Numerical Stability of Fast Trigonometric Transforms – A Worst Case Study

D. Potts
Institute of Mathematics
Medical University of Luebeck
D – 23560 Luebeck, Germany
e–mail: potts@math.mu-luebeck.de

G. Steidl
Faculty of Mathematics and Informatics
University of Mannheim
D – 68131 Mannheim, Germany
e–mail: steidl@math.uni-mannheim.de

M. Tasche*
Department of Mathematics
University of Rostock
D – 18051 Rostock, Germany
e–mail: manfred.tasche@mathematik.uni-rostock.de

Abstract

This paper presents some new results on numerical stability for various fast trigonometric transforms. In a worst case study, we consider the numerical stability of the classical fast Fourier transform (FFT) with respect to different precomputation methods for the involved twiddle factors and show the strong influence of precomputation errors on the numerical stability of the FFT. The examinations are extended to fast algorithms for the computation of discrete cosine and sine transforms and to efficient computations of discrete Fourier transforms for nonequispaced data. Numerical tests confirm the theoretical estimates of numerical stability.

*__Contact author:__ Manfred Tasche, Mailing address: Department of Mathematics,University of Rostock,D – 18051 Rostock, Germany, Tel. 0049 / 381 – 498 – 1549, Fax 0049 / 381 – 498 – 1520, e–mail: manfred.tasche@mathematik.uni-rostock.de

2000 AMS subject classification: 65T50, 65G50, 94A11.

Key words and phrases: Numerical stability, roundoff error, worst case study, fast trigonometric transform, fast Fourier transform, fast cosine transform, fast Fourier transform for nonequispaced data.

1 Introduction

Discrete Fourier transforms and related discrete trigonometric transforms (such as discrete cosine transforms and discrete sine transforms) have found wide applications in numerical analysis and digital signal processing (see [15, 5]). Repeated use of discrete Fourier transforms occurs in fast convolutions and deconvolutions (see [15, pp. 205 – 209]), more general in solving of Toeplitz–plus–Hankel systems [11, 17]. If the transform length is large, then it is important to have fast and numerically stable realizations of discrete Fourier transforms.
In this paper we consider the numerical stability of the fast Fourier transform (FFT). In a worst case study, we show that various precomputation schemes of twiddle factors lead to different behaviour of the numerical stability of the FFT. As always observed in [20], the twiddle factors can be the dominate source of roundoff errors. As conclusion, these twiddle factors should be pretabulated to high accuracy. The methods of repeated subvector scaling and recursive bisection possess both low complexity and good numerical stability. On the other hand, the often used method of forward recursion leads to a bad numerical stability of the FFT.
We use the following concept of numerical stability: Let $\mathbf{x} = (x_k)_{k=0}^{N-1} \in \mathbb{C}^N$ be an arbitrary input vector, $\mathbf{F}_N \in \mathbb{C}^{N,N}$ be the unitary Fourier matrix

$$\mathbf{F}_N := N^{-1/2}(w_N^{jk})_{j,k=0}^{N-1}, \quad w_N := \mathrm{e}^{-2\pi \mathrm{i}/N} \tag{1.1}$$

and $\mathbf{y} := \mathbf{F}_N \mathbf{x}$ be the exact output vector. Let $\hat{\mathbf{y}} \in \mathbb{C}^N$ be the vector computed by floating point arithmetic with unit roundoff u. Then $\hat{\mathbf{y}}$ can be represented in the form

$$\hat{\mathbf{y}} = \mathbf{F}_N(\mathbf{x} + \Delta\mathbf{x}) \quad (\Delta\mathbf{x} \in \mathbb{C}^N).$$

By $\|\mathbf{x}\|_2 := \left(\sum_{k=0}^{N-1} |x_k|^2\right)^{1/2}$ we denote the Euclidean norm of $\mathbf{x}$. An algorithm used for computing $\mathbf{F}_N \mathbf{x}$ is called *normwise backward stable* (see [13, p. 142]), if there exists a positive constant k_N with $k_N u \ll 1$ such that for all vectors $\mathbf{x} \in \mathbb{C}^N$

$$\|\Delta\mathbf{x}\|_2 \le \left(k_N\, u + \mathcal{O}(u^2)\right) \|\mathbf{x}\|_2 . \tag{1.2}$$

Note that the size of k_N, i.e. its asymptotic behaviour with respect to N, is a natural measure of numerical stability for the used algorithm. Since $\mathbf{F}_N$ is unitary, we conclude that

$$\|\Delta\mathbf{x}\|_2 = \|\mathbf{F}_N(\Delta\mathbf{x})\|_2 = \|\hat{\mathbf{y}} - \mathbf{y}\|_2, \quad \|\mathbf{x}\|_2 = \|\mathbf{F}_N\mathbf{x}\|_2 = \|\mathbf{y}\|_2 .$$

Consequently, we have also *normwise forward stability* by

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_2 \leq \big(k_N\, u + \mathcal{O}(u^2)\big)\|\mathbf{y}\|_2\,.$$

An important example of a fast *and* numerically stable algorithm is the FFT. For the Cooley–Tukey and Gentleman–Sande algorithms with $N = 2^n$ (see [15, pp. 17 – 22, 64 – 75]), G. U. Ramos [18], C. Y. Chu [7], M. Arioli et al. [1] and P. Y. Yalamov [28] have shown that $k_N = \mathcal{O}(\log N)$ under the assumption that all twiddle factors are exactly known or precomputed by direct call. See also [12, pp. 9 – 14]. These results can be proved by a clever factorization of the Fourier matrix into a product of sparse unitary matrices which goes back to C. F. Gauss (see [15, pp. 20 – 21]). Using this factorization technique, M. Tasche and H. Zeuner have recently presented both worst case and average case analysis of roundoff errors for various FFTs (see [24, 25]) and for different precomputation methods of twiddle factors (see [26]).
This paper is organized as follows: In Section 2, we consider seven different methods for the precomputation of complex exponentials and present corresponding error estimates in binary floating point arithmetic. We are interested in methods with low complexity and small roundoff error. A new unified approach to error estimates with small constants is derived by means of difference equations. In Section 3, we consider the numerical stability for the direct computation of discrete Fourier transforms and for the FFT (Cooley–Tukey and Gentleman–Sande algorithm) with precomputed twiddle factors. We examine the dependence of the numerical stability of the FFT on precomputation errors. The stability estimates are mainly based on the factorization of the Fourier matrix into a product of unitary sparse matrices. Numerical tests illustrate the theoretical results. In Section 4, we consider fast algorithms for discrete cosine transforms (DCT) and discrete sine transforms (DST). We obtain remarkably stable realizations of DCT and DST by using FFT. Finally in Section 5, we describe a fast and robust algorithm for the computation of discrete Fourier transforms for nonequispaced data.

2 Precomputation of roots of unity

In the following, we use the standard model of binary floating point arithmetic in $\mathbb{R}$ (see [13], p. 44). If $\xi \in \mathbb{R}$ is represented by the floating point number $\mathrm{fl}\,(\xi)$, then

$$\mathrm{fl}\,(\xi) = \xi\,(1+\varepsilon) \quad (|\varepsilon| \leq u)\,,$$

where u denotes the *unit roundoff* (or *machine precision*). For arbitrary $\xi, \eta \in \mathbb{R}$ and any operation $\circ \in \{+, -, \times, /\}$ the exact value $\xi \circ \eta$ and the computed value $\mathrm{fl}\,(\xi \circ \eta)$ are related by

$$\mathrm{fl}\,(\xi \circ \eta) = (\xi \circ \eta)(1+\varepsilon^\circ) \quad (|\varepsilon^\circ| \leq u)\,.$$

In the case of single precision (24 bits for the mantissa (with 1 sign bit), 8 bits for the exponent), we have $u = 2^{-24} \approx 5.96 \times 10^{-8}$ and for double precision (53 bits for

the mantissa (with 1 sign bit), 11 bits for the exponent) $u = 2^{-53} \approx 1.11 \times 10^{-16}$. Since complex arithmetic is implemented using real operations, the complex floating point error model is a consequence of the corresponding real arithmetic model (see [13, pp. 78 – 80]). For arbitrary $\xi, \eta \in \mathbb{C}$, we have

$$\begin{aligned} \mathrm{fl}\,(\xi + \eta) &= (\xi + \eta)(1 + \varepsilon^+) \quad (|\varepsilon^+| \le u), \\ \mathrm{fl}\,(\xi\eta) &= \xi\,\eta\,(1 + \varepsilon^\times) \quad \Big(|\varepsilon^\times| \le \frac{2\sqrt{2}\,u}{1 - 2u} = 2\sqrt{2}\,u + \mathcal{O}(u^2)\Big)\,. \end{aligned} \tag{2.1}$$

For simplicity, we omit the $\mathcal{O}(u^2)$–term and use $2\sqrt{2}\,u$ as upper bound of $|\varepsilon^\times|$. Note that C. Y. Chu [7] has obtained a better upper bound of $|\varepsilon^\times|$ in (2.1), namely $(1 + \sqrt{2})\,u$. But the best possible upper bound of $|\varepsilon^\times|$ is $4\sqrt{3}\,u/3$, see [26]. In particular, if $\xi \in \mathbb{R} \cup \mathrm{i}\mathbb{R}$ and $\eta \in \mathbb{C}$, then

$$\mathrm{fl}\,(\xi\eta) = \xi\,\eta\,(1 + \varepsilon^\times) \quad (|\varepsilon^\times| \le u). \tag{2.2}$$

Now we compare different methods for precomputing w_N^k $(k = 1, \dots, N-1)$ with $N := 2^n$ and $N_j := 2^{-j}N$ $(j = 0, \dots, n)$ (see [7], [15, pp. 23 – 28]). Using symmetries of the complex exponentials, we only need to precompute $N_3 - 1$ values w_N^k $(k = 1, \dots, N_3 - 1)$. The other complex exponentials w_N^l $(l = N_3, \dots, N-1)$ can be obtained by

$$\hat{w}_N^{N_2-k} := -\mathrm{i}\,\overline{\hat{w}_N^k}\,, \quad \hat{w}_N^{N_2+k} := -\mathrm{i}\,\hat{w}_N^k\,, \quad \hat{w}_N^{N_1-k} := -\,\overline{\hat{w}_N^k}\,, \quad \hat{w}_N^{N_1+k} := -\,\hat{w}_N^k$$

for $k = 1, \dots, N_3 - 1$ and by

$$\hat{w}_N^{N_3} := \hat{w}_8 = \mathrm{fl}\left(\sqrt{2}/2\right)(1 - \mathrm{i})\,, \quad \hat{w}_N^{N_2} := w_4 = -\mathrm{i}\,, \quad \hat{w}_N^{N_1} := w_2 = -1\,.$$

The most obvious method is to call repeatedly library routines for cosine and sine functions:

Algorithm 2.1 (Direct call)
Input: $N := 2^n$ $(n \ge 4)$, $\varphi := 2\pi/N$.
For $k = 1\,(1)\,N_3 - 1$ *form*

$$\hat{w}_N^k := \mathrm{fl}\,(\cos(k\varphi)) - \mathrm{i}\,\mathrm{fl}\,(\sin(k\varphi))\,.$$

Output: $\hat{w}_N^k$ *precomputed value of* w_N^k $(k = 1, \dots, N_3 - 1)$.

