A NONUNIFORM FAST FOURIER TRANSFORM BASED ON LOW RANK APPROXIMATION

DIEGO RUIZ–ANTOLN[*] AND ALEX TOWNSEND[†]

Abstract. By viewing the nonuniform discrete Fourier transform (NUDFT) as a perturbed version of a uniform discrete Fourier transform, we propose a fast, stable, and simple algorithm for computing the NUDFT that costs $\mathcal{O}(N \log N \log(1/\epsilon)/\log\log(1/\epsilon))$ operations based on the fast Fourier transform, where N is the size of the transform and $0 < \epsilon < 1$ is a working precision. Our key observation is that a NUDFT and DFT matrix divided entry-by-entry is often well-approximated by a low rank matrix, allowing us to express a NUDFT matrix as a sum of diagonally-scaled DFT matrices. Our algorithm is simple to implement, automatically adapts to any working precision, and is competitive with state-of-the-art algorithms. In the fully uniform case, our algorithm is essentially the FFT. We also describe quasi-optimal algorithms for the inverse NUDFT and two-dimensional NUDFTs.

Key words. fast Fourier transform, nonequispaced, low rank matrices, Chebyshev

AMS subject classifications. 41A10, 65T50

1. Introduction. The nonuniform discrete Fourier transform (NUDFT) is an important task in computational mathematics that appears in signal processing [4], the numerical solution of partial differential equations [20], and in magnetic resonance imaging [12]. Quasi-optimal algorithms for computing the NUDFT are referred to as nonuniform fast Fourier transforms (NUFFTs), and state-of-the-art NUFFTs are usually based on oversampling, discrete convolutions, and the fast Fourier transform (FFT) on an oversampled grid [10, 15, 23, 28]. In this paper, we propose a NUFFT that is embarrassingly parallelizable. It is numerically stable without the need for oversampling, and costs K FFTs, where K is a carefully selected integer. Our central idea is to exploit a low rank observation (see (3)).

Let $N \geq 1$ be an integer and $\underline{c} = (c_0, \ldots, c_{N-1})^\mathsf{T}$ be an $N \times 1$ vector with complex entries. The one-dimensional NUDFT computes the vector $\underline{f} = (f_0, \ldots, f_{N-1})^\mathsf{T}$, defined by the following sums:

$$f_j = \sum_{k=0}^{N-1} c_k e^{-2\pi i x_j \omega_k}, \qquad 0 \leq j \leq N-1, \tag{1}$$

where $x_0, \ldots, x_{N-1} \in [0, 1]$ are *samples* and $\omega_0, \ldots, \omega_{N-1} \in [0, N]$ are *frequencies*. Since (1) involves N sums with each sum containing N terms, computing the vector $\underline{f}$ naively costs $\mathcal{O}(N^2)$ operations. If the samples are equispaced, i.e., $x_j = j/N$, and the frequencies are integer, i.e., $\omega_k = k$, then the transform is fully uniform and (1) can be computed by the FFT in $\mathcal{O}(N \log N)$ operations by exploiting algebraic redundancies [8]. Unfortunately, these algebraic redundancies are "brittle" [5] and the ideas behind the FFT are not immediately useful when either the samples are nonequispaced or the frequencies are noninteger. To develop a NUFFT, one has to exploit a nonzero working precision of $0 < \epsilon < 1$ and make careful approximations.

There are three types of NUDFTs [15]:

---

[*]Departamento de Matemáticas, Estadística y Computación Universidad de Cantabria, Av. de los Castros 48 E-39005 Santander, Spain. (`diego.ruizantolin@unican.es`)

[†]Department of Mathematics, Cornell University, Ithaca, NY 14853. (`townsend@cornell.edu`)

- NUDFT-I (Uniform samples and noninteger frequencies): In (1) the samples are equispaced, i.e., $x_j = j/N$, and the frequencies $\omega_0, \ldots, \omega_{N-1}$ are noninteger. This corresponds to evaluating a generalized Fourier series at equispaced points. In Section 3.1, we describe a quasi-optimal algorithm for computing the NUDFT-I referred to as a NUFFT-I.
- NUDFT-II (Nonuniform samples and integer frequencies): In (1) the frequencies are integers and the samples $x_0, \ldots, x_{N-1}$ are nonequispaced points in $[0, 1]$. This NUDFT corresponds to evaluating a Fourier series at nonequispaced points. In Section 2, we describe an $\mathcal{O}(N \log N \log(1/\epsilon)/\log\log(1/\epsilon))$ algorithm, referred to hereafter as the NUFFT-II, for computing the NUDFT-II with a working precision of ϵ. Note that this transform also goes by the acronym NFFT [22].
- NUDFT-III (Nonuniform samples and nonuniform frequencies): In (1) the samples $x_0, \ldots, x_{N-1}$ are nonequispaced and the frequencies $\omega_0, \ldots, \omega_{N-1}$ are noninteger. This is the fully nonuniform transform and corresponds to evaluating a generalized Fourier series at nonequispaced points. The NUDFT-III and its applications in image processing and the numerical solution of partial differential equations are discussed in [20]. In Section 3.2, we derive an $\mathcal{O}(N \log N \log(1/\epsilon)/\log\log(1/\epsilon))$ complexity algorithm for computing the NUDFT-III by combining our NUFFT-I and NUFFT-II. We refer to this as a NUFFT-III [20], but others use the acronym NNFFT [22].

Initially, we focus on computing the NUDFT-II. This is perhaps the easiest to think about as it corresponds to evaluating a Fourier series at nonequispaced points. A convenient and compact way to write the NUDFT-II in (1) is as a matrix-vector product: Given Fourier coefficients $\underline{c} \in \mathbb{C}^{N \times 1}$, compute values $\underline{f} \in \mathbb{C}^{N \times 1}$ such that

$$\underline{f} = \tilde{F}_2 \underline{c}, \qquad (\tilde{F}_2)_{jk} = e^{-2\pi i x_j k}, \quad 0 \le j, k \le N - 1, \tag{2}$$

where $x_0, \ldots, x_{N-1}$ are sample points. Therefore, a NUFFT-II is simply a quasi-optimal complexity algorithm for computing the matrix-vector product $\tilde{F}_2 \underline{c}$. In the fully uniform case when $x_j = j/N$ and $\omega_k = k$, we use the notation $F_{jk} = e^{-2\pi i j k/N}$ for the DFT matrix and note that the FFT algorithm computes $F\underline{c}$ in $\mathcal{O}(N \log N)$ operations [8].

Our NUFFT-II algorithm is based on the simple observation that if the samples are near-equispaced, then $\tilde{F}_2 \oslash F$ can be well-approximated by a low rank matrix.[1] That is, for a small integer K (see Table 1), we find that

$$\tilde{F}_2 \oslash F \approx \underline{u}_0 \underline{v}_0^\mathsf{T} + \cdots + \underline{u}_{K-1} \underline{v}_{K-1}^\mathsf{T}, \qquad \underline{u}_0, \ldots, \underline{u}_{K-1}, \underline{v}_0, \ldots, \underline{v}_{K-1} \in \mathbb{C}^{N \times 1}, \tag{3}$$

where '$\oslash$' denotes the Hadamard division, i.e., $C = A \oslash B$ means that $C_{jk} = A_{jk}/B_{jk}$. With (3) in hand, we have

$$\tilde{F}_2 \underline{c} \approx \left(\left(\underline{u}_0 \underline{v}_0^\mathsf{T} + \cdots + \underline{u}_{K-1} \underline{v}_{K-1}^\mathsf{T} \right) \circ F \right) \underline{c} = \sum_{r=0}^{K-1} D_{\underline{u}_r} F D_{\underline{v}_r} \underline{c}, \tag{4}$$

where '$\circ$' is the Hadamard product[2] and $D_{\underline{u}}$ is the diagonal matrix with the entries of $\underline{u}$ on the diagonal. Therefore, the NUFFT-II can be computed in $\mathcal{O}(KN \log N)$

---

[1] A similar observation was made in [19, Sec. 4], but we believe that it has not been developed into a practical algorithm. A different Hadamard product matrix decomposition was exploited in [26] to derive a fast Chebyshev-to-Legendre transform.

[2] If $C = A \circ B$, then $C_{jk} = A_{jk}B_{jk}$.

operations via K diagonally-scaled FFTs. The approximation in (4) is the main idea in this paper. All that remains is to select the integer K and compute the vectors $\underline{u}_0, \ldots, \underline{u}_{K-1}, \underline{v}_0, \ldots, \underline{v}_{K-1}$. The observation will lead to a NUFFT-II algorithm that is quasi-optimal for any set of samples and frequencies (see Section 2) and similar observations lead to our NUFFT-I and NUFFT-III algorithms.

