy. We define the nonuniform grid as follows:

$$x_j = x_j^{unif} + h * \eta * factor$$

where x_j^{unif} represents the j^{th} grid point on a uniform grid, h is the grid spacing, η a uniformly distributed random number between 0 and 1 and $factor$ is either .01 (Table 1) or .1 (Table 2). As one can see from Table 1 and Table 2, for both perturbation factors, our method is slower than the direct evaluation of (3) for $n < 64$ and faster for $n > 64$. One would expect as the perturbation factor increases from .01 to .1, the number of Taylor series terms required to obtain a given accuracy increases. This is in fact what happens. We use an error tolerance of 10^{-10}. The number of points for which our method is faster than the direct method is lower than the break even point reported for a similar problem in either [2] or [3]. In [2], the break even point occurs for $n \geq 256$ and in [3] the break even point occurs for $n \geq 2048$.

For Table 1 and Table 2, we use smooth Fourier coefficients generated by $\beta_k = \sin(x_k)$. In Table 3 we use randomly generated coefficients. The coefficients are contained in the interval $[0, 1]$. As one can see, for a given perturbation factor, the required number of Taylor series terms does not depend on the smoothness of the coefficients. Again, the order of the Taylor series required is substantially lower than the order predicted by our error estimates.

In Table 4 we present the results for a calculation with uniformly spaced points which are not a power of 2. One sees that our method is faster than the direct evaluation of (3) for calculations of more than 371 points.

4.2 Forward Transform Experiments

As we have mentioned previously, the computational task for the forward transform for equally spaced points has the same form as that for the inverse

N	p bound	p required	sec.	dir. sec.	E_∞	E_2
16	17	7	.01	0	1.1825×10^{-8}	1.2164×10^{-8}
32	18	7	.02	.01	1.1792×10^{-8}	1.5152×10^{-8}
64	18	7	.03	.03	8.2338×10^{-9}	1.2762×10^{-8}
128	18	7	.09	.12	5.8926×10^{-9}	1.3183×10^{-8}
256	18	8	.21	.49	4.8255×10^{-9}	1.4178×10^{-8}
512	19	8	.48	2.02	3.0118×10^{-9}	1.4329×10^{-8}
1024	19	8	1.02	8.32	2.4349×10^{-9}	1.4618×10^{-8}

Table 1: p is the order of the Taylor series. Random .01 perturbation for the unfirom grid. Smooth coefficients. Error tolerance is 10^{-10}.

N	p bound	p required	sec.	dir. sec.	E_∞	E_2
16	17	11	.01	0	1.6951×10^{-8}	1.6017×10^{-8}
32	18	11	.03	.01	1.1192×10^{-8}	1.5587×10^{-8}
64	18	11	.05	.05	8.0280×10^{-9}	1.4234×10^{-8}
128	18	11	.12	.13	5.7089×10^{-9}	1.4554×10^{-8}
256	18	11	.3	.49	4.4565×10^{-9}	1.4754×10^{-8}
512	19	12	.68	2.02	3.8060×10^{-9}	1.4754×10^{-8}
1024	19	12	1.56	8.32	4.0123×10^{-9}	1.4547×10^{-8}

Table 2: p is the order of the Taylor series. Random .1 perturbation for the uniform grid. Smooth coefficients. Error tolerance is 10^{-10}.

N	p bound	p required	sec.	dir. sec.	E_∞	E_2
16	17	11	.01	0	1.2223×10^{-8}	9.1421×10^{-9}
32	18	11	.02	.01	8.0229×10^{-9}	7.0160×10^{-9}
64	18	11	.05	.03	6.7398×10^{-9}	7.1751×10^{-9}
128	18	11	.12	.12	4.9839×10^{-9}	6.4089×10^{-9}
256	18	12	.32	.49	3.6305×10^{-9}	5.8583×10^{-9}
512	19	12	.74	2.16	3.4041×10^{-9}	4.8901×10^{-9}
1024	19	12	1.63	8.14	2.4524×10^{-9}	4.0691×10^{-9}

Table 3: p is the order of the Taylor series. Random perturbation of .1. Random coefficients. Error tolerance is 10^{-10}.