This algorithm involves almost N_2 trigonometric function calls. If the cosine and sine functions are quality library routines, then very accurate roots of unity are precomputed such that with some $c > 0$

$$|\hat{w}_N^k - w_N^k| \le c\,u \quad (k = 1, \dots, N_3 - 1). \tag{2.3}$$

Let us assume that cosine and sine are internally computed with higher precision and then rounded towards the next binary floating point number. Within $[-1, 1]$ these floating point numbers are spaced with a distance $\leq u$. Therefore we have

$$|\,\mathrm{fl}\,(\cos(k\varphi)) - \cos(k\varphi)| \leq u/2\,, \quad |\,\mathrm{fl}\,(\sin(k\varphi)) - \sin(k\varphi)| \leq u/2$$

such that $c = \sqrt{2}/2$ in (2.3). Since direct calls are time consuming, we consider other methods for on–line computation of w_N^k.
The next algorithm uses only two trigonometric function calls and is based on repeated multiplication $w_N^k = w_N\, w_N^{k-1}$:

Algorithm 2.2 (Repeated multiplication)
Input: $N := 2^n$ $(n \geq 4)$, $\varphi := 2\pi/N$.
1. *Form by direct call*

$$\hat{w}_N := \mathrm{fl}\,(\cos\,\varphi) - \mathrm{i}\,\mathrm{fl}\,(\sin\,\varphi)\,.$$

2. *For* $k = 2\,(1)\,N_3 - 1$ *form*

$$\hat{w}_N^k := \mathrm{fl}\left(\hat{w}_N\,\hat{w}_N^{k-1}\right).$$

Output: $\hat{w}_N^k$ *precomputed value of* w_N^k $(k = 1, \ldots, N_3 - 1)$.

Since one complex multiplication requires 4 real multiplications and 2 real additions (i.e. 6 flops), a total of $6\,(N_3 - 1) < 3N_2$ flops and 2 function calls are involved. Using Algorithm 2.2, it follows from (2.1) that

$$\hat{w}_N^k = \hat{w}_N\,\hat{w}_N^{k-1}\,(1 + \varepsilon_k^{\times}) = w_N\,\hat{w}_N^{k-1} + \mu_k \quad (k \geq 2)\,, \tag{2.4}$$

where

$$\mu_k := \Omega_1\,\hat{w}_N^{k-1} + \varepsilon_k^{\times}\hat{w}_N\,\hat{w}_N^{k-1} = \Omega_1\,w_N^{k-1} + \varepsilon_k^{\times}\,w_N^k + \mathcal{O}(u^2)\,,$$

$|\varepsilon_k^{\times}| \leq 2\sqrt{2}u$, and $\Omega_1 := \hat{w}_N - w_N$. As usual, we omit the $\mathcal{O}(u^2)$–term such that $|\mu_k| \leq 5\sqrt{2}\,u/2$. Setting $\Omega_k := \hat{w}_N^k - w_N^k$, we obtain by (2.4) that

$$\Omega_k = w_N\,\Omega_{k-1} + \mu_k \quad (k \geq 1)$$

with $\Omega_0 := 0$, $\mu_1 := \Omega_1$. Then the solution of this two–term recursion reads as follows

$$\Omega_k = \sum_{j=1}^{k} \mu_j\,w_N^{k-j}$$

such that

$$|\hat{w}_N^k - w_N^k| = |\Omega_k| \leq \sum_{j=1}^{k} |\mu_j| \leq c\,u\,k \quad (k = 1, \ldots, N_3 - 1) \tag{2.5}$$

with $c := 5\sqrt{2}/2$.
The third method combines the accuracy of Algorithm 2.1 with the arithmetical simplicity of Algorithm 2.2 by using the fact that

$$\left(w_N^{2^j+k}\right)_{k=1}^{2^j-1} = w_N^{2^j} \left(w_N^k\right)_{k=1}^{2^j-1} \quad (j = 1, \dots, n-4).$$

Algorithm 2.3 (Repeated subvector scaling)
Input: $N := 2^n$ $(n \geq 5)$.
1. *For* $j = 0\,(1)\,n-4$ *form*

$$\hat{w}_{N_j} = \hat{w}_N^{2^j} := \mathrm{fl}\,(\cos\,\varphi_j) - \mathrm{i}\,\mathrm{fl}\,(\sin\,\varphi_j)$$

with $\varphi_j := 2\pi/N_j$ *by direct call.*
2. *For* $j = 1\,(1)\,n-4$ *multiply*

$$\left(\hat{w}_N^{2^j+k}\right)_{k=1}^{2^j-1} := \mathrm{fl}\left(\hat{w}_{N_j} \left(\hat{w}_N^k\right)_{k=1}^{2^j-1}\right).$$

Output: $\hat{w}_N^k$ *precomputed value of* w_N^k $(k = 1, \dots, N_3 - 1)$.

This algorithm requires about $3N_2$ flops and $2n - 3 < 2\log_2 N$ function calls. By the same method as before we can estimate the roundoff error

$$|\hat{w}_N^k - w_N^k| \leq \begin{cases} \sqrt{2}\,u/2 & k = 2^j\ (j = 0, \dots, n-4), \\ 5\sqrt{2}\,(j+1)\,u/2 & 2^j < k < 2^{j+1}\ (j = 1, \dots, n-4). \end{cases}$$

Thus we obtain

$$|\hat{w}_N^k - w_N^k| \leq 5\sqrt{2}\,(n-3)\,u/2 \quad (k = 1, \dots, N_3 - 1). \tag{2.6}$$

This is a fast and stable method and therefore very convenient for practical implementations of FFT.
The next algorithm uses the fact that w_N^k satisfies the three–term recursion

$$w_N^k = \tau\, w_N^{k-1} - w_N^{k-2} \quad (k \geq 2) \tag{2.7}$$

with the real multiplicator $\tau = 2\cos\varphi$ $(\varphi := 2\pi/N)$. In other words, both $a_k = \sin(k\varphi)$ and $a_k = \cos(k\varphi)$ fulfill

$$a_k = \tau\, a_{k-1} - a_{k-2} \quad (k \geq 2).$$

Algorithm 2.4 (Forward recursion)
Input: $N := 2^n$ $(n \geq 4)$, $\varphi := 2\pi/N$.
1. *Set* $\hat{w}_N^0 := 1$ *and evaluate by direct call*

$$\hat{\tau} := 2\,\mathrm{fl}\,(\cos\varphi), \quad \hat{w}_N^1 = \mathrm{fl}\,(\cos\varphi) - \mathrm{i}\,\mathrm{fl}\,(\sin\varphi).$$

2. *For* $k = 2\,(1)\,N_3 - 1$ *form*

$$\hat{w}_N^k = \mathrm{fl}\left(\hat{\tau}\,\hat{w}_N^{k-1} - \hat{w}_N^{k-2}\right). \tag{2.8}$$

Output: $\hat{w}_N^k$ *precomputed value of* w_N^k $(k = 1, \ldots, N_3 - 1)$.

The forward recursion uses only $4\,(N_3 - 2) < N_1$ flops and 2 direct calls. Now we estimate the roundoff error of $\hat{w}_N^k$ computed with Algorithm 2.4. Since the binary floating point numbers of the interval [1, 2] are equispaced with the distance $2u$, we can assume that $\hat{\tau} = \tau + \delta^\tau \in \mathbb{R}$ $(|\delta^\tau| \le u)$. Using (2.1) – (2.2), it follows from (2.8) that

$$\begin{aligned} \hat{w}_N^k &= \left((\tau + \delta^\tau)\,\hat{w}_N^{k-1}(1 + \varepsilon_k^\times) - \hat{w}_N^{k-2}\right)(1 + \varepsilon_k^+) \\ &= \tau\,\hat{w}_N^{k-1} - \hat{w}_N^{k-2} + \mu_k \end{aligned} \tag{2.9}$$

with $|\varepsilon_k^\times|$, $|\varepsilon_k^+| \le u$ and

$$\mu_k = \tau\,w_N^{k-1}\,\varepsilon_k^\times + \delta^\tau\,w_N^{k-1} - w_N^{k-2}\,\varepsilon_k^+ + \mathcal{O}(u^2).$$

As usual, we omit the $\mathcal{O}(u^2)$–term in μ_k such that

$$|\mu_k| \le (\tau + 2)\,u \le 4u \quad (k \ge 2).$$

Setting $\Omega_k := \hat{w}_N^k - w_N^k$, we obtain by (2.7) and (2.9)

$$\Omega_k = \tau\,\Omega_{k-1} - \Omega_{k-2} + \mu_k \quad (k \ge 2) \tag{2.10}$$

with $\Omega_0 := 0$ and $\Omega_1 := \hat{w}_N^1 - w_N$.

Lemma 2.5 *The inhomogeneous three–term recursion* (2.10) *has the solution*

$$\Omega_k = \sum_{j=1}^{k} \mu_j\,\frac{\sin(k - j + 1)\varphi}{\sin\varphi} \tag{2.11}$$

with $\mu_1 := \hat{w}_N^1 - w_N$.