The major computational cost of our NUFFTs is K FFTs that can be performed in parallel, where K is an adaptively selected integer that depends on the working precision $0 < \epsilon < 1$ and the distribution of the samples and frequencies. This allows us to reduce the cost of our NUFFTs — by reducing K — when the working precision is loosened, the samples are near-equispaced, or the frequencies are close to being integers. In particular, if any of our NUFFT codes are given equispaced samples and integer frequencies, then $K = 1$, and our implementation reduces to a single FFT. By always computing the NUDFT via K FFTs, we are able to leverage the efficient FFTW library that has an implementation of the FFT that adapts to individual computer architectures [13]. Our algorithm relies on FFTs that are of the same size as the original NUFFT and we automatically exploit the distribution of the samples and frequencies if they happen to be quasi-uniform for extra computational speed.

There are many other NUFFTs in the literature based on various ideas such as discrete convolutions and oversampling [10, 15, 23], min-max interpolation [12], oversampling and interpolation [7], and a Taylor-based approach [1]. The Taylor-based approach results in an easily implementable algorithm, which is avoided in practice because it is numerically unstable [18, Ex. 3.10]. For the last two decades, discrete convolutions and oversampling has been preferred. The transforms that we develop here are convenient and simple while being numerically stable. We benchmark our algorithms against the Julia implementation of the NFFT software [22] to demonstrate that our proposed algorithm is competitive with existing state-of-the-art approaches.

The paper is structured as follows. In Section 2, we derive the NUFFT-II algorithm by first assuming that the nonuniform samples are a perturbed equispaced grid (see Section 2.1) before generalizing to any distribution of samples (see Section 2.2). In Section 3 we extend the algorithm to derive an NUFFT-I, NUFFT-III, and inverse transforms. In Section 4, we describe the two-dimensional analogue of our NUFFT-II.

2. The nonuniform fast Fourier transform of type II. In this section, we describe an $\mathcal{O}(N \log N \log(1/\epsilon)/\log\log(1/\epsilon))$ algorithm to compute the NUDFT-II of size N (see (2)) with a working precision of $0 < \epsilon < 1$. We begin by making the simplifying assumption that the samples $x_0, \ldots, x_{N-1}$ are nearly equispaced before describing the general algorithm.

2.1. Samples are a perturbed equispaced grid. Suppose that the samples $x_0, \ldots, x_{N-1}$ are distributed such that there exists a parameter $0 \le \gamma \le 1/2$ satisfying

$$\left| x_j - \frac{j}{N} \right| \le \frac{\gamma}{N}, \qquad 0 \le j \le N - 1. \tag{5}$$

This assumption guarantees that the closest equispaced point to x_j is j/N, which simplifies the description of our algorithm.

Using the fact that $\omega_k = k$ for $0 \le k \le N - 1$ and properties of the exponential function, we can factor the entries of $\tilde{F}_2$ as

$$(\tilde{F}_2)_{jk} = e^{-2\pi i x_j k} = e^{-2\pi i (x_j - j/N)k} e^{-2\pi i jk/N}, \qquad 0 \le j, k \le N - 1, \tag{6}$$

which shows that the (j, k) entry of $\tilde{F}_2$ can be written as a complex number multiplied by the (j, k) entry of the DFT matrix. The expression in (6) gives us the following

matrix decomposition:

$$\tilde{F}_2 = A \circ F, \qquad A_{jk} = e^{-2\pi i (x_j - j/N)k}, \tag{7}$$

where '$\circ$' is the Hadamard product. The observation in (3) is equivalent to the matrix A being well-approximated by a low rank matrix so that $A \approx A_K = \underline{u}_0 \underline{v}_0^\mathsf{T} + \cdots + \underline{u}_{K-1} \underline{v}_{K-1}^\mathsf{T}$. Since $(\underline{u}\,\underline{v}^\mathsf{T}) \circ F = D_{\underline{u}} F D_{\underline{v}}$, we conclude that

$$\tilde{F}_2 \underline{c} = (A \circ F)\underline{c} \approx (A_K \circ F)\underline{c} = \sum_{r=0}^{K-1} D_{\underline{u}_r} F D_{\underline{v}_r} \underline{c}, \qquad D_{\underline{u}_r} = \mathrm{diag}((\underline{u}_r)_1, \ldots, (\underline{u}_r)_N). \tag{8}$$

Therefore, an approximation to $\tilde{F}_2 \underline{c}$ can be computed in $\mathcal{O}(KN \log N)$ operations via the FFT as each term in the sum in (8) involves diagonal matrices and the DFT matrix. Moreover, each matrix-vector product in the sum can be computed independently and the resulting vectors added together afterwards.

All that remains is to show that A can in fact be well-approximated by a low rank matrix, or equivalently, that K is relatively small, and to construct a low rank approximation A_K for A. We cannot use the singular value decomposition for this[3] because that costs $\mathcal{O}(N^3)$ operations and would dominate the algorithmic complexity of the NUFFT-II. Instead, we note that A can be viewed as a matrix obtained by sampling e^{-ixy} at points in $[-\gamma, \gamma] \times [0, 2\pi]$ and we construct a low rank approximation via an approximation of the function e^{-ixy}.

2.1.1. Low rank approximation via Taylor expansions. A natural way to construct a low rank approximation to A is via Taylor expansion by exploiting the fact that $(x_j - j/N)k$ is relatively small for $0 \le j, k \le N - 1$. The NUFFT developed here is equivalent to [1] (without oversampling) and is numerically unstable. In this direction, consider the Taylor expansion of $e^{-x} = 1 - x + x^2/2 - x^3/6 + \cdots$ about $x = 0$. Applying this Taylor series to each entry of A, we find that for $0 \le j, k \le N-1$

$$A_{jk} = e^{-2\pi i (x_j - j/N)k} = \sum_{r=0}^{\infty} \frac{(-2\pi i (x_j - j/N)k)^r}{r!} \approx \sum_{r=0}^{K-1} \frac{(-2\pi i (x_j - j/N)k)^r}{r!}, \tag{9}$$

where the expansion is truncated after K terms to deliver an approximation. Now, if we let $\underline{x} = (x_0, \ldots, x_{N-1})^\mathsf{T}$, $\underline{e} = (0, 1/N, \ldots, (N-1)/N)^\mathsf{T}$, and $\underline{\omega} = (0, 1, \ldots, N-1)^\mathsf{T}$, then (9) can be applied to each entry of A to find that

$$A = \exp(-2\pi i (\underline{x} - \underline{e})\underline{\omega}^\mathsf{T}) \approx \sum_{r=0}^{K-1} \frac{(-i)^r}{r!} (2\pi (\underline{x} - \underline{e})\underline{\omega}^\mathsf{T})^r = A_K.$$

Here, the notation $\underline{x}\,\underline{y}^\mathsf{T}$ denotes a rank 1 matrix, $\exp(\underline{x}\,\underline{y}^\mathsf{T})$ is the matrix formed by applying the exponential function entry-by-entry to $\underline{x}\,\underline{y}^\mathsf{T}$, and $(\underline{x}\,\underline{y}^\mathsf{T})^r$ is the entry-by-entry rth power of $\underline{x}\,\underline{y}^\mathsf{T}$.

Since $|2\pi i (x_j - j/N)k| \le 2\pi\gamma$ for $0 \le j, k \le N - 1$, error estimates for the truncated Taylor expansion of e^{-x} for $x \in [0, 2\pi\gamma]$ shows that $\|A - A_K\|_{\max} \le \epsilon$ for $K = \mathcal{O}(\log(1/\epsilon))$ [1], where $\|\cdot\|_{\max}$ is the absolute maximum matrix entry. To avoid overflow issues, one should take the vectors $\underline{u}_r = (N(\underline{x} - \underline{e}))^r$ and $\underline{v}_r = (-i)^r (2\pi\underline{\omega}/N)^r / r!$

---

[3]Recall that the truncated singular value decomposition of A, formed by taking the first K singular vectors and values, leads to the best rank K approximation to A in the spectral norm [11].

for $1 \leq r \leq K$ in (8). Unfortunately, we observe that the Taylor-based approach is numerically unstable (even with modest oversampling) in agreement with the experiments in [18, Ex. 3.10]. This is because for moderate K (≥ 7) the matrix A_K is constructed by evaluating high-degree monomial powers. For this reason, the NUFFT-II described in [1] is seldom used. We must construct the matrix A_K in a different way.

2.1.2. Low rank approximation via Chebyshev expansions. One can often stabilize high-degree Taylor expansions by replacing them with Chebyshev expansions. We do that now.

For an integer $p \geq 0$, the Chebyshev polynomial of degree p is given by $T_p(x) = \cos(p \cos^{-1} x)$ on $x \in [-1, 1]$ and the set $\{T_0, T_1, \ldots, T_{K-1}\}$ is an orthogonal basis for the space of polynomials of degree at most $K - 1$, with respect to the weight function $(1 - x^2)^{-1/2}$ on $[-1, 1]$. We can use a Chebyshev series to represent nonperiodic functions, in the same way that a Fourier series can represent periodic functions [27].