N	p bound	p required	sec.	dir. sec.	E_∞	E_2
11	17	8	.01	0	1.5206×10^{-8}	1.5441×10^{-8}
37	18	8	.04	.01	9.4308×10^{-9}	1.5486×10^{-8}
61	18	9	.04	.03	1.0360×10^{-8}	1.5632×10^{-8}
123	18	9	.1	.11	7.0404×10^{-9}	1.3590×10^{-8}
179	18	9	.26	.24	6.7775×10^{-9}	1.4953×10^{-8}
371	19	9	.49	1.04	5.2059×10^{-9}	1.4134×10^{-8}

Table 4: p is the order of the Taylor series. Uniformly spaced non power of 2. Smooth coefficients. Error tolerance is 10^{-10}.

transform, therefore there is no need to discuss it independently. Thus, in this section we only consider the case of the forward transform for data which is non-equispaced and whose number is a power of two. In Table 5 and Table 6, we consider a .01 perturbation and a .1 perturbation respectively. For a given perturbation the number of GMRES iterations needed to converge remains almost constant. This is an indication that the preconditioner is doing a good job. Without the preconditioner the number of iterations increases dramatically with increasing N.

As the size of the perturbation increases, the number of iterations also increases. One should expect this behavior because the larger the perturbation the further the preconditioner is from the true matrix inverse. Even so, a ten fold increase in the perturbation produces less than a three fold increase in the cpu seconds needed for the calculation. In comparison with directly evaluating (1), the break even point for the forward transform is about N = 4096. As better preconditioners are developed this number should drop.

In [2], Dutt and Rohklin present an iterative method for the solution of the algebraic problem. Our approach differs from theirs in that they use a conjugate gradient method whereas we use GMRES. Since the matrix A in (8) is not symmetric, one cannot use the conjugate gradient method directly for this system. Dutt and Rohklin therefore work with a reformulation of the matrix problem. By using GMRES we avoid using this reformulation. We also employ a preconditioner - a component which we found to be critical for success.

N	iterations	sec.	residual error
16	3	.12	5.8663×10^{-7}
32	3	.22	9.3676×10^{-7}
64	3	.46	8.8786×10^{-7}
128	3	.98	8.7670×10^{-7}
256	3	2.32	9.6058×10^{-7}
512	4	6.69	1.2012×10^{-8}

Table 5: Forward transform. Random .01 perturbation from uniform grid.

N	iterations	sec.	residual error
16	8	.25	1.0326×10^{-7}
32	9	.59	4.09265×10^{-7}
64	9	1.27	9.6050×10^{-7}
128	10	2.85	3.8367×10^{-7}
256	10	6.95	7.1286×10^{-7}
512	10	15.5	6.5907×10^{-7}

Table 6: Forward transform. Random .1 perturbation from uniform grid.

5 Conclusions

In this paper we present a method for computing the forward and inverse discrete Fourier transform. The method for the inverse transform (and the forward transform of equispaced data with arbitrary numbers of points) is a combination of the standard FFT and approximation using local Taylor series expansions. Both the forward and inverse transforms are easy to implement and faster than directly evaluating (1) and (3) for reasonably small numbers of points. The procedures reduce significantly the penalty of using the discrete Fourier transform on numbers of points which are not power of two or the product of primes.

The forward transform for non-equispaced data relies on the inverse transform and the iterative method GMRES with a preconditioner. The choice of the preconditioner was essential for the iterative method to converge rapidly. In our implementation we found that the procedure for the forward transform on non-equispaced points is less efficient than the direct method for numbers of points less than 4096. However, with improved coding and de-

velopment of more appropriate preconditioners, this value should decrease. The methods presented here clearly extend to higher dimensional discrete Fourier transforms.

6 Acknowledgements

The work of the first author was partially supported by ONR Contract #N00014-92-J-1890 and the second author was partially supported by the National Science Foundation, DMS-9206192.

References

[1] J. W. Cooley and J. W. Tukey, "An algorithm for the machine computation of complex Fourier series," *Math. Comp.*, 19 (1965), pp.297-301.

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

[3] A. Dutt and V. Rokhlin, " Fast Fourier transforms for nonequispaced data II," research report Yaleu/dcs/rr-980, August 1993.

[4] A. P. Peirce, S. Spottiswoode, and J. A. L. Napier, " The spectral boundary element method: a new window on boundary elements in rock mechanics," *Int. J. Rock Mech. Min. Sci. and Geomech. Abstr.* 29 (1992), pp. 379-400.

[5] Y. Saad and M. Schultz, " GMRES: a generalized minimal residual algorithm for solving nonymmetric linear systems," *SIAM J.Sci. Stat. Comput.* 7 (1986), pp. 856-869.