The proof is a straightforward calculation and therefore omitted. Note that

$$g(j,k) := \frac{\sin(|k - j| + 1)\varphi}{\sin\varphi}$$

is the *discrete Green function* of (2.10) (see [8, pp. 164 – 167]). By (2.11) and

$$\sum_{j=1}^{k-1} \frac{\sin(j\varphi)}{\sin\varphi} = \frac{1}{2}\left(\frac{1 - \cos(k\varphi)}{1 - \cos\varphi} - \frac{\sin(k\varphi)}{\sin\varphi}\right) \quad (k \ge 2) \tag{2.12}$$

we can estimate for $k = 1, \ldots, N_3 - 1$

$$\begin{aligned} |\hat{w}_N^k - w_N^k| &= |\Omega_k| \le \sum_{j=1}^{k} |\mu_j| \, \frac{\sin(k-j+1)\varphi}{\sin\varphi} \\ &\le 2u \, \frac{\sin(k\varphi)}{\sin\varphi} + 4u \sum_{j=1}^{k-1} \frac{\sin(j\varphi)}{\sin\varphi} = 2u \, \frac{1-\cos(k\varphi)}{1-\cos\varphi} \,. \end{aligned}$$

Hence we obtain the estimate

$$|\hat{w}_N^k - w_N^k| \le 2\,k^2\,u \quad (k = 1, \ldots, N_3 - 1) \,.$$

This popular method is "unstable" and hence unsuitable for practical implementation of FFT.
In order to improve the "stability" of Algorithm 2.4 , C. Reinsch (see [8, p. 173]) has proposed the following procedure. By (2.7) we have

$$w_N^{k-1} - w_N^k = \sigma\, w_N^{k-1} + w_N^{k-2} - w_N^{k-1}$$

with $\sigma := 2 - \tau = (2\sin\varphi/2)^2$. Setting $d_k := w_N^{k-1} - w_N^k$, we obtain a system of two–term recursions

$$\begin{aligned} d_k &= \sigma\, w_N^{k-1} + d_{k-1} \,, \qquad (k \ge 2) \,, \\ w_N^k &= w_N^{k-1} - d_k \end{aligned}$$

where $d_1 := 1 - w_N$, $w_N^1 := w_N$. This leads to:

Algorithm 2.6 (Stabilized forward recursion)
Input: $N = 2^n$ $(n \ge 5)$, $\varphi := 2\pi/N$.
1. *Evaluate by direct call*

$$\hat{\sigma} := \mathrm{fl}\big((\,\mathrm{fl}\,(2\sin\varphi/2))^2\big), \quad \hat{w}_N^1 := \mathrm{fl}\,(1 - \hat{\sigma}/2) - \mathrm{i}\,\mathrm{fl}\,(\sin\varphi) \,, \quad \hat{d}_1 = \hat{\sigma}/2 + \mathrm{i}\,\mathrm{fl}\,(\sin\varphi) \,.$$

2. *For* $k = 2\,(1)\,N_3 - 1$ *form*

$$\hat{d}_k := \mathrm{fl}\left(\hat{\sigma}\,\hat{w}_N^{k-1} + \hat{d}_{k-1}\right), \quad \hat{w}_N^k := \mathrm{fl}\left(\hat{w}_N^{k-1} - \hat{d}_k\right). \tag{2.13}$$

Output: $\hat{w}_N^k$ *precomputed value of* w_N^k $(k = 1, \ldots, N_3 - 1)$.

Algorithm 2.6 requires $6(N_3 - 2) + 2 < 3N_2$ flops and 2 direct calls. Now we estimate the roundoff error of $\hat{w}_N^k$ computed by Algorithm 2.6. The precomputation of $\hat{\sigma}$ yields $\hat{\sigma} = \sigma + \delta^\sigma \in \mathbb{R}$ with $|\delta^\sigma| \le 3\,\varphi^2\,u$. Using (2.1) – (2.2), from (2.13) it follows that

$$\begin{aligned} \hat{d}_k &= \big((\sigma + \delta^\sigma)\,\hat{w}_N^{k-1}\,(1 + \varepsilon_k^\times) + \hat{d}_{k-1}\big)(1 + \varepsilon_k^+) \\ &= \sigma\,\hat{w}_N^{k-1} + \hat{d}_{k-1} + \mu_k \end{aligned} \tag{2.14}$$

with $|\varepsilon_k^\times|$, $|\varepsilon_k^+| \le u$ and $\mu_k = \sigma\, w_N^{k-1}\, \varepsilon_k^\times + d_k\, \varepsilon_k^+ + w_N^{k-1}\, \delta^\sigma + \mathcal{O}(u^2)$ and further

$$\hat{w}_N^k = (\hat{w}_N^{k-1} - \hat{d}_k)(1 + \varepsilon_k^-) = \hat{w}_N^{k-1} - \hat{d}_k + \nu_k \tag{2.15}$$

with $|\varepsilon_k^-| \le u$ and $\nu_k = w_N^k\, \varepsilon_k^- + \mathcal{O}(u^2)$. Again we omit the $\mathcal{O}(u^2)$–terms in μ_k and ν_k such that $|\mu_k| \le (\varphi + 4\,\varphi^2)\, u$ and $|\nu_k| \le u$. Setting $\Omega_k := \hat{w}_N^k - w_N^k$ and $\Delta_k := \hat{d}_k - d_k$, we obtain by (2.13) – (2.15)

$$\begin{array}{rcll} \Delta_k &=& \sigma\, \Omega_{k-1} + \Delta_{k-1} + \mu_k\,, & \\ \Omega_k &=& \Omega_{k-1} - \Delta_k + \nu_k & (k = 2, \ldots, N_3 - 1)\,, \end{array} \tag{2.16}$$

where $\Delta_1 := \hat{d}_1 - d_1$ and $\Omega_1 := \hat{w}_N - w_N$ such that $|\Delta_1| \le \frac{3}{2}\,\varphi^2\, u + \varphi\, u$ and $|\Omega_1| \le \frac{1}{2}\sqrt{2}\, u$. With $\sigma = 2 - \tau$, we conclude from (2.16) that

$$\Omega_k = \tau\, \Omega_{k-1} - \Omega_{k-2} + \nu_k - \nu_{k-1} - \mu_k \quad (k = 2, \ldots, N_3 - 1)$$

with $\Omega_0 := 0$, $\Omega_1 := \hat{w}_N - w_N$ and $\nu_1 := \Omega_1 + \Delta_1$. By Lemma 2.5, this inhomogeneous three–term recursion has the solution

$$\begin{aligned} \Omega_k &= (\hat{w}_N - w_N)\, \frac{\sin(k\varphi)}{\sin\varphi} + \sum_{j=2}^{k} (\nu_j - \nu_{j-1} - \mu_j)\, \frac{\sin(k-j+1)\varphi}{\sin\varphi} \\ &= (\hat{w}_N - w_N - \nu_1)\, \frac{\sin(k\varphi)}{\sin\varphi} + \nu_k + \sum_{j=2}^{k-1} \nu_j\, \frac{\sin(k-j+1)\varphi - \sin(k-j)\varphi}{\sin\varphi} \\ &\quad - \sum_{j=2}^{k} \mu_j\, \frac{\sin(k-j+1)\varphi}{\sin\varphi}\,. \end{aligned}$$

Using (2.12) and $\hat{w}_N - w_N - \nu_1 = -\Delta_1$, we obtain for $k = 2, \ldots, N_3 - 1$

$$\begin{aligned} |\Omega_k| &\le \frac{\sin(k\varphi)}{\sin\varphi}\, \big(\frac{3}{2}\varphi^2 + \varphi\big)\, u + u + \frac{\sin(k-1)\varphi - \sin\varphi}{\sin\varphi}\, u \\ &\quad + \frac{1}{2}\Big(\frac{1 - \cos(k\varphi)}{1 - \cos\varphi} - \frac{\sin(k\varphi)}{\sin\varphi}\Big)\, (\varphi + 4\varphi^2)\, u \\ &\le \big(\frac{1}{2}\,(\varphi - \varphi^2)\, k + k - 1\big) u + (\frac{1}{2} + 2\varphi)\, \varphi\, k^2 u\,. \end{aligned}$$

Since $k\varphi \le \pi/4$, we can estimate

$$|\hat{w}_N^k - w_N^k| \le \big(1 + \frac{\pi}{8}\big)(k+1)\, u \quad (k = 1, \ldots, N_3 - 1)\,.$$

Using the trigonometric identities

$$\begin{aligned} \cos(2^s + m)\varphi &= 2\cos(2^s\varphi)\cos(m\varphi) \;-\; \cos(2^s - m)\varphi\,, \\ \sin(2^s + m)\varphi &= 2\cos(2^s\varphi)\sin(m\varphi) \;+\; \sin(2^s - m)\varphi \end{aligned}$$

we obtain the recursion

$$w_N^{2^s+m} \;=\; \tau_{2^s}\, w_N^m \;-\; \overline{w_N^{2^s-m}} \tag{2.17}$$

with $\tau_{2^s} := 2\,\mathrm{Re}\, w_N^{2^s}$. If $w_N^{2^s}$ $(s = 0, \dots, n-4)$ are precomputed by direct call, then the rest of roots of unity can be derived recursively by (2.17).

Algorithm 2.7 (Logarithmic recursion)
Input: $N = 2^n$ $(n \geq 5)$, $\varphi := 2\pi/N$.
1. *For* $s = 0\,(1)\,n-4$ *form by direct call*

$$\hat{w}_N^{2^s} := \mathrm{fl}\,(\cos(2^s\varphi)) - \mathrm{i}\;\mathrm{fl}\,(\sin(2^s\varphi))\,.$$

2. *For* $s = 1\,(1)\,n-4$ *compute*

$$j := 2^s\,, \quad \hat{\tau}_j := 2\;\mathrm{fl}\left(\mathrm{Re}\,\hat{w}_N^j\right)$$

and for $m = 1\,(1)\,j-1$ *form*

$$\hat{w}_N^{j+m} := \mathrm{fl}\left(\hat{\tau}_j \hat{w}_N^m \;-\; \overline{\hat{w}_N^{j-m}}\right). \tag{2.18}$$

Output: $\hat{w}_N^k$ *precomputed value of* w_N^k $(k = 1, \dots, N_3 - 1)$.