In the Appendix in Theorem 2, we derive a low rank approximation for A by using Chebyshev expansions. If $\gamma = 0$, then A is the matrix of all ones and the low rank approximation is trivial. If $\gamma > 0$, then for $0 < \epsilon < 1$ we find an integer K (see (11)) and a matrix A_K such that $\|A - A_K\|_{\max} \leq \epsilon$, where $\|\cdot\|_{\max}$ denotes the absolute maximum matrix entry. The matrix A_K is defined by (see Theorem 2)

$$A_K = \sum_{r=0}^{K-1}{}' \underbrace{\left[\sum_{p=0}^{K-1}{}' a_{pr} \left(\exp\left(-i\pi N(\underline{x} - \underline{e})\right) \circ T_p\left(\tfrac{N(\underline{x}-\underline{e})}{\gamma}\right) \right) \right]}_{=\underline{u}_r} \underbrace{T_r\left(\tfrac{2\underline{\omega}^{\mathsf{T}}}{N} - \mathbf{1}^{\mathsf{T}}\right)}_{=\begin{cases} \underline{v}_r^{\mathsf{T}} & r \geq 1 \\ 2\underline{v}_0^{\mathsf{T}} & r = 0 \end{cases}}, \qquad (10)$$

where $\mathbf{1}$ is the $N \times 1$ vector of ones and the primes on the summands indicate that the first term is halved. The coefficients a_{pr} for $0 \leq p, r \leq K - 1$ are known explicitly as

$$a_{pr} = \begin{cases} 4i^r J_{(p+r)/2}(-\gamma\pi/2) J_{(r-p)/2}(-\gamma\pi/2), & \mod(|p - r|, 2) = 0, \\ 0, & \text{otherwise}, \end{cases}$$

where $J_\nu(z)$ is the Bessel function of parameter ν at z [21, Chap. 10]. Here in (10), $\exp(\underline{x})$ and $T_s(\underline{x})$ denote the exponential and Chebyshev polynomial evaluated at each entry of $\underline{x}$ to form another vector, respectively.

The expansion in (10) provides us with a rank K matrix that approximates A as $A = \lim_{K \to \infty} A_K$. From the convergence properties of Chebyshev expansions, for each fixed K, an explicit upper bound is known for $\|A - A_K\|_{\max}$ (see Appendix A). The vectors $\underline{u}_0, \ldots, u_{K-1}, \underline{v}_0, \ldots, \underline{v}_{K-1}$ in (10) are evaluated via computing the Chebyshev polynomials using a three-term recurrence relation [21, Tab. 18.9.1]. This requires a total of $\mathcal{O}(K^2 N)$ operations. This cost should strictly be included in the final complexity of the NUFFT-II, but we will not include it because this is part of the "planning stage" (see Section 2.3).

In (10) for $\gamma > 0$, the integer K is given by the expression (see Theorem 2)

$$K = \max\left\{3, \left\lceil 5\gamma e^{W(\log(140/\epsilon)/(5\gamma))} \right\rceil\right\} = \mathcal{O}\left(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\right), \qquad (11)$$

where $W(x)$ is the Lambert-W function [21, (4.13.1)], $0 \leq \gamma \leq 1/2$ is the perturbation parameter from (5), and $\lceil x \rceil$ is the nearest integer above or equal to $x \geq 0$. By asymptotic approximations of $W(x)$ as $x \to \infty$, we find that $K =$

	$\gamma = 0$	$0 < \gamma \leq \frac{1}{32}$	$\frac{1}{32} < \gamma \leq \frac{1}{16}$	$\frac{1}{16} < \gamma \leq \frac{1}{8}$	$\frac{1}{8} < \gamma \leq \frac{1}{4}$	$\frac{1}{4} < \gamma \leq \frac{1}{2}$
double	1	8	9	11	13	16
single	1	5	6	7	8	10
half	1	3	3	4	5	7

Table 1: When the samples $x_0, \ldots, x_{N-1}$ are perturbed equispaced samples with respect to a parameter $0 \leq \gamma \leq 1/2$ (see (5)) the NUDFT-II can be decomposed as $\tilde{F}_2 = A \circ F$, where F is the DFT matrix and A can be approximated by a rank K matrix, up to a working accuracy of $0 < \epsilon < 1$. We give the values of K that we use in (10) for various values of γ (see (5)) and working accuracies: 1st row $\epsilon \approx 2.2 \times 10^{-16}$, 2nd row $\epsilon \approx 1.2 \times 10^{-7}$, and last row $\epsilon \approx 9.8 \times 10^{-4}$. Our NUFFT-II roughly costs K FFTs of size N, though these can be performed in parallel.

$\mathcal{O}(\log(1/\epsilon)/\log\log(1/\epsilon))$ as $\epsilon \to 0$ [21, (4.13.10)] and hence, $\tilde{F}_2 \underline{c}$ can be computed in a total of $\mathcal{O}(N \log N \log(1/\epsilon)/\log\log(1/\epsilon))$ operations using (8).

It is relatively common in practice to have perturbed equispaced samples so we always compute the parameter $0 \leq \gamma \leq 1/2$ in (5) in order to select the smallest possible integer K with $\|A - A_K\|_{\max} \leq \epsilon$. In our implementation of the NUFFT-II, we do not use the formula for K in (11) because it is only asymptotically sharp and the constants are not tight. Instead, in (10) we use the values of K given in Table 1, which are selected from empirical observations. In particular, in double precision we use at most $K = 16$, corresponding to the cost of an NUFFT-II being approximately 16 FFTs of size N.

In practice, it is also common to not always need a working precision of $\epsilon \approx 2.2 \times 10^{-16}$ so we adaptively select the integer K based on that parameter too. For example, with a working precision of 9.8×10^{-4} the NUFFT-II costs at most seven FFTs of size N.

2.2. Arbitrarily distributed samples. Suppose that the samples $x_0, \ldots, x_{N-1}$ in (2) are arbitrarily distributed real numbers. The properties of the complex exponential, $x \mapsto e^{-2\pi i x k}$ for $0 \leq k \leq N - 1$, allow us to assume, without loss of generality, that the samples are in the interval $[0, 1)$; otherwise, they can be translated to that interval using periodicity. For convenience, in this section we assume that $x_0, \ldots, x_{N-1} \in [0, 1)$, though our implementation does not have this restriction. In this general setting, the observation in (3) is no longer valid because the samples are arbitrarily distributed.

Instead, define a sequence $s_0, \ldots, s_{N-1}$ that takes values from $\{0, \ldots, N\}$ and is defined so that s_j/N is the closest node to x_j from an equispaced grid of size $N + 1$ (ties can be broken arbitrarily). Since each x_j is a distance of at most $1/(2N)$ from these equispaced nodes, we have

$$\left| x_j - \frac{s_j}{N} \right| \leq \frac{1}{2N}, \qquad 0 \leq j \leq N - 1. \tag{12}$$

Figure 1 illustrates this process when $N = 8$. The sequence can be easily computed via the relationship $\underline{s} = \text{round}(N\underline{x})$, where $\text{round}(N\underline{x})$ returns the nearest integer to each entry of the vector $N\underline{x}$.

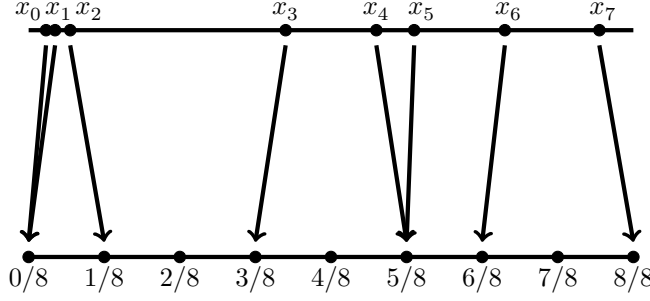If $s_j = N$ then we must reassign x_j because the uniform DFT does not contain

Fig. 1: An illustration of how nonuniform samples on $[0, 1]$ are assigned to the closest equispaced grid point for $N = 8$. In this example, the sequence $s_0, \ldots, s_{N-1}$ takes the values $s_0 = s_1 = s_2 = 0$, $s_3 = 3$, $s_4 = s_5 = 5$, $s_6 = 6$, and $s_7 = 8$. Here $0 \leq x_0 \leq x_1 \leq \cdots \leq x_{N-1} < 1$, but the samples do not necessarily need to be ordered. Since $s_7 = 8$, the sample x_7 is later assigned to the equispaced point at 0 (see (13)).

a sample at $s_j/N = 1$. Using the periodicity of the complex exponential, we use the identity $e^{-2\pi i s_j k/N} = e^{-2\pi i 0 k/N} = 1$ to assign x_j to the equispaced node at 0. This can be done simply by defining another sequence $t_0, \ldots, t_{N-1}$, which takes values from $\{0, \ldots, N-1\}$, and is given by

$$t_j = \begin{cases} s_j, & 0 \leq s_j \leq N - 1, \\ 0, & s_j = N. \end{cases} \tag{13}$$

In practice, one can easily compute the vector $\underline{t}$ directly from $\underline{x}$ since

$$\underline{t} = \mathrm{mod}(\mathrm{round}(N\underline{x}), N),$$

where $\mathrm{mod}(\underline{a}, N)$ is the modulo-N operation on each entry of $\underline{a}$.

From the properties of the exponential function and the definition of $t_0, \ldots, t_{N-1}$, we find that

$$(\tilde{F}_2)_{jk} = e^{-2\pi i x_j k} = e^{-2\pi i (x_j - s_j/N)k} e^{-2\pi i s_j k/N} = e^{-2\pi i (x_j - s_j/N)k} e^{-2\pi i t_j k/N}. \tag{14}$$

This means that the (j, k) entry of $\tilde{F}_2$ can be expressed as a product of $e^{-2\pi i (x_j - s_j/N)k}$ and the (t_j, k) entry of F for $0 \leq j, k \leq N - 1$. Equivalently, by setting $A_{jk} = e^{-2\pi i (x_j - s_j/N)k}$ for $0 \leq j, k \leq N - 1$, we can write (14) as the following matrix decomposition:

$$\tilde{F}_2 = A \circ F(\underline{t}, :), \qquad (F(\underline{t}, :))_{jk} = e^{-2\pi i t_j k/N}, \quad 0 \leq j, k \leq N - 1,$$

where $\underline{t} = (t_0, \ldots, t_{N-1})^\mathsf{T}$. Note that $F(\underline{t}, :)$ denotes the matrix formed by extracting the rows indexed by $(t_0, \ldots, t_{N-1})$ from the DFT matrix.

Since $N(x_j - s_j/N) \in [-1/2, 1/2]$ and $k/N \in [0, 1]$, we find that A can be well-approximated by a low rank matrix using the same idea as in Section 2.1.2. This leads to the rank K approximation A_K to A, given by

$$A_K = \sum_{r=0}^{K-1}{}' \underbrace{\left[\sum_{p=0}^{K-1}{}' a_{pr} \left(\exp\left(-i\pi N(\underline{x} - \underline{s}/N)\right) \circ T_p(2N(\underline{x} - \underline{s}/N)) \right) \right]}_{= \underline{u}_r} \underbrace{T_r\left(\tfrac{2\underline{\omega}^\mathsf{T}}{N} - \mathbf{1}^\mathsf{T}\right)}_{= \begin{cases} \underline{v}_r^\mathsf{T} & r \geq 1 \\ 2\underline{v}_0^\mathsf{T} & r = 0 \end{cases}},$$

7

where $\underline{s} = (s_0, \ldots, s_{N-1})^\mathsf{T}$. Here, $\|A - A_K\|_{\max} \leq \epsilon$ for some $0 < \epsilon < 1$ and K is the value in (11) with $\gamma = 1/2$.

In summary, we find that the matrix-vector product, $\tilde{F}_2 \underline{c}$, can be approximately computed with a working accuracy of $0 < \epsilon < 1$ via the approximation

$$\tilde{F}_2 \underline{c} = (A \circ F(\underline{t}, :)) \, \underline{c} \approx (A_K \circ F(\underline{t}, :)) \, \underline{c} = \sum_{r=0}^{K-1} D_{\underline{u}_r} F(\underline{t}, :) D_{\underline{v}_r} \underline{c}. \tag{15}$$

This leads to an $\mathcal{O}(N \log N \log(1/\epsilon)/\log\log(1/\epsilon))$ complexity NUFFT-II because: (1) The matrix-vector products with the diagonal matrices $D_{\underline{u}_r}$ and $D_{\underline{v}_r}$ can be performed in $\mathcal{O}(N)$ operations, and (2) The matrix-vector product $F(\underline{t}, :)\underline{c}$ can be computed in $\mathcal{O}(N \log N)$ operations via the FFT and the relationship $F(\underline{t}, :)\underline{c} = I_N(\underline{t}, :)F\underline{c}$, where I_N is the $N \times N$ identity matrix and $I_N(\underline{t}, :)$ denotes the matrix obtained by extracting the $(t_0, \ldots, t_{N-1})$ rows of the identity matrix. Again, each term in the sum in (15) can be computed in parallel and the resulting vectors added together afterwards.

2.3. Algorithmic details. There are a handful of algorithmic details.

- **Oversampling:** In Section 2.2, we assign N samples $x_0, \ldots, x_{N-1}$ to an equispaced grid of size N. The process of oversampling, which occurs in many other NUFFTs, translates N samples to an equispaced grid of size M, where $M > N$. In our setting, this results in FFTs of size M in (15) with potentially a smaller integer K because $|x_j - s_j/M| \leq 1/(2M) < 1/(2N)$. Naively, since our algorithm is numerically stable without oversampling, it would seem that oversampling is never beneficial for us. For example, in double precision if $M = 2N$, then 13 FFTs of size $2N$ are required (see Table 1) instead of 16 FFTs of size N. In practice, it is a little more complicated as one may benefit from selecting an integer $N \leq M < 2N$ that has a convenient prime factorization for the FFT [8]. We have not explored this possibility yet.

- **Vectorization:** One can vectorize the FFTs in (15) by computing $\tilde{F}_2\underline{c}$ in two steps:

$$(\text{Step 1}) \quad X = I_N(\underline{t}, :)F \left[D_{\underline{v}_0}\underline{c} \, | \, \cdots \, | \, D_{\underline{v}_{K-1}}\underline{c} \right],$$

$$(\text{Step 2}) \quad \tilde{F}_2\underline{c} = \left[D_{\underline{u}_0}X_0 \, | \, \cdots \, | \, D_{\underline{u}_{K-1}}X_{K-1} \right] \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix},$$

where X_k denotes the kth column of X. In the programming language Julia [6] this can be implemented in the one-liner:

```
nufft2(c) = (U.*(fft(Diagonal(c)*V,1)[t+1,:]))*ones(K),
```

where $\mathtt{U} = \left[\underline{u}_0 \, | \, \cdots \, | \, \underline{u}_{K-1} \right]$, $\mathtt{V} = \left[\underline{v}_0 \, | \, \cdots \, | \, \underline{v}_{K-1} \right]$, and the variable $\mathtt{t}$ is the vector $\underline{t}$.

- **Planning the transform:** Most implementations of fast transforms these days have a *planning stage* [13], where ancillary quantities are computed that do not depend on the entries of $\underline{c}$. This stage may also involve memory allocation and the finalization of recursion details [13]. For our NUFFT-II, the planning stage consists of computing γ, $\underline{t}$, and K, planning the FFTs [13], as well as computing the vectors $\underline{u}_0, \ldots, \underline{u}_{N-1}, \underline{v}_0, \ldots, \underline{v}_{N-1}$ for the low rank approximation A_K. These quantities and data structures are then stored in memory so that the NUFFT-II is computationally faster. After the planning stage of our NUFFT-II, there is an *online stage*, where the transform is essentially the one-liner for the $\mathtt{nufft2(c)}$ call above. It is particularly important to plan a NUFFT-II when the matrix-vector product with $\tilde{F}_2$ is desired for many vectors.

8

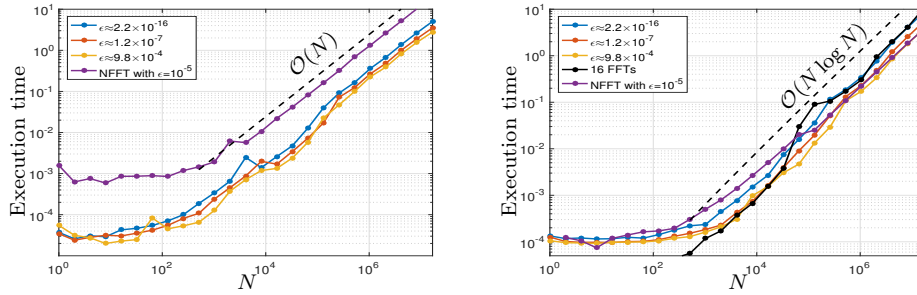Fig. 2: Left: The computational cost of planning our NUFFT-II transform for $\epsilon \approx 2.2 \times 10^{-16}$, $\epsilon \approx 1.2 \times 10^{-7}$, and $\epsilon \approx 9.8 \times 10^{-4}$ as well as the planning cost of the Julia implementation of the NFFT transform [22, 17]. Right: The computational cost of computing our NUFFT-II after planning for $\epsilon \approx 2.2 \times 10^{-16}$, $\epsilon \approx 1.2 \times 10^{-7}$, and $\epsilon \approx 9.8 \times 10^{-4}$. For comparison we have include the cost of the Julia implementation of the NFFT transform after planning [17] and the cost of 16 FFTs after planning.