This algorithm requires almost N_1 flops and $2(n-3) < 2\log_2 N$ direct calls. Now we sketch an estimate of the roundoff error. Assume that $\hat{\tau}_j = \tau_j + \delta_j^\tau$ $(|\delta_j^\tau| \leq u)$. Using (2.1) – (2.2), from (2.18) it follows that

$$\hat{w}_N^{j+m} := \tau_j \hat{w}_N^m \;-\; \overline{\hat{w}_N^{j-m}} \;+\; \mu_{j,m} \tag{2.19}$$

with

$$\mu_{j,m} \;=\; \tau_j\, w_N^m\, \varepsilon_{j,m}^{\times} \;+\; \delta_j^\tau\, w_N^m \;+\; w_N^{j+m}\, \varepsilon_{j,m}^{+} \;+\mathcal{O}(u^2) \quad (|\varepsilon_{j,m}^{\times}|,\, |\varepsilon_{j,m}^{+}| \leq u)\,.$$

Omitting the $\mathcal{O}(u^2)$–term, we see that $|\mu_{j,m}| \leq 4u$. Then we obtain by (2.17) and (2.19) that

$$\Omega_{j+m} \;=\; \tau_j\, \Omega_m \;-\; \overline{\Omega}_{j-m} \;+\; \mu_{j,m}$$

and hence

$$|\Omega_{j+m}| \leq 2\,|\Omega_m| \;+\; |\Omega_{j-m}| \;+ 4u\,.$$

By induction one can show that

$$|\hat{w}_N^k - w_N^k| \;=\; |\Omega_k| \leq \left(\left(\frac{\sqrt{2}}{2} + 2\right)k - 2\right) u \quad (k = 1, \dots, N_3 - 1).$$

The last method of recursive bisection (see [6]) is based on recursive application of the trigonometric identities

$$\begin{aligned}\cos\alpha &= \frac{1}{2\cos\beta}\big(\cos(\alpha-\beta)+\cos(\alpha+\beta)\big),\\ \sin\alpha &= \frac{1}{2\cos\beta}\big(\sin(\alpha-\beta)+\sin(\alpha+\beta)\big).\end{aligned}$$

Assume that $w_N^{2^s} = w_{N_s}$ $(s = 0,\dots,n-3)$ are precomputed by direct call. Then the other roots of unity w_N^j $(j = 1,\dots,N_3-1)$ can be determined stepwise by

$$w_N^j = h_p\,(w_N^{j-p} + w_N^{j+p}), \tag{2.20}$$

where $h_p := (2\,\mathrm{Re}\,w_N^p)^{-1} = (2\cos(p\varphi))^{-1}$ and p is the largest power of 2 dividing j.

Algorithm 2.8 (Recursive bisection)
Input: $N = 2^n$ $(n \ge 5)$, $\varphi := 2\pi/N$.
1. *For* $s = 0\,(1)\,n-3$ *form by direct call*

$$\hat{w}_N^{2^s} := \mathrm{fl}\,(\cos(2^s\varphi)) - \mathrm{i}\,\mathrm{fl}\,(\sin(2^s\varphi)).$$

2. *For* $s = 1\,(1)\,n-4$ *compute*

$$p := 2^{n-s-4}, \quad \hat{h}_p := \mathrm{fl}\Big(\mathrm{fl}\big((2\,\mathrm{Re}\,\hat{w}_N^p)\big)^{-1}\Big)$$

and for $j = 3p\,(2p)\,N_3 - p$ *form*

$$\hat{w}_N^j := \mathrm{fl}\Big(\hat{h}_p(\hat{w}_N^{j-p} + \hat{w}_N^{j+p})\Big).$$

Output: $\hat{w}_N^k$ *precomputed value of* w_N^k $(k = 1,\dots,N_3-1)$.

This algorithm requires $4\,(N_3 - n) < N_1$ flops and $2(n-2) < 2\log_2 N$ direct calls. Now we estimate the roundoff error of $\hat{w}_N^k$ computed by Algorithm 2.8. The computation of $\hat{h}_p$ yields $\hat{h}_p = h_p\,(1+\varepsilon_p^h) \in \mathbb{R}$ with $|\varepsilon_p^h| \le 2\,u$.
Using (2.1) – (2.2), it follows that

$$\begin{aligned}\hat{w}_N^j &= h_p\,(1+\varepsilon_p^h)(\hat{w}_N^{j-p} + \hat{w}_N^{j+p})(1+\varepsilon_j^+)(1+\varepsilon_j^\times)\\ &= h_p\,(\hat{w}_N^{j-p} + \hat{w}_N^{j+p}) + \mu_j\,,\end{aligned} \tag{2.21}$$

where $|\varepsilon_j^+|$, $|\varepsilon_j^\times| \le u$ and $\mu_j = w_N^j\,(\varepsilon_j^+ + \varepsilon_j^\times + \varepsilon_p^h) + \mathcal{O}(u^2)$. We omit the $\mathcal{O}(u^2)$–term in μ_j such that $|\mu_j| \le 4\,u$. By (2.20) and (2.21) we obtain

$$\Omega_j = h_p\,(\Omega_{j-p} + \Omega_{j+p}) + \mu_j\,. \tag{2.22}$$

By direct call we know that

$$|\Omega_{2^s}| \le \sqrt{2}\, u/2 \quad (s = 0, \dots, n-3)\,.$$

Now we show that

$$|\hat{w}_N^j - w_N^j| = |\Omega_j| \le 3\, \lceil \log_2 j \rceil\, u \quad (j = 3, \dots, N_3 - 1)\,. \tag{2.23}$$

Similar to Algorithm 2.6 we estimate $|\Omega_j|$ by (2.22). Without lost of generality we consider only $j \in \{N_4+1, \dots, N_3-1\}$, i.e. $\lceil \log_2 j \rceil = n-3$. It is sufficient to prove by induction on s that

$$|\Omega_j| \le (2 + 3\, s)\, u\,, \tag{2.24}$$

if 2^{n-s-4} is the largest power of dividing j. For $s = 1$, i.e. $p = 2^{n-5}$ and $h_p = (2\cos(\pi/16))^{-1} < \sqrt{2}/2$, we have by (2.22)

$$\Omega_{3p} = h_p\, (\Omega_{2p} + \Omega_{4p}) + \mu_{3p}$$

and hence $|\Omega_{3p}| \le 5\, u$. For $s = 2$, i.e. $p = 2^{n-6}$ and $h_p = (2\cos(\pi/32))^{-1}$, we have by (2.22)

$$\Omega_{5p} = h_p\, (\Omega_{4p} + \Omega_{6p}) + \mu_{5p}$$

and hence $|\Omega_{5p}| \le 8\, u$. The same estimate is true for $|\Omega_{7p}|$.
For $s \in \{3, \dots, n-4\}$ $(n \ge 7)$, i.e. $p = 2^{n-s-4}$ and $h_p = (2\cos(p\varphi))^{-1} = 1/2 + \delta_p$ with $0 < \delta_p < 4^{-s}$, we have by (2.22) with $j = N_4 + p\,(2p)\, N_3 - p$

$$|\Omega_j| \le (1/2 + \delta_p)(|\Omega_{j-p}| + |\Omega_{j+p}|) + 4\, u\,.$$

Note that $|\Omega_{j-p}|$ and $|\Omega_{j+p}|$ have been estimated before in different "levels" $s-1$ and $\le s-2$ such that

$$|\Omega_{j-p}| + |\Omega_{j+p}| \le (6\, s - 5)\, u\,.$$

By simple calculations it follows (2.24). This completes the proof of (2.23) for $j \in \{N_4 + 1, \dots, N_3 - 1\}$.

3 Numerical stability of FFT

Let $\mathbf{x} = (x_k)_{k=0}^{N-1} \in \mathbb{C}^N$ be an arbitrary input vector and $\mathbf{y} := \mathbf{F}_N \mathbf{x}$ be the exact output vector. Further let $\hat{\mathbf{y}}$ be the computed output vector using floating point arithmetic with unit roundoff u.
First we consider the direct computation of $\mathbf{F}_N \mathbf{x}$.

Theorem 3.1 *Let $N = 2^n$. Assume that*

$$|\hat{w}_N^k - w_N^k| \le c_N\, u \quad (k = 1, \dots, N-1)\,.$$

Then the direct computation of $\mathbf{F}_N\mathbf{x}$ *is normwise backward stable with the constant*

$$k_N = \begin{cases} N^{1/2}(N+2+c_N) & \textit{for recursive summation,} \\ N^{1/2}(\log_2 N + 3 + c_N) & \textit{for cascade summation.} \end{cases}$$

Proof. 1. For arbitrary $\mathbf{x}, \mathbf{y} \in \mathbb{C}^N$, we evaluate $\mathbf{x}^{\mathrm{T}}\mathbf{y}$ by recursive and cascade summation, respectively. Then we obtain for the roundoff error (see [13, p. 539 and p. 69]) that

$$|\operatorname{fl}\left(\mathbf{x}^{\mathrm{T}}\mathbf{y}\right) - \mathbf{x}^{\mathrm{T}}\mathbf{y}| \le \begin{cases} \frac{(N+2)\,u}{1-(N+2)\,u}\,|\mathbf{x}|^{\mathrm{T}}\,|\mathbf{y}| & \text{for recursive summation,} \\ \frac{(n+3)\,u}{1-(n+3)\,u}\,|\mathbf{x}|^{\mathrm{T}}\,|\mathbf{y}| & \text{for cascade summation,} \end{cases}$$

where $|\mathbf{x}| := (|x_j|)_{j=0}^{N-1}$.
2. The above result can be extended to the matrix–vector product $\mathbf{F}_N\mathbf{x}$. Setting

$$\hat{\mathbf{F}}_N := N^{-1/2}\,(\hat{w}_N^{jk})_{j,k=0}^{N-1}\,,$$

a direct computation of $\mathbf{F}_N\mathbf{x}$ with precomputed roots of unity yields the componentwise estimate

$$|\operatorname{fl}\left(\hat{\mathbf{F}}_N\mathbf{x}\right) - \mathbf{F}_N\mathbf{x}| \le \begin{cases} \left((N+2+c_N)\,u + \mathcal{O}(u^2)\right)|\mathbf{F}_N|\,|\mathbf{x}| & \text{for recursive summation,} \\ \left((n+3+c_N)\,u + \mathcal{O}(u^2)\right)|\mathbf{F}_N|\,|\mathbf{x}| & \text{for cascade summation} \end{cases}$$

with $|\mathbf{F}_N| := N^{-1/2}\,(1)_{j,k=0}^{N-1}$. Note that $\|\,|\mathbf{x}|\,\|_2 = \|\mathbf{x}\|_2$ and that the *spectral norm* $\|\,|\mathbf{F}_N|\,\|_2 = N^{1/2}$. Taking the Euclidean norm of above inequality, we obtain

$$\|\operatorname{fl}\left(\hat{\mathbf{F}}_N\mathbf{x}\right) - \mathbf{F}_N\mathbf{x}\|_2 \le \begin{cases} \left(N^{1/2}(N+2+c_N)\,u + \mathcal{O}(u^2)\right)\|\mathbf{x}\|_2 & \text{for recursive summation,} \\ \left(N^{1/2}(n+3+c_N)\,u + \mathcal{O}(u^2)\right)\|\mathbf{x}\|_2 & \text{for cascade summation.} \end{cases}$$

This completes the proof. ∎

For direct computation of $\mathbf{F}_N\mathbf{x}$, where the entries of $\mathbf{F}_N$ are precomputed by direct call, the roundoff error is relatively large, namely

$$k_N = \begin{cases} \mathcal{O}(N^{3/2}) & \text{for recursive summation,} \\ \mathcal{O}(N^{1/2}\log_2 N) & \text{for cascade summation.} \end{cases}$$