2.4. Numerical results. We have two different implementations of the transforms in this paper: (1) A MATLAB implementation, where the NUFFT-II transform is assessable via the `chebfun.nufft` command in Chebfun [9],[4] and (2) A Julia implementation, which is publicly available via the `nufft2` command in the `FastTransforms.jl` package [24]. Since the dominating computational cost of our transforms are FFTs, and these are computed via the FFTW library [13], the cost of our algorithms are approximately the same in MATLAB and Julia.[5]

Recall that there are two stages of the transform: (1) A planning stage in which ancillary quantities are computed (see Section 2.3) and (2) An online stage, where the transform needs knowledge of the vector $\underline{c}$ in (2) and the desired vector $\underline{f}$ is computed. When the same NUFFT-II transform is applied to multiple vectors, the planning stage is only performed once while the online stage is executed for every new vector.

Figure 2 (left) shows the execution times[6] of the NUFFT-II transform in both the planning stage and the online stage the NUFFT-II (right). The online stage of the NUFFT-II is approximately 16 FFTs in double precision, as expected from Table 1, and takes approximately 8 seconds to compute the transform when N is 16 million. Figure 2 shows that our NUFFT-II is competitive to the Julia implementation of the NFFT software [17].

Figure 3 (left) demonstrates the execution times of the online stage of our NUFFT-II for samples that are perturbed equispaced grids with $\gamma = 1/2$, $\gamma = 1/8$, $\gamma = 1/32$, and $\gamma = 0$ (see (5)). For definitiveness, we chose the samples to be the so-called *worst*

---

[4]Note to reviewer: The code is currently publicly available through GitHub, but is still under code review. It will hopefully appear in the next release of Chebfun.

[5]By default the `fft` command in MATLAB has multithreading capabilities. To see a similar performance in Julia, one must execute the command `FFTW.set_num_threads(n)`, where `n` is an appropriate number of threads.

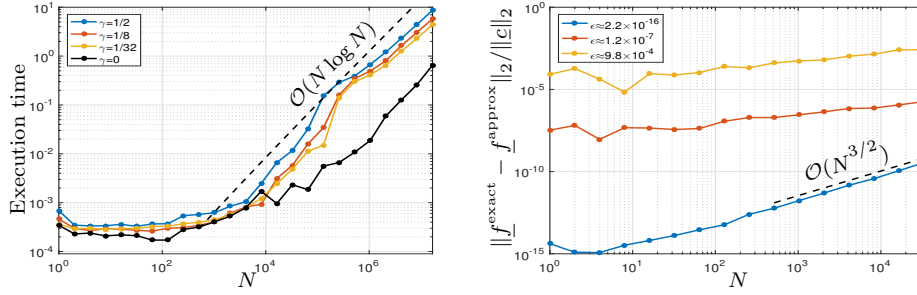[6]Computational results were performed on an Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz in Julia v0.5.0.

Fig. 3: Left: The computational cost of our NUFFT-II transform with $\epsilon = 2.2 \times 10^{-16}$ for $\gamma = 1/2$, $\gamma = 1/8$, $\gamma = 1/32$, and $\gamma = 0$ after planning (see (5)). When $\gamma = 0$, our NUFFT-II is essentially one FFT. Right: The accuracy of the NUFFT-II transform for working accuracies of $\epsilon \approx 2.2 \times 10^{-16}$, $\epsilon \approx 1.2 \times 10^{-7}$, and $\epsilon \approx 9.8 \times 10^{-4}$. Here, the vector of Fourier coefficients $\underline{c}$ are realizations of standard Gaussian random variables, $\underline{f}^{\text{exact}} = \tilde{F}_2 \underline{c}$ is the vector computed using the naive $\mathcal{O}(N^2)$ algorithm in high precision (using `BigFloat` in Julia), and $\underline{f}^{\text{approx}}$ is the vector obtained from our NUFFT-II transform with the stated working accuracies.

grid for each γ in the NUFFT-II (see [2, Sec. 3.3.1]), i.e.,

$$
x_j = \begin{cases} (j + \gamma)/N, & 0 \leq j \leq \lfloor N/2 \rfloor, \\ (j - \gamma)/N, & \text{otherwise}. \end{cases}
$$

We see that the NUFFT-II is more computationally efficient when the samples are closer to an equispaced grid, as expected from the values of K in Table 1.

Our NUFFT-II relies on a matrix approximation; namely, the approximation of the matrix A in (7) by a low rank approximation A_K. Therefore, if $\underline{f}^{\text{exact}}$ is the vector calculated from $\underline{f}^{\text{exact}} = \tilde{F}_2 \underline{c} = (A \circ F)\underline{c}$, then our algorithm calculates the approximation $\underline{f}^{\text{approx}} = (A_K \circ F)\underline{c}$. The incurred error can be simply bounded as follows:

$$
\begin{aligned}
\|\underline{f}^{\text{exact}} - \underline{f}^{\text{approx}}\|_2 &= \|((A - A_K) \circ F)\underline{c}\|_2 \\
&\leq \|((A - A_K) \circ F)\|_2 \|\underline{c}\|_2 \\
&\leq \|((A - A_K) \circ F)\|_{\text{F}} \|\underline{c}\|_2 \\
&\leq \|A - A_K\|_{\max} \|F\|_{\text{F}} \|c\|_2 \\
&\leq N\epsilon \|\underline{c}\|_2,
\end{aligned}
$$

where $\|\cdot\|_{\text{F}}$ denotes the matrix Frobenius norm and the last inequality follows from the fact that $\|A - A_K\|_{\max} \leq \epsilon$ and $\|F\|_{\text{F}} = N$. In Figure 3 (right) we observe that the relative error $\|\underline{f}^{\text{exact}} - \underline{f}^{\text{approx}}\|_2 / \|\underline{c}\|_2$ grows like $\mathcal{O}(N^{3/2})$, where the extra $\mathcal{O}(N^{1/2})$ is probably due to the fact that a sum of N Gaussian random variable is of size $\mathcal{O}(N^{1/2})$. When we repeat the experiment with a random vector $\underline{c}$ with $\mathcal{O}(n^{-2})$ decay, i.e., `c = randn(N)./(1:N).^2` in Julia, the relative error $\|\underline{f}^{\text{exact}} - \underline{f}^{\text{approx}}\|_2 / \|\underline{c}\|_2$ grows like $\mathcal{O}(N)$. More often than not, Fourier coefficients do decay as the coefficients are derived from expanding a smooth periodic function.

3. Other nonuniform fast Fourier transforms. Many other nonuniform discrete Fourier transforms are related to the NUDFT-II including: (1) The NUDFT-I,

(2) NUDFT-III, (3) inverse NUDFTs. We describe these transforms in this section.

3.1. The nonuniform fast Fourier transform of Type I. The NUDFT-I transform computes the vector $\underline{f}$, given the vector $\underline{c} \in \mathbb{C}^{N \times 1}$ and frequencies $\underline{\omega} \in [0, N]$, such that

$$f_j = \sum_{k=0}^{N-1} c_k e^{-2\pi i \frac{j}{N} \omega_k}, \qquad 0 \le j \le N - 1.$$

It is equivalent to evaluating a generalized Fourier series at equispaced points and computing the matrix-vector product $\tilde{F}_1 \underline{c}$, where $(\tilde{F}_1)_{jk} = e^{-2\pi i(j/N)\omega_k}$ for $0 \le j, k \le N - 1$.

For this transform, we immediately find that

$$(\tilde{F}_1)_{jk} = e^{-2\pi i \frac{j}{N} \omega_k} = e^{-2\pi i \frac{\omega_k}{N} j} = (\tilde{F}_2)_{kj} \qquad 0 \le j, k \le N - 1,$$

where the frequencies $\omega_0/N, \ldots, \omega_{N-1}/N$ act as nonequispaced sampled in a NUDFT-II. Therefore, we see that the NUDFT-I matrix is equivalent to a transposed NUDFT-II matrix. Since the transpose of a sum of matrices is equal to the sum of the individual terms transposed, (15) immediately leads to our NUFFT-I:

$$\tilde{F}_1 \underline{c} = \tilde{F}_2^\mathsf{T} \underline{c} \approx \sum_{r=0}^{K-1} D_{\underline{v}_r} F_2^\mathsf{T} I_N(:, \underline{t}) D_{\underline{u}_r} \underline{c} = \sum_{r=0}^{K-1} D_{\underline{v}_r} F_2^\mathsf{T} I_N(:, \underline{t}) D_{\underline{u}_r} \underline{c}.$$

Therefore, $\tilde{F}_1 \underline{c}$, can be computed in $\mathcal{O}(N \log N \log(1/\epsilon)/\log\log(1/\epsilon))$ operations using the relationship $F^\mathsf{T} \underline{c} = N\overline{F}^{-1} \underline{c}$ and the inverse FFT.