Now we consider the roundoff error of FFT. We will see that FFT is very stable provided that accurately precomputed twiddle factors are used. Further we will show that FFT is very sensitive for errors of precomputation. The results are mainly based on a factorization of $\mathbf{F}_N$ into a product of sparse unitary matrices. As known

(see [15, pp. 17 – 18]), the Cooley–Tukey algorithm for computing $\mathbf{F}_N\mathbf{x}$ corresponds to the factorization

$$\mathbf{F}_N = 2^{-n/2}\,\mathbf{M}_N^{(n)}\,\mathbf{M}_N^{(n-1)}\ldots\mathbf{M}_N^{(1)}\,\mathbf{B}_N \quad (N=2^n), \tag{3.1}$$

where $\mathbf{B}_N$ is the *bit–reversal permutation matrix* and $\mathbf{M}_N^{(j)}$ is the *Kronecker product*

$$\mathbf{M}_N^{(j)} := \mathbf{I}_{N_j}\otimes\mathbf{A}_{2^j} \quad (j=1,\ldots,n;\ N_j = N/2^j) \tag{3.2}$$

with

$$\mathbf{A}_{2^j} := \begin{pmatrix} \mathbf{I}_{2^{j-1}} & \mathbf{W}_{2^{j-1}} \\ \mathbf{I}_{2^{j-1}} & -\mathbf{W}_{2^{j-1}} \end{pmatrix}, \quad \mathbf{W}_{2^{j-1}} := \operatorname{diag}\left(w_{2^j}^k\right)_{k=0}^{2^{j-1}-1}. \tag{3.3}$$

Clearly, $\mathbf{M}_N^{(j)}$ contains only 2 nonzero entries in each row and column, respectively. The nontrivial entries $\neq \pm 1$ of $\mathbf{M}_N^{(j)}$ are called *twiddle factors*. Furthermore, $2^{-1/2}\,\mathbf{M}_N^{(j)}$ is unitary, since

$$\mathbf{A}_{2^j}\,(\overline{\mathbf{A}}_{2^j})^{\mathrm{T}} = 2\,\mathbf{I}_{2^j}$$

and hence

$$\frac{1}{2}\,\mathbf{M}_N^{(j)}\,(\overline{\mathbf{M}}_N^{(j)})^{\mathrm{T}} = \frac{1}{2}\,(\mathbf{I}_{N_j}\otimes\mathbf{A}_{2^j}\,\overline{\mathbf{A}}_{2^j}^{\mathrm{T}}) = \mathbf{I}_{N_j}\otimes\mathbf{I}_{2^j} = \mathbf{I}_N.$$

Consequently, for $0\le j<k\le n$ we see that the matrix product

$$(2^{-1/2}\,\mathbf{M}_N^{(k)})\ \ldots\ (2^{-1/2}\,\mathbf{M}_N^{(j+1)})$$

is also unitary and has the spectral norm 1, i.e.

$$\|\mathbf{M}_N^{(k)}\ldots\mathbf{M}_N^{(j+1)}\|_2 = 2^{(k-j)/2}. \tag{3.4}$$

We summarize:

Algorithm 3.2 (Cooley–Tukey Algorithm)
Input: $N := 2^n$ $(n\ge 5)$, $\mathbf{x}\in\mathbb{C}^N$.
0. *Precompute* w_N^k $(k=1,\ldots,N_1-1)$ *by a method of Section* 2.
1. *Permute*

$$\mathbf{x}_0 := \mathbf{B}_N\mathbf{x}.$$

2. *For* $j = 1\,(1)\,n$ *form*

$$\mathbf{x}_j := \mathbf{M}_N^{(j)}\mathbf{x}_{j-1}.$$

3. *Multiply*

$$\mathbf{y} := 2^{-n/2}\mathbf{x}_n.$$

Output: $\mathbf{y} = \mathbf{F}_N\mathbf{x}$.

Algorithm 3.2 is an example of a *decimation–in–time* FFT. Since all twiddle factors are precomputed by one of the algorithms in Section 2, we use matrices $\mathbf{M}_N^{(j)}$ with precomputed entries $\hat{w}_N^k$ in step 2. These matrices will be denoted by $\hat{\mathbf{M}}_N^{(j)}$. Now we describe the influence of the precomputed twiddle factors and the floating point arithmetic in the roundoff error of the FFT. Assume that the entries $w_{2^j}^k = w_N^{N_j k}$ $(j = 3, \ldots, n;\ k = 1, \ldots, 2^{j-1}-1)$ of $\mathbf{M}_N^{(j)}$ (see (3.2) – (3.3)) have been precomputed up to an absolute error at most $c_N\, u$. By Section 2, we have

$$|\hat{w}_N^{N_j k} - w_{2^j}^k| \le c_N\, u\,, \tag{3.5}$$

where

$$c_N := \begin{cases} \frac{\sqrt{2}}{2} & \text{for direct call,} \\ \frac{5\sqrt{2}}{16}\, N & \text{for repeated multiplication,} \\ \frac{5\sqrt{2}}{2}\, \log_2 N & \text{for repeated subvector scaling,} \\ \frac{1}{32}\, N^2 & \text{for forward recursion,} \\ \frac{8+\pi}{64}\, N & \text{for stabilized forward recursion,} \\ \frac{\sqrt{2}+4}{16}\, N & \text{for logarithmic recursion,} \\ 3\, \log_2 N & \text{for recursive bisection.} \end{cases} .$$

Theorem 3.3 *Let $N = 2^n$ $(n \ge 5)$. Assume that*

$$|\hat{w}_N^{N_j k} - w_{2^j}^k| \le c_N\, u \quad (j = 3, \ldots, n;\ k = 1, \ldots, 2^{j-1} - 1)\,.$$

Then the Cooley–Tukey Algorithm 3.2 *is normwise backward stable with the constant*

$$k_N = (\sqrt{2}\, c_N + 4 + \sqrt{2}) \log_2 N\,. \tag{3.6}$$

As always observed in [20], the twiddle factors can be the dominate source of roundoff error. Theorem 3.3 implies that

$$k_N = \begin{cases} \mathcal{O}(\log_2 N) & \text{for direct call,} \\ \mathcal{O}((\log_2 N)^2) & \text{for repeated subvector scaling and recursive bisection,} \\ \mathcal{O}(N \log_2 N) & \text{for repeated multiplication, stabilized forward recursion and logarithmic recursion,} \\ \mathcal{O}(N^2 \log_2 N) & \text{for forward recursion.} \end{cases}$$

Hence, the best asymptotic behaviour of k_N is obtained for direct call, repeated subvector scaling, and recursive bisection, respectively. For the other precomputation methods, the size of k_N is mainly determined by the precomputation error. If we use the popular forward recursion for precomputation of twiddle factors, then the numerical stability of FFT is worse.

Proof. Let $\hat{\mathbf{x}}_0 = \mathbf{x}_0 := \mathbf{B}_N \mathbf{x}$. By $\hat{\mathbf{x}}_j$, we denote the computed vector $\hat{\mathbf{M}}_N^{(j)} \hat{\mathbf{x}}_{j-1}$ $(j = 1, \dots, n)$, i.e.

$$\hat{\mathbf{x}}_j := \mathrm{fl}\left(\hat{\mathbf{M}}_N^{(j)} \hat{\mathbf{x}}_{j-1}\right).$$

Then we introduce the error vector $\mathbf{e}_j \in \mathbb{C}^N$ of step j $(j = 1, \dots, n)$ by

$$\hat{\mathbf{x}}_j = \hat{\mathbf{M}}_N^{(j)} \hat{\mathbf{x}}_{j-1} + \mathbf{e}_j, \tag{3.7}$$

i.e., $\mathbf{e}_j$ contains the error of floating point arithmetic for computing $\hat{\mathbf{M}}_N^{(j)} \hat{\mathbf{x}}_{j-1}$ and the precomputation errors of $w_{2^j}^k$ $(k = 1, \dots, 2^{j-1} - 1)$, too.
1. Step $j = 1$: Note that

$$\hat{\mathbf{M}}_N^{(1)} = \mathbf{M}_N^{(1)} = \mathbf{I}_{2^{n-1}} \otimes \mathbf{A}_2, \quad \mathbf{A}_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

Hence the first step consists of $N/2$ butterfly operations

$$\begin{aligned} \eta_1 &:= \xi_1 + \xi_2\,, \\ \eta_2 &:= \xi_1 - \xi_2 \end{aligned} \quad (\xi_1, \xi_2 \in \mathbb{C}). \tag{3.8}$$

By (2.1) we obtain for the computed values

$$\begin{aligned} \hat{\eta}_1 &:= (\xi_1 + \xi_2)(1 + \varepsilon_1^+) = \eta_1 + (\xi_1 + \xi_2)\varepsilon_1^+\,, \\ \hat{\eta}_2 &:= (\xi_1 - \xi_2)(1 + \varepsilon_2^+) = \eta_2 + (\xi_1 + \xi_2)\varepsilon_2^+ \end{aligned} \quad (|\varepsilon_1^+|, |\varepsilon_2^+| \le u)$$

and hence

$$|\hat{\eta}_1 - \eta_1|^2 + |\hat{\eta}_2 - \eta_2|^2 \le u^2 \left(|\xi_1 + \xi_2|^2 + |\xi_1 - \xi_2|^2\right) = 2u^2 \left(|\xi_1|^2 + |\xi_2|^2\right).$$

Consequently, we have

$$\|\mathbf{e}_1\|_2 \le \sqrt{2} u \, \|\mathbf{x}_0\|_2 = \sqrt{2} u \, \|\mathbf{x}\|_2. \tag{3.9}$$

2. Step $j = 2$: Note that

$$\hat{\mathbf{M}}_N^{(2)} = \mathbf{M}_N^{(2)} = \mathbf{I}_{2^{n-2}} \otimes \mathbf{A}_4$$

with

$$\mathbf{A}_4 = \begin{pmatrix} \mathbf{I}_2 & \mathbf{W}_2 \\ \mathbf{I}_2 & -\mathbf{W}_2 \end{pmatrix}, \quad \mathbf{W}_2 = \begin{pmatrix} 1 & 0 \\ 0 & -\mathrm{i} \end{pmatrix}.$$

Then the second step consists of $N/4$ butterfly operations (3.8) and of $N/4$ butterfly operations

$$\begin{aligned} \eta_1 &:= \xi_1 - \mathrm{i}\,\xi_2\,, \\ \eta_2 &:= \xi_1 + \mathrm{i}\,\xi_2 \end{aligned} \quad (\xi_1,\, \xi_2 \in \mathbb{C}).$$

Analogously to the first step, we get the estimate $\|\mathbf{e}_2\|_2 \le \sqrt{2}u\,\|\hat{\mathbf{x}}_1\|_2$. By (3.7) and (3.9), we see that

$$\|\hat{\mathbf{x}}_1\|_2 \le \|\mathbf{M}_N^{(1)}\|_2\,\|\hat{\mathbf{x}}_0\|_2 + \|\mathbf{e}_1\|_2 \le \sqrt{2}\,(1+u)\,\|\mathbf{x}\|_2$$

and therefore

$$\|\mathbf{e}_2\|_2 \le (2u + \mathcal{O}(u^2))\,\|\mathbf{x}\|_2. \tag{3.10}$$

3. Step j ($j \in \{3,\dots,n\}$): Using the structure of $\mathbf{M}_N^{(j)}$ (see (3.2) – (3.3)), it follows that step j consists of 2^{n-j} butterfly operations of the form

$$\begin{aligned} \eta_1 &:= \xi_1 + w_{2^j}^k\,\xi_2\,, \\ \eta_2 &:= \xi_1 - w_{2^j}^k\,\xi_2 \end{aligned} \qquad (\xi_1,\,\xi_2 \in \mathbb{C}\,;\, k = 0,\dots,2^{j-1}-1). \tag{3.11}$$

Now $w_{2^j}^k$ is precomputed by

$$\hat{w}_{2^j}^k = w_{2^j}^k + \delta_{2^j}^k \quad (\delta_{2^j}^0 := 0)\,, \tag{3.12}$$

where by our assumption $|\delta_{2^j}^k| \le c_N\,u$. Using complex floating point arithmetic (see (2.1)), we obtain the computed values

$$\begin{aligned} \hat{\eta}_1 &= \big(\xi_1 + \hat{w}_{2^j}^k\,\xi_2\,(1+\varepsilon^\times)\big)(1+\varepsilon_1^+)\,, \\ \hat{\eta}_2 &= \big(\xi_1 - \hat{w}_{2^j}^k\,\xi_2\,(1+\varepsilon^\times)\big)(1+\varepsilon_2^+) \end{aligned} \tag{3.13}$$

with

$$|\varepsilon^\times| \le 2\sqrt{2}\,u\,, \quad |\varepsilon_1^+|,\,|\varepsilon_2^+| \le u\,. \tag{3.14}$$

By (3.12) and (3.13), it follows that

$$\hat{\eta}_1 - \eta_1 = \xi_1\varepsilon_1^+ + \hat{w}_{2^j}^k\,\xi_2(\varepsilon^\times + \varepsilon_1^+ + \varepsilon^\times\varepsilon_1^+) + \delta_{2^j}^k\,\xi_2 + \mathcal{O}(u^2)\,.$$

Using (3.13) and (3.14), we obtain that

$$|\hat{\eta}_1 - \eta_1| \le |\xi_1|\,u + |\xi_2|\,\big((2\sqrt{2}+1+c_N)\,u + \mathcal{O}(u^2)\big)$$

and hence

$$|\hat{\eta}_1 - \eta_1|^2 \le 2\,\big((2\sqrt{2}+1+c_N)\,u + \mathcal{O}(u^2)\big)^2(|\xi_1|^2 + |\xi_2|^2)\,.$$

The same estimate is true for $|\hat{\eta}_2 - \eta_2|^2$ such that

$$|\hat{\eta}_1 - \eta_1|^2 + |\hat{\eta}_2 - \eta_2|^2 \le 4\,\big((2\sqrt{2}+1+c_N)\,u + \mathcal{O}(u^2)\big)^2(|\xi_1|^2 + |\xi_2|^2)\,.$$

Thus for the error vector $\mathbf{e}_j$, we obtain the estimate

$$\|\mathbf{e}_j\|_2 \leq 2\left((2\sqrt{2}+1+c_N)\,u+\mathcal{O}(u^2)\right)\|\hat{\mathbf{x}}_{j-1}\|_2\,.$$

From (3.4) and (3.7), it follows that

$$\|\hat{\mathbf{x}}_{j-1}\|_2 \leq \left(2^{(j-1)/2}+\mathcal{O}(u)\right)\|\mathbf{x}\|_2$$

and hence

$$\|\mathbf{e}_j\|_2 \leq 2^{(j+1)/2}\left((2\sqrt{2}+1\,+\,c_N)\,u\,+\,\mathcal{O}(u^2)\right)\|\mathbf{x}\|_2. \tag{3.15}$$

4. Now we estimate the roundoff error $\|\hat{\mathbf{x}}_n-\mathbf{x}_n\|_2$. Applying repeatedly (3.7), we obtain

$$\begin{aligned}
\hat{\mathbf{x}}_n &= \mathbf{M}_N^{(n)}\,\hat{\mathbf{x}}_{n-1}+\mathbf{e}_n\\
&= \mathbf{M}_N^{(n)}\,\mathbf{M}_N^{(n-1)}\,\hat{\mathbf{x}}_{n-2}+\mathbf{M}_N^{(n)}\,\mathbf{e}_{n-1}+\mathbf{e}_n\\
&\;\;\vdots\\
&= \mathbf{M}_N^{(n)}\ldots\mathbf{M}_N^{(1)}\,\mathbf{B}_N\,\mathbf{x}+\mathbf{M}_N^{(n)}\ldots\mathbf{M}_N^{(2)}\,\mathbf{e}_1+\ldots+\mathbf{M}_N^{(n)}\mathbf{e}_{n-1}+\mathbf{e}_n
\end{aligned}$$

such that by $\mathbf{x}_n=\mathbf{M}_N^{(n)}\ldots\mathbf{M}_N^{(1)}\,\mathbf{B}_N\,\mathbf{x}$ (see Algorithm 3.2) and by (3.4), (3.9), (3.10) and (3.15)

$$\begin{aligned}
\|\hat{\mathbf{x}}_n-\mathbf{x}_n\|_2 &\leq \|\mathbf{M}_N^{(n)}\ldots\mathbf{M}_N^{(2)}\|_2\|\mathbf{e}_1\|_2+\ldots+\|\mathbf{M}_N^{(n)}\|_2\|\mathbf{e}_{n-1}\|_2+\|\mathbf{e}_n\|_2\\
&\leq 2^{n/2}\Big(2u+(n-2)\big((\sqrt{2}\,c_N+4+\sqrt{2})\,u+\mathcal{O}(u^2)\big)\Big)\,\|\mathbf{x}\|_2.
\end{aligned}$$

5. The final step of Algorithm 3.2 is the scaling $\mathbf{y}=2^{-n/2}\mathbf{x}_n$. Let $\hat{\mathbf{y}}:=\mathrm{fl}(2^{-n/2}\hat{\mathbf{x}}_n)$. Using (2.2), (3.4) and (3.7), we obtain that

$$\|\hat{\mathbf{y}}-2^{-n/2}\,\hat{\mathbf{x}}_n\|_2 \leq 2^{-n/2}u\,\|\hat{\mathbf{x}}_n\|_2 \leq \left(u+\mathcal{O}(u^2)\right)\|\mathbf{x}\|_2$$

and hence

$$\begin{aligned}
\|\hat{\mathbf{y}}-\mathbf{y}\|_2 &\leq \|\hat{\mathbf{y}}-2^{-n/2}\hat{\mathbf{x}}_n\|_2+2^{-n/2}\,\|\hat{\mathbf{x}}_n-\mathbf{x}_n\|_2\\
&\leq \left((\sqrt{2}\,c_N+4+\sqrt{2})(n-2)\,u+3u+\mathcal{O}(u^2)\right)\|\mathbf{x}\|_2\,.
\end{aligned}$$

This completes the proof. ■

For $\mathbf{x}=(x_k)_{k=0}^{N-1}\in\mathbb{C}$ we introduce the norms

$$\|\mathbf{x}\|_1:=\sum_{k=0}^{N-1}|x_k|\,,\quad \|\mathbf{x}\|_\infty:=\max_{k=-N/2,\ldots,N/2}|x_k|\,.$$

Corollary 3.4 *Under the assumptions of Theorem* 3.3 *we have the estimates*

$$\|\Delta\mathbf{x}\|_\infty \le \left(k_N\, u + \mathcal{O}(u^2)\right) \|\mathbf{x}\|_1\,, \quad \|\hat{\mathbf{y}} - \mathbf{y}\|_\infty \le \left(k_N\, u + \mathcal{O}(u^2)\right) \|\mathbf{y}\|_1\,.$$

Proof. Use $\|\mathbf{x}\|_\infty \le \|\mathbf{x}\|_2 \le \|\mathbf{x}\|_1$ $(\mathbf{x} \in \mathbb{C}^N)$ in the results of Theorem 3.3. ∎

By taking the transpose of the Cooley–Tukey factorization (3.1), we obtain another sparse unitary factorization of the Fourier matrix

$$\mathbf{F}_N = 2^{-n/2}\, \mathbf{B}_N\, (\mathbf{M}_N^{(1)})^{\mathrm{T}}\, (\mathbf{M}_N^{(2)})^{\mathrm{T}} \ldots\, (\mathbf{M}_N^{(n)})^{\mathrm{T}} \quad (N = 2^n).$$

On this fact, the following FFT (see [15, pp. 65 – 67]) is based:

Algorithm 3.5 (Gentleman–Sande algorithm)

Input: $N := 2^n$ $(n \ge 5)$, $\mathbf{x} \in \mathbb{C}^N$.
0. *Precompute* w_N^k $(k = 1, \ldots, N_1 - 1)$ *by a method of Section* 2.
1. *For* $j = 0\,(1)\,n-1$ *form*

$$\mathbf{x}_{j+1} := (\mathbf{M}_N^{(n-j)})^{\mathrm{T}} \mathbf{x}_j \quad (\mathbf{x}_0 := \mathbf{x})\,. \tag{3.16}$$

2. *Permute*

$$\mathbf{x}_{n+1} := \mathbf{B}_N \mathbf{x}_n.$$

3. *Multiply*

$$\mathbf{y} := 2^{-n/2} \mathbf{x}_n.$$

Output: $\mathbf{y} = \mathbf{F}_N \mathbf{x}$.

Algorithm 3.5 is an example of a *decimation–in–frequency* FFT. Comparing with Algorithm 3.2, we see that other butterfly operations occur in (3.16), namely

$$\begin{aligned} \eta_1 &= \xi_1 + \xi_2\,, \\ \eta_2 &= (\xi_1 - \xi_2)\, w_{2^{n-j}}^k \end{aligned} \qquad (j = 0, \ldots, n-1\,; k = 0, \ldots, 2^{n-j} - 1)\,.$$

For that reason, we obtain another constant k_N for Algorithm 3.5.