For implementations of this transform, see the `chebfun.nufft` command in Chebfun [9] and the `nufft1` command in `FastTransforms.jl` [24].

3.2. The nonuniform fast Fourier transform of Type III. Let $x_0, \ldots, x_{N-1} \in [0, 1)$ be samples and $\omega_0, \ldots, \omega_{N-1} \in [0, N)$ be frequencies. Suppose that we wish to compute the vector $\underline{f}$ in (1), given $c_0, \ldots, c_{N-1}$. This is equivalent to computing the matrix-vector product $\tilde{F}_3 \underline{c}$, where $(\tilde{F}_3)_{jk} = e^{-2\pi i x_j \omega_k}$. From the properties of the exponential function, the sequence $s_0, \ldots, s_{N-1}$ in (12), and the sequence $t_0, \ldots, t_{N-1}$ in (13), we can write

$$(\tilde{F}_3)_{jk} = e^{-2\pi i x_j \omega_k} = e^{-2\pi i(x_j - s_j/N)w_k} e^{-2\pi i \frac{s_j - t_j}{N} \omega_k} \underbrace{e^{-2\pi i \frac{t_j}{N} \omega_k}}_{=(\tilde{F}_1)(\underline{t},:)_{jk}}, \qquad 0 \le j, k \le N - 1.$$

Applying the product above to every entry of $\tilde{F}_3$ leads to the following matrix decomposition:

$$\tilde{F}_3 = A \circ B \circ \tilde{F}_1(\underline{t}, :), \qquad A_{jk} = e^{-2\pi i(x_j - s_j/N)w_k}, \qquad B_{jk} = e^{-2\pi i \frac{s_j - t_j}{N} \omega_k},$$

where $\tilde{F}_1(\underline{t}, :)$ denotes the NUDFT-I matrix permuted by the sequence $t_0, \ldots, t_{N-1}$.

Since $|N(x_j - s_j/N)| \le 1/2$ for $0 \le j \le N-1$ and $\omega_k/N \in [0, 1)$ for $0 \le k \le N-1$, we know from Theorem 2 that A can be approximated by a rank K matrix A_K such that $\|A - A_K\|_{\max} \le \epsilon$ and $K = \mathcal{O}(\log(1/\epsilon)/\log\log(1/\epsilon))$, where $0 < \epsilon < 1$ is a working precision. Moreover, the matrix B is of rank at most 2 since

$$B = (\mathbf{1} - (\underline{s} - \underline{t})/N)\mathbf{1}^\mathsf{T} + (\underline{s} - \underline{t})/N \exp(-2\pi i \underline{\omega}^\mathsf{T}),$$

where $\mathbf{1}$ is the $N \times 1$ column vector of ones. Therefore, $A \circ B$ can be well-approximated by a rank $\mathcal{O}(K)$ matrix and hence, $\tilde{F}_3 \underline{c}$ can be computed in $\mathcal{O}(KN \log N)$ operations.

In double precision, the cost of this NUFFT-III is at most $32(= 16 \times 2)$ NUFFT-I's or, equivalently, $512(= 32 \times 16)$ FFTs of size N. These FFTs can all still be performed in parallel. In the case when the sequences $s_0, \ldots, s_{N-1}$ and $t_0, \ldots, t_{N-1}$ are the same, which often occurs (see (13)), the matrix B is the matrix of all ones. In this situation, $A \circ B = A$ and the cost of the NUFFT-III is reduced by a factor of 2.

This transform is available in the `chebfun.nufft` command in Chebfun [9].

3.3. Inverse nonuniform fast Fourier transforms. In the NUFFT-I, -II, and -III, severely nonequispaced samples or noninteger frequencies were not a numerical issue and the parameter γ in (5) only mildly affected the computational cost of the transform. For the inverse transforms, nonuniform samples or frequencies are far more detrimental in terms of both accuracy and computational cost.

The inverse NUDFT-II requires that the linear system $\tilde{F}_2 \underline{c} = \underline{f}$ is solved for the vector $\underline{c}$, where $\tilde{F}_2$ is given in (2). Here, we will assume that the samples $x_0, \ldots, x_{N-1}$ are perturbed equispaced samples with $0 \le \gamma < 1/2$ (see (5)) to ensure that $\tilde{F}_2^{-1}$ exists. Since we have a fast matrix-vector product for $\tilde{F}_2$ (see Section 2), one naturally tries a variety of Krylov methods. After trying several of them, we advocate the following approach based on the conjugate gradient method (CG).

The matrix $\tilde{F}_2$ is not a positive definite matrix, i.e., it is not symmetric with positive eigenvalues, so the conjugate gradient method cannot be immediately applied. Instead, we use the conjugate gradient method on the normal equations: $\tilde{F}_2^* \tilde{F}_2 \underline{c} = \tilde{F}_2^* \underline{f}$. By considering the (j,k) entry of $\tilde{F}_2^* \tilde{F}_2$, we find that it only depends on the value of $j - k$:

$$(\tilde{F}_2^* \tilde{F}_2)_{jk} = \sum_{p=0}^{N-1} e^{2\pi i x_p (j-k)}, \qquad 0 \le j, k \le N - 1.$$

Hence, $\tilde{F}_2^* \tilde{F}_2$ is a Toeplitz matrix, i.e., a matrix with constant diagonal entries, as noted previously in [10]. Therefore, a matrix-vector product with $\tilde{F}_2^* \tilde{F}_2$ can be computed using a fast Toeplitz multiply, costing just one FFT and one inverse FFT of size $2N$ [14, Sec. 4.7.7].[7]

Let the number of conjugate gradient iterations be denoted by R_{CG}. Since CG requires one matrix-vector product per iteration, the inverse transform costs the same as $2R_{\mathrm{CG}}$ FFTs of size $2N$, ignoring $\mathcal{O}(K)$ FFTs to compute $\tilde{F}_2 \underline{f}$ and the calculation of the eigenvalues of a circulant matrix (see [14, Sec. 4.7.7]). Therefore, this iterative method leads to an inverse NUFFT-II with a computational cost of $\mathcal{O}(R_{\mathrm{CG}} N \log N)$ operations, which is quasi-optimal provided that R_{CG} does not grow too quickly with N.

Figure 4 shows that empirically R_{CG} is observed to be small and, perhaps, bounded with N when $0 < \gamma < 1/4$.[8] When the samples are uniformly sampled, $\tilde{F}_2 = F$ and $R_{\mathrm{CG}} = 1$. As the perturbation parameter, γ, is increased from 0 to $1/2$, the condition number of $\tilde{F}_2$ — and hence R_{CG} — can increase without bound. For example, when $\gamma = 1/2$, the samples may not be distinct and so $\tilde{F}_2^{-1}$ may not exist.

---

[7]Note that the first column and row of $\tilde{F}_2^* \tilde{F}_2$ are the same due to symmetry and the first column of $\tilde{F}_2^* \tilde{F}_2$ can be obtained via the relation $\tilde{F}_2^* \tilde{F}_2 \underline{e}_1$, where $\underline{e}_1$ is the first canonical vector.

[8]The Kadec-1/4 theorem from the literature on the theory of frames [16] (also see [2, Thm 3.1]) makes us believe that R_{CG} remains bounds as $N \to \infty$ with $0 \le \gamma < 1/4$, but grows with N when $1/4 \le \gamma < 1/2$.
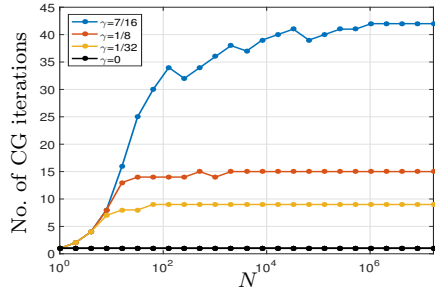
Fig. 4: The number of conjugate gradient (CG) iterations required for the inverse NUFFT-II transform with a convergence tolerance of $\epsilon \approx 2.2 \times 10^{-14}$ and $\gamma = 7/16$, $\gamma = 1/8$, $\gamma = 1/32$, and $\gamma = 0$ (see (5)). When $\gamma = 0$, the inverse NUFFT-II is the inverse FFT and only one CG iteration is required. The inverse NUFFT-II has a cost that is like an $\mathcal{O}(N \log N)$ algorithm for all practical N. This paper does not provide a bound on R_{CG}.