Theorem 3.6 *Let* $N = 2^n$ $(n \ge 5)$. *Assume that*

$$|\hat{w}_N^{N_j k} - w_{2^j}^k| \le c_N\, u \quad (j = 3, \ldots, n;\; k = 1, \ldots, 2^{j-1} - 1)\,.$$

Then the Gentleman–Sande Algorithm 3.5 *is normwise backward stable with the constant*

$$k_N = (c_N + 2\sqrt{2} + 1) \log_2 N\,.$$

The proof is omitted here, since it follows similar lines as the proof of Theorem 3.3.

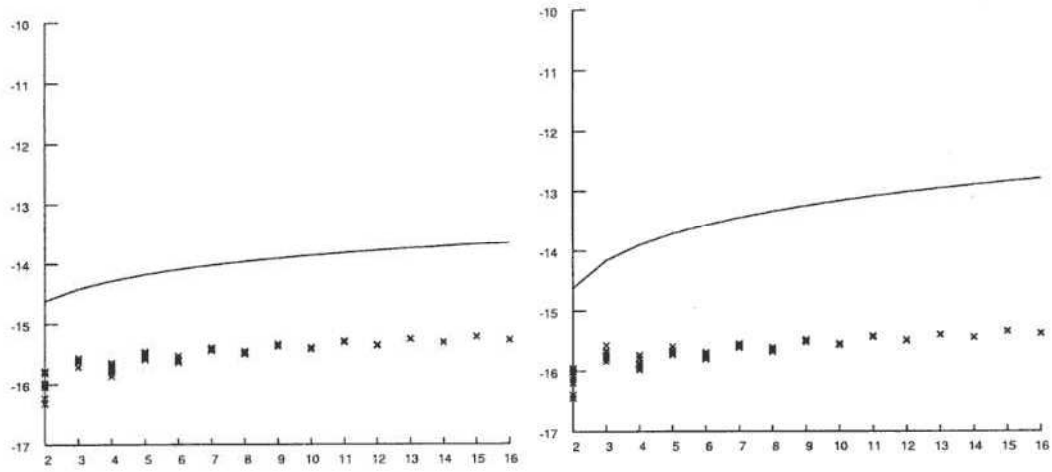

Figure 1. Relative errors (3.17) and theoretical error bounds for direct call (left) and repeated subvector scaling (right)

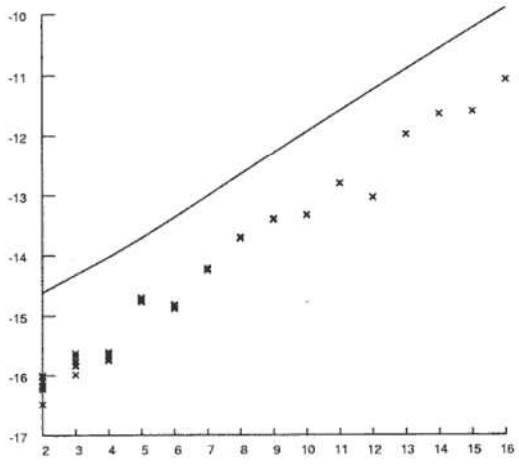

Figure 2. Relative errors (3.17) and theoretical error bound for repeated multiplication

Finally, we confirm our theoretical results by numerical experiments. As example, we choose 10 vectors $\mathbf{x} := (x_k)_{k=0}^{2^{16}-1} \in \mathbb{C}^{2^n}$ with random $\operatorname{Re} x_k$ and $\operatorname{Im} x_k$ from

normal $N(0,1)$ distribution. For each subsampled vector

$$\mathbf{x}_{2^n} := (x_{2^{16-n}k})_{k=0}^{2^n-1} \quad (n = 2, \ldots, 16),$$

we evaluate

$$\tilde{\mathbf{x}}_{2^n} = \mathbf{J}'_{2^n}\, \mathrm{fl}\,(\mathbf{F}_{2^n}\ \mathrm{fl}\,(\mathbf{F}_{2^n}\, \mathbf{x}_{2^n}))$$

by Cooley–Tukey Algorithm 3.2 in double precision, where $\mathbf{J}'_{2^n}$ denotes the flip matrix $\mathbf{J}'_{2^n} := 1 \oplus \mathbf{J}_{2^n-1}$, $\mathbf{J}_N := (\delta_{j,N-1-k})_{j,k=0}^{N-1}$ means the counteridentity matrix, and where the twiddle factors are precomputed by Algorithm 2.1, 2.2, and 2.3, respectively. Note that $\mathbf{F}_{2^n}^{-1} = \mathbf{J}'_{2^n}\mathbf{F}_{2^n}$. Figures 1 – 2 show the behaviour of the relative errors

$$\log_{10}\left(\|\tilde{\mathbf{x}}_{2^n} - \mathbf{x}_{2^n}\|_2 / \|\mathbf{x}_{2^n}\|_2\right) \quad (n = 2, \ldots, 16) \tag{3.17}$$

and the approximate error bound $\log_{10}(2k_{2^n}u)$ of Theorem 3.3. The algorithms were implemented in C and tested in double precision.

4 Numerical stability of DCT and DST

We introduce four discrete cosine transforms (DCT) and four discrete sine transforms (DST) as classified by Wang [27] (see also [19, pp. 11 – 21]). These transforms are generated by the following matrices:

$$\begin{aligned}
\text{DCT–I} \quad &: \quad \mathbf{C}_{N+1}^{I} := \left(\frac{2}{N}\right)^{1/2} \left(\epsilon_j^N \epsilon_k^N \cos\frac{jk\pi}{N}\right)_{j,k=0}^{N} \in \mathbb{R}^{N+1,N+1},\\
\text{DCT–II} \quad &: \quad \mathbf{C}_{N}^{II} := \left(\frac{2}{N}\right)^{1/2} \left(\epsilon_j^N \cos\frac{j(2k+1)\pi}{2N}\right)_{j,k=0}^{N-1} \in \mathbb{R}^{N,N},\\
\text{DCT–III} \quad &: \quad \mathbf{C}_{N}^{III} := \left(\mathbf{C}_{N}^{II}\right)^{\mathrm{T}} \in \mathbb{R}^{N,N},\\
\text{DCT–IV} \quad &: \quad \mathbf{C}_{N}^{IV} := \left(\frac{2}{N}\right)^{1/2} \left(\cos\frac{(2j+1)(2k+1)\pi}{4N}\right)_{j,k=0}^{N-1} \in \mathbb{R}^{N,N}
\end{aligned}$$

and

$$\begin{aligned}
\text{DST–I} \quad &: \quad \mathbf{S}_{N-1}^{I} := \left(\frac{2}{N}\right)^{1/2} \left(\sin\frac{(j+1)(k+1)\pi}{N}\right)_{j,k=0}^{N-2} \in \mathbb{R}^{N-1,N-1},\\
\text{DST–II} \quad &: \quad \mathbf{S}_{N}^{II} := \left(\frac{2}{N}\right)^{1/2} \left(\epsilon_{j+1}^N \sin\frac{(j+1)(2k+1)\pi}{2N}\right)_{j,k=0}^{N-1} \in \mathbb{R}^{N,N},\\
\text{DST–III} \quad &: \quad \mathbf{S}_{N}^{III} := \left(\mathbf{S}_{N}^{II}\right)^{\mathrm{T}} \in \mathbb{R}^{N,N},\\
\text{DST–IV} \quad &: \quad \mathbf{S}_{N}^{IV} := \left(\frac{2}{N}\right)^{1/2} \left(\sin\frac{(2j+1)(2k+1)\pi}{4N}\right)_{j,k=0}^{N-1} \in \mathbb{R}^{N,N},
\end{aligned}$$

where $\epsilon_k^N := \sqrt{2}/2$ $(k = 0, N)$ and $\epsilon_k^N := 1$ otherwise. We refer to the corresponding transforms as *discrete trigonometric transforms*. It is well–known that the above matrices are orthogonal and that a fast multiplication of such matrix with an arbitrary vector takes only $\mathcal{O}(N \log N)$ arithmetical operations.
We consider the orthogonal matrix $\mathbf{C}_N^{III}$. Let $N = 2^n$ $(n > 1)$. By

$$\cos \frac{(2(2N-2j-1)+1)k\pi}{2N} = \cos \frac{(4j+1)k\pi}{2N}$$

we obtain the representation

$$\tilde{\mathbf{E}}_N \mathbf{C}_N^{III} = \sqrt{\frac{2}{N}}\, \tilde{\mathbf{C}}_N \mathbf{D}_N\,, \tag{4.1}$$

where

$$\tilde{\mathbf{C}}_N := \left(\cos \frac{(4j+1)k\pi}{2N}\right)_{j,k=0}^{N-1}, \quad \mathbf{D}_N := \operatorname{diag}\left(\epsilon_k^N\right)_{k=0}^{N-1}$$

and where $\tilde{\mathbf{E}}_N$ denotes the *modified even–odd permutation matrix*

$$\tilde{\mathbf{E}}_N := \begin{pmatrix} 1 & 0 & & & & & & \\ & & 1 & 0 & & & & \\ & & & & 1 & 0 & & \\ & & & & & & \ddots & \\ & & & & & & 1 & 0 \\ & & & & & & 0 & 1 \\ & & & & 0 & 1 & & \\ & & & 0 & 1 & & & \\ 0 & 1 & \iddots & & & & & \end{pmatrix} \in \mathbb{R}^{N,N}.$$

Note that the *even–odd permutation matrix*

$$\mathbf{E}_N := \begin{pmatrix} 1 & 0 & & & & & & \\ & & 1 & 0 & & & & \\ & & & & 1 & 0 & & \\ & & & & & & \ddots & \\ & & & & & & 1 & 0 \\ 0 & 1 & & & & & & \\ & & 0 & 1 & & & & \\ & & & & 0 & 1 & & \\ & & & & & & \ddots & \\ & & & & & & 0 & 1 \end{pmatrix} \in \mathbb{R}^{N,N}$$

is related to the bit–reversal permutation matrix $\mathbf{B}_N$ (see [15, pp. 20 – 21]) by

$$\mathbf{B}_N = (\mathbf{I}_{N_2} \otimes \mathbf{E}_4) \ldots (\mathbf{I}_2 \otimes \mathbf{E}_{N_1}) \mathbf{E}_N\,. \tag{4.2}$$