To fully understand the algorithmic complexity of our inverse NUFFT-II, we need to bound R_{CG}. One can do this immediately if a bound on the condition number of $\tilde{F}_2$ is known. The recent theoretical work on the Lebesgue constant for trigonometric interpolation with nonequispaced points in [2, 3] is potentially helpful for bounding the condition number of $\tilde{F}_2$; however, we have not been able to derive a bound in terms of γ on this yet.

An analogous idea applies $\tilde{F}_1$ to derive an inverse NUFFT-I because $\tilde{F}_1^{-1} = \tilde{F}^*(\tilde{F}_1\tilde{F}_1^*)^{-1}$ and $\tilde{F}_1\tilde{F}_1^*$ is a Toeplitz matrix (see, also, [10]). The inverse transforms are implemented in the `chebfun.inufft` command in Chebfun [9] and the `inufft` commands in `FastTransforms.jl` [24].

4. The two-dimensional nonuniform fast Fourier transform of type II. Given an $m \times n$ matrix of Fourier coefficients $C \in \mathbb{C}^{m \times n}$ and nonuniform samples $(x_0, y_0), \ldots, (x_{N-1}, y_{N-1}) \in \mathbb{R}^2$, the two-dimensional NUDFT-II is the task of computing the following vector:

$$f_j = \sum_{k_1=0}^{m-1} \sum_{k_2=0}^{n-1} C_{k_1 k_2} e^{-2\pi i (k_1 x_j + k_2 y_j)}, \qquad 0 \leq j \leq N - 1. \tag{16}$$

Naively, this requires $\mathcal{O}(Nmn)$ operations since there are N sums with each sum contain mn terms. Here, we describe an algorithm that requires only $\mathcal{O}(mn(\log(n) + \log(m)) + N)$ operations.

It is helpful to start by reviewing the uniform two-dimensional FFT, which computes the vector $\underline{f} \in \mathbb{C}^{mn \times 1}$ (by default $N = mn$) such that

$$f_j = \sum_{k_1=0}^{m-1} \sum_{k_2=0}^{n-1} C_{k_1 k_2} e^{-2\pi i (k_1 \lfloor j/m \rfloor / n + k_2 \bmod(j, m)/m)}, \qquad 0 \leq j \leq mn - 1. \tag{17}$$

The samples in (17) lie on the $m \times n$ equispaced tensor grid $(s/m, t/n)$ for $0 \leq s \leq m-1$ and $0 \leq t \leq n-1$. In the Julia language, the vector $\underline{f}$ in (17) can be computed by the command `fft(C)[:]` in $\mathcal{O}(mn(\log m + \log n))$ operations.
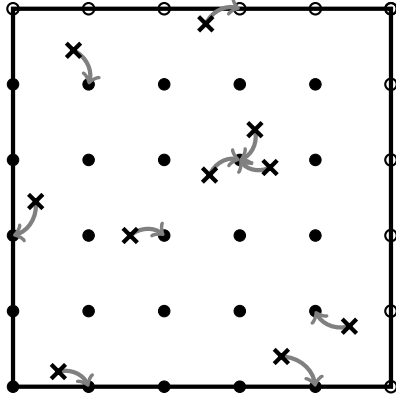
13

Fig. 5: An illustration of how nonuniform samples in $[0,1] \times [0,1]$ are assigned to their nearest $m \times n$ equispaced grid point. The sequence $(s_0^x, s_0^y), \ldots, (s_{N-1}^x, s_{N-1}^y) \in \mathbb{N}^2$ is defined so that $(s_j^x/n, s_j^y/m)$ is the closest $m \times n$ equispaced grid point to (x_j, y_j) for $0 \leq j \leq N-1$. If (x_j, y_j) is assigned to one of the grid points denoted by one of the unfilled black circles, i.e., either $s_j^x = n$ or $s_j^y = m$ or both, then the sample is reassigned using the periodicity of the complex exponential function (see (**??**)).

As in Section 2.2, we first define a sequence $(s_0^x, s_0^y), \ldots, (s_{N-1}^x, s_{N-1}^y) \in \mathbb{N}^2$ such that $(s_j^x/n, s_j^y/m)$ is the closest point from an $m \times n$ equispaced grid to (x_j, y_j) for $0 \leq j \leq N-1$. By definition, we have

$$\left| x_j - \frac{s_j^x}{n} \right| \leq \frac{1}{2n}, \qquad \left| y_j - \frac{s_j^y}{m} \right| \leq \frac{1}{2m}, \qquad 0 \leq j \leq N-1.$$

REFERENCES

[1] C. ANDERSON AND M. D. DAHLEH, *Rapid computation of the discrete Fourier transform*, SIAM J. Sci. Comput., 17 (1996), pp. 913–919.

[2] A. AUSTIN, *Some New Results on and Applications of Interpolation in Numerical Computation*, DPhil thesis, University of Oxford, 2016.

[3] A. AUSTIN AND L. N. TREFETHEN, *Trigonometric interpolation and quadrature in perturbed points*, arXiv preprint arXiv:1612.04018, (2016).

[4] S. BAGCHI AND S. K. MITRA, *The nonuniform discrete Fourier transform and its applications in signal processing*, vol. 463, Springer Science and Business Media, 2012.

[5] R. BEATSON AND L. GREENGARD, *A short course on fast multipole methods*, Wavelets, multilevel methods and elliptic PDEs, 1 (1997), pp. 1–37.

[6] J. BEZANSON, A. EDELMAN, S. KARPINSKI, AND V. B. SHAH, *Julia: A fresh approach to numerical computing*, arXiv preprint arXiv:1411.1607, 2016.

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

[8] J. W. Cooley and J. W. Tukey, *An algorithm for the machine calculation of complex Fourier series*, Math. Comput., 19 (1965), pp. 297–301.

[9] T. A. Driscoll, N. Hale, and L. N. Trefethen, editors, *Chebfun Guide*, Pafnuty Publications, Oxford, 2014.

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

[11] C. Eckart and G. Young, *The approximation of one matrix by another of lower rank*, Psychometrika, 1 (1936), pp. 211–218.

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

[13] M. Frigo and S. G. Johnson, *FFTW: An adaptive software architecture for the FFT*, Acoustics, Speech and Signal Processing, Proceedings of the 1998 IEEE International Conference on., Vol. 3. IEEE, 1998.

[14] G. Golub and C. R. Van Loan, *Matrix Computations*, Johns Hopkins University Press, 1996.

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

[16] M. I. Kadec, *The exact value of the Paley-Wiener constant*, Soviet Math. Dokl., 5 (1964), pp. 559–561.

[17] T. Knopp, et al., https://github.com/tknopp/NFFT.jl/graphs/contributors, Jan. 2017.

[18] S. Kunis, *Nonequispaced FFT: Generalisation and Inversion*, Universität zu Lübeck, 2006.

[19] S. Kunis and I. Melzer, *Fast evaluation of real and complex exponential sums*, Preprint, Univ. Osnabrück, 2014.

[20] J.-Y. Lee and L. Greengard, *The type 3 nonuniform FFT and its applications*, J. Comput. Phys., 206 (2005), pp. 1–5.

[21] F. W. J. Olver, D. W. Lozier, R. F. Boisvert, and C. W. Clark, *NIST Handbook of Mathematical Functions*, Cambridge University Press, 2010.

[22] D. Potts and G. Steidl, *Fast summation at nonequispaced knots by NFFT*, SIAM J. Sci. Comput., 24 (2003), pp. 2013–2037.

[23] D. Potts, S. Steidl, and M. Tasche, *Fast Fourier transforms for nonequispaced data: A tutorial*, Modern sampling theory., Birkhäuser, Boston, (2001), pp. 247–270.

[24] R. M. Slevinsky, et al., https://github.com/MikaelSlevinsky/FastTransforms.jl, v0.0.6, 2016.

[25] A. Townsend, *Computing with functions in two dimensions*, DPhil thesis, University of Oxford, 2014.

[26] A. Townsend, M. Webb, and S. Olver, *Fast polynomial transforms based on Toeplitz and Hankel matrices*, arXiv preprint arXiv:1604.07486, (2016).

[27] L. N. Trefethen, *Approximation Theory and Approximation Practice*, SIAM, 2013.

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

Appendix A. Constructing a low rank matrix approximation via a bivariate Chebyshev expansion. In Section 2.1 we require a low rank approximation for the matrix A in (7). We first note that we can consider A as the matrix obtained by sampling the bivariate function $(x, y) \mapsto e^{-ixy}$ on the domain $[-\gamma, \gamma] \times [0, 2\pi]$. If we construct a bivariate polynomial approximation $q(x, y)$ of degree $K - 1$ in both the x- and y-variable to e^{-ixy} on $[-\gamma, \gamma] \times [0, 2\pi]$, then

$$A = e^{-2\pi i (\underline{x} - \underline{e}) \underline{\omega}^{\mathsf{T}}} \approx q\left(N(\underline{x} - \underline{e}), \frac{\underline{\omega}^{\mathsf{T}}}{N}\right) = A_K$$

is a rank K approximation to A [25, Sec. 3.1]. We construct the polynomial $q(x, y)$ by a truncated bivariate Chebyshev expansion for e^{-ixy}.

LEMMA 1. *Let $0 < \epsilon < 1$ be a working precision and $\gamma > 0$. The following holds:*

$$\sup_{(x,y) \in [-\gamma, \gamma] \times [0, 2\pi]} \left| e^{-ixy} - \sum_{r=0}^{K-1}{}' \sum_{p=0}^{K-1}{}' a_{pr} e^{-i\pi x} T_p\left(\frac{x}{\gamma}\right) T_r\left(\frac{y}{\pi} - 1\right) \right| \le \epsilon,$$

where T_p is the degree p Chebyshev polynomial, the a_{pq} coefficients are given in (21), and the primes on the summands indicate that the first term should be halved. Here,

the integer K satisfies:

$$K = \max\left\{3, \left\lceil 5\gamma e^{W(\log(140/\epsilon)/(5\gamma))} \right\rceil\right\} = \mathcal{O}\left(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\right), \qquad \epsilon \to 0,$$

where $W(x)$ is the Lambert W function [21, (4.13.1)].

Proof. A bivariate Chebyshev expansion of e^{-ixy} on $(x,y) \in [-\gamma, \gamma] \times [0, 2\pi]$ is given by [25, Lem. A.3]

$$e^{-ixy} = {\sum_{p=0}^{\infty}}' {\sum_{r=0}^{\infty}}' a_{pr} e^{i\pi x} T_p\left(\tfrac{x}{\gamma}\right) T_r\left(\tfrac{y}{\pi} - 1\right), \quad (x,y) \in [-\gamma, \gamma] \times [0, 2\pi],$$

where T_p is the degree p Chebyshev polynomial and the primes on the summands indicate that the first term should be halved. Moreover, the expansion coefficients are given by [25, Lem. A.3]

$$a_{pr} = \begin{cases} 4i^r J_{(p+r)/2}(-\gamma\pi/2) J_{(r-p)/2}(-\gamma\pi/2), & \mod(|p - r|, 2) = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{18}$$

Pick $K \geq 1$ to be an integer. Then, by the triangle inequality, $|T_p(x)| \leq 1$ for $x \in [-1, 1]$, and $|e^{i\theta}| = 1$, we have

$$\sup_{(x,y) \in [-\gamma, \gamma] \times [0, 2\pi]} \left| e^{-ixy} - {\sum_{p=0}^{K-1}}' {\sum_{r=0}^{K-1}}' a_{pr} e^{i\pi x} T_p\left(\tfrac{x}{\gamma}\right) T_r\left(\tfrac{y}{\pi} - 1\right) \right| \leq \sum_{p=K}^{\infty} \sum_{r=K}^{\infty} |a_{pr}|.$$

Using [21, (10.14.1) and (10.14.7)], we find that

$$|a_{pr}| \leq 4\left(\frac{e\gamma\pi}{p+r}\right)^{(p+r)/2}, \qquad \max(p, r) \geq 1.$$

Therefore, by setting $s = p + r$, we can bound the error as

$$\sum_{p=K}^{\infty} \sum_{r=K}^{\infty} |a_{pr}| \leq 4\sum_{s=0}^{\infty}(s+1)\left(\frac{e\gamma\pi}{s+2K}\right)^{(s+2K)/2} \leq 4(e\gamma\pi)\sum_{s=0}^{\infty}\left(\frac{e\gamma\pi}{2K}\right)^{(s+2K-2)/2}.$$

Assuming $K \geq 3$, we find that $\sum_{s=0}^{\infty} \rho^{s/2} = 1/(1-\rho) \leq 7$ with $\rho = (e\gamma\pi)/(2K)$. Hence, we have

$$\sum_{p=K}^{\infty} \sum_{r=K}^{\infty} |a_{pr}| \leq 28(e\gamma\pi)\left(\frac{e\gamma\pi}{2K}\right)^{K-1} \leq 140\left(\frac{5\gamma}{K-1}\right)^{K-1},$$

where the last inequality used $e\gamma\pi \leq 5$, $e\pi/2 \leq 5$, and $K \geq K - 1$. By solving for $K \geq 3$ such that $\sum_{p=K}^{\infty} \sum_{r=K}^{\infty} |a_{pr}| \leq \epsilon$, we find that we can take K to be

$$K = \max\left\{3, \left\lceil 5\gamma e^{W(\log(140/\epsilon)/(5\gamma))} \right\rceil\right\} = \mathcal{O}\left(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\right), \qquad \epsilon \to 0,$$

where $W(x)$ is the Lambert W function. The asymptotic approximation for the lower bound on K as $\epsilon \to 0$ is derived from the asymptotic expansion for $W(x)$ as $x \to \infty$ [21, (4.13.10)]. $\square$

We now evaluate the truncated Chebyshev expansion constructed in Lemma 1 to derive a rank K approximation to the matrix A in (7). We make the additional restriction that $\gamma \leq 1/2$ in the statement of the theorem below because we do not construct low rank approximations to A when $\gamma > 1/2$ (see Section 2.2).

THEOREM 2. *Let $N \geq 1$ be an integer, $0 < \epsilon < 1$, and $x_0, \ldots, x_{N-1}$ samples such that (5) holds with $0 < \gamma \leq 1/2$. Consider the $N \times N$ matrix*

$$A_{jk} = e^{-2\pi i (x_j - j/N)\omega_k}, \qquad 0 \leq j, k \leq N - 1,$$

where $\omega_0, \ldots, \omega_{N-1} \in [0, N]$. Then, there exists a rank K matrix A_K such that $\|A - A_K\|_{\max} \leq \epsilon$, where

$$K = \max\left\{3, \left\lceil 5\gamma e^{W(\log(140/\epsilon)/(5\gamma))} \right\rceil\right\} = \mathcal{O}\left(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\right), \qquad \epsilon \to 0$$

and $\|A\|_{\max}$ is the absolute maximum entry of A.

Proof. Let $\underline{x} = (x_0, x_1, \ldots, x_{N-1})^{\mathsf{T}}$, $\underline{e} = (0/N, 1/N, \ldots, (N-1)/N)^{\mathsf{T}}$ and $\underline{\omega} = (\omega_0, \omega_1, \ldots, \omega_{N-1})^{\mathsf{T}}$. Then, $A = \exp(-2\pi i (\underline{x} - \underline{e})\underline{\omega}^{\mathsf{T}})$, where the exponential function is applied entry-by-entry to its matrix input. Since the entries in $N\underline{x}$ are in $[-\gamma, \gamma]$ and the entries of $2\pi\underline{\omega}/N$ are in $[0, 2\pi]$, we can apply Lemma 1 to each entry of A. We conclude that for $K = \max\{3, \lceil 5\gamma e^{W(\log(140/\epsilon)/(5\gamma))} \rceil\}$ we have

$$\|A - A_K\|_{\max} \leq \epsilon, \quad A_K = \sum_{p=0}^{K-1}{}' \sum_{r=0}^{K-1}{}' a_{pr} \left(\exp\left(-i\pi N(\underline{x} - \underline{e})\right) \circ T_p\left(\tfrac{N(\underline{x}-\underline{e})}{\gamma}\right)\right) T_r\left(\tfrac{2\underline{\omega}^{\mathsf{T}}}{N} - \mathbf{1}^{\mathsf{T}}\right),$$

(19)

where $T_p(x)$ is the degree p Chebyshev polynomial, the coefficients a_{pr} are given in (21), $\mathbf{1}$ is the $N \times 1$ column vector of ones, and the prime on the summands indicate that the first term is halved.

Each term in the double sum in (22) is a rank-1 term so it may look like A_K is of rank at most K^2; however, by appropriately grouping the terms as follows:

$$A_K = \sum_{r=0}^{K-1}{}' \underbrace{\left(\sum_{p=0}^{K-1}{}' a_{pr} \left(\exp\left(-i\pi N(\underline{x} - \underline{e})\right) \circ T_p\left(\tfrac{N(\underline{x}-\underline{e})}{\gamma}\right)\right)\right) T_r\left(\tfrac{2\underline{\omega}^{\mathsf{T}}}{N} - \mathbf{1}^{\mathsf{T}}\right)}_{\text{A rank-1 matrix}},$$

we conclude that A_K is a matrix of rank at most K, as required. The asymptotic order of K given in the statement of the theorem comes from the asymptotic expansion of $W(x)$ as $x \to \infty$ [21, (4.13.10)]. $\square$