As known, fast algorithms for computing $\mathbf{y} = \mathbf{C}_N^{III}\mathbf{x}$ for arbitrary given vectors $\mathbf{x} \in \mathbb{R}^N$ are based on a factorization of $\tilde{\mathbf{C}}_N$ into a product of sparse matrices. By observing

$$\tilde{\mathbf{C}}_N = \operatorname{Re}\left(w_{4N}^{(4j+1)k}\right)_{j,k=0}^{N-1} = \sqrt{N}\operatorname{Re}\left(\mathbf{F}_N \operatorname{diag}\left(w_{4N}^k\right)_{k=0}^{N-1}\right)$$

with the Fourier matrix $\mathbf{F}_N$ defined in (1.1), we obtain

$$\mathbf{C}_N^{III} = \sqrt{2}\operatorname{Re}\left(\tilde{\mathbf{E}}_N^{\mathrm{T}}\mathbf{F}_N \operatorname{diag}\left(\epsilon_k^N w_{4N}^k\right)_{k=0}^{N-1}\right). \tag{4.3}$$

Using the Cooley–Tukey factorization (3.1) of $\mathbf{F}_N$, we get:

Theorem 4.1 *Let $N = 2^n$ $(n > 1)$. Then $\mathbf{C}_N^{III}$ has the complex factorization*

$$\mathbf{C}_N^{III} = \operatorname{Re}\left(\tilde{\mathbf{E}}_N^{\mathrm{T}}\mathbf{M}_N^{(n-1)}\ldots\mathbf{M}_N^{(1)}\mathbf{B}_N \operatorname{diag}\left(\sqrt{2}\,\epsilon_k^N w_{4N}^k\right)_{k=0}^{N-1}\right). \tag{4.4}$$

This complex factorization is closely related to a fast complex algorithm for computing $\mathbf{y} = \mathbf{C}_N^{III}\mathbf{x}$. This algorithm is mainly based on the FFT of length N, but not on a FFT of length $2N$ (compare with [15, pp. 49 – 53]).

Algorithm 4.2 (Fast DCT–III via FFT)
Input: $N = 2^n$ $(n \geq 5)$, $\mathbf{x} = (x_k)_{k=0}^{N-1} \in \mathbb{R}^N$.
1. *Form*

$$\mathbf{y}_0 := \left(\sqrt{2}\,\epsilon_k^N\, w_{4N}^k\, x_k\right)_{k=0}^{N-1}.$$

2. *Form* $\mathbf{y}_1 := \mathbf{F}_N\mathbf{y}_0$ *by Algorithm* 3.2.
3. *Take the real part of* $\mathbf{y}_1$, *i.e.*

$$\mathbf{y}_2 := \operatorname{Re}\mathbf{y}_1.$$

4. *Permute*

$$\mathbf{y} := \tilde{\mathbf{E}}_N^{\mathrm{T}}\,\mathbf{y}_2.$$

Output: $\mathbf{y} := \mathbf{C}_N^{III}\,\mathbf{x}$.

This algorithm needs $3Nn$ real additions and $\frac{5}{2}nN + 2N$ real multiplications, if one complex multiplication is realized by two real additions and four real multiplications. Let $\mathbf{x} \in \mathbb{R}^N$ be an arbitrary input vector and $\mathbf{y} = \mathbf{C}_N^{III}\mathbf{x}$ be the exact output vector. Let $\hat{\mathbf{y}}$ be the computed vector using floating point arithmetic with unit roundoff u. Since $\mathbf{C}_N^{III}$ is regular, $\hat{\mathbf{y}}$ can be represented in the form

$$\hat{\mathbf{y}} = \mathbf{C}_N^{III}(\mathbf{x} + \Delta\mathbf{x}) \quad (\Delta\mathbf{x} \in \mathbb{R}^N).$$

Note that an algorithm used for computing $\mathbf{C}_N^{III}\mathbf{x}$ is normwise backward stable, if there exists a positive constant k_N^* with $k_N^* u \ll 1$ such that for all vectors $\mathbf{x} \in \mathbb{R}^N$

$$\|\Delta\mathbf{x}\|_2 \le \left(k_N^* u + \mathcal{O}(u^2)\right)\|\mathbf{x}\|_2.$$

Since $\mathbf{C}_N^{III}$ is orthogonal, we have also normwise forward stability by

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_2 \le \left(k_N^* u + \mathcal{O}(u^2)\right)\|\mathbf{y}\|_2 .$$

We precompute the needed roots of unity w_{4N}^k $(k = 0, \dots, N-1)$ by means of

$$w_{4N}^{4k+l} \;=\; w_{4N}^l \, w_N^k \quad (l = 0, 1, 2, 3;\; k = 0, \dots, N_2 - 1) .$$

This implies by (2.1) and (3.5) that

$$|\hat{w}_{4N}^k \;-\; w_{4N}^k| \le (c_N + \frac{5}{2}\sqrt{2})\, u \;+\; \mathcal{O}(u^2) \quad (k = 1, \dots, N-1) . \tag{4.5}$$

Theorem 4.3 *Under the assumptions of Theorem* 3.3*, Algorithm* 4.2 *is normwise backward stable with constant*

$$k_N^* \;= \sqrt{2}\,(c_N + k_N) ,$$

where k_N *is given by* (3.6).

Proof. Algorithm 4.2 is based on (4.4) and the FFT. As usual we denote the computed vectors with a hat accent.
1. First we analyze the error of step 1 in Algorithm 4.2. We obtain an error vector $\mathbf{e}_0 := (e_{0,k})_{k=0}^{N-1} \in \mathbb{C}^N$ such that $\hat{\mathbf{y}}_0 = \mathbf{y}_0 + \mathbf{e}_0$. By

$$\hat{y}_{0,k} = \sqrt{2}\epsilon_k^N \hat{w}_{4N}^k x_k (1 + \varepsilon_k^{\times}) , \quad |\varepsilon_k^{\times}| \le 2\, u$$

and by (2.2) and (4.5) we conclude that

$$|\hat{y}_{0,k} - y_{0,k}| \;=\; |e_{0,k}| \le (\sqrt{2}\, c_N + \,8)\, u\, |x_k| \quad (k = -N/2, \dots, N/2)$$

and hence

$$\|\mathbf{e}_0\|_2 \le (\sqrt{2}\, c_N + \,8)\, u\, \|\mathbf{x}\|_2 ,$$

where the $\mathcal{O}(u^2)$–term is omitted. Consequently, we have

$$\|\hat{\mathbf{y}}_0\|_2 \le \|\mathbf{y}_0\|_2 + \|\mathbf{e}_0\|_2 \le \left(\sqrt{2} + (\sqrt{2}\, c_N \,+\, 8)\, u\right)\|\mathbf{x}\|_2 . \tag{4.6}$$

2. Step 2 implies the FFT of length N. By Theorem 3.3, we know that

$$\hat{\mathbf{y}}_1 = \mathbf{F}_N(\hat{\mathbf{y}}_0 + \Delta\hat{\mathbf{y}}_0) , \tag{4.7}$$

where $\Delta\hat{\mathbf{y}}_0 \in \mathbb{C}^N$ fulfills

$$\|\Delta\hat{\mathbf{y}}_0\|_2 \leq \left(k_N\, u + \mathcal{O}(u^2)\right) \|\hat{\mathbf{y}}_0\|_2\,.$$

Then by (4.6), it follows that

$$\|\Delta\hat{\mathbf{y}}_0\|_2 \leq \left(\sqrt{2}\, k_N u + \mathcal{O}(u^2)\right)\|\mathbf{x}\|_2\,.$$

By (4.7), we get

$$\hat{\mathbf{y}}_1 - \mathbf{y}_1 = \mathbf{F}_N(\mathbf{e}_0 + \Delta\hat{\mathbf{y}}_0)$$

with

$$\|\mathbf{F}_N(\mathbf{e}_0 + \Delta\hat{\mathbf{y}}_0)\|_2 \leq \left(\sqrt{2}\,(c_N + 4\sqrt{2} + k_N)\, u + \mathcal{O}(u^2)\right)\|\mathbf{x}\|_2.$$

This completes the proof, since steps 3 and 4 contain no further floating point operations. ■

We discuss the numerical stability for computing $\mathbf{C}\,\mathbf{x}$ with $\mathbf{C} \in \{\mathbf{C}_{N+1}^I, \mathbf{C}_N^{II}, \mathbf{C}_N^{IV}\}$. Using

$$\begin{aligned} \mathbf{C}_{N+1}^I &= \tfrac{1}{2}\,\mathrm{Re}\left(\mathbf{R}\,\mathbf{F}_{2N}\,\mathbf{P}\right), \quad \mathbf{C}_N^{II} = (\mathbf{C}_N^{III})^{\mathrm{T}} \\ \mathbf{C}_N^{IV} &= \mathrm{Re}\left(w_{4N}^{(2j+1)(2k+1)}\right)_{j,k=0}^{N-1} = \sqrt{2}\,\mathrm{Re}\left(\mathrm{diag}\left(w_{4N}^{2j+1}\right)_{j=0}^{N-1}\mathbf{F}_N\,\mathrm{diag}\left(w_{2N}^{k}\right)_{k=0}^{N-1}\right) \end{aligned} \tag{4.8}$$

with

$$\mathbf{R} := \begin{pmatrix} 1/\sqrt{2} & 0 & 0 & \cdots & & & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & & & \cdots & 0 & 1 \\ \vdots & & \ddots & & & & & \iddots & \\ 0 & & & 1 & 0 & 1 & & & 0 \\ 0 & & \cdots & 0 & 1/\sqrt{2} & 0 & \cdots & & 0 \end{pmatrix} \in \mathbb{R}^{N+1,2N}$$

and

$$\mathbf{P} := \begin{pmatrix} \sqrt{2} & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & & & 1 & 0 \\ 0 & & \cdots & 0 & \sqrt{2} \\ 0 & & & 1 & 0 \\ \vdots & & \iddots & & \vdots \\ 0 & 1 & 0 & \cdots & 0 \end{pmatrix} \in \mathbb{R}^{2N,N+1}\,,$$

we obtain a complex factorization for $\mathbf{C}_{N+1}^I, \mathbf{C}_N^{II}$ and $\mathbf{C}_N^{IV}$.

The numerical stability of the FFT has a strong influence on the numerical stability of the DCT. Theorem 4.3 and straightforward computation implies that an algorithm based on (4.8) for computing $\mathbf{C}\,\mathbf{x}$ with $\mathbf{C} \in \{\mathbf{C}_{N+1}^I, \mathbf{C}_N^{II}, \mathbf{C}_N^{III}, \mathbf{C}_N^{IV}\}$ is normwise backward stable with

$$k_N^* = \begin{cases} \mathcal{O}(\log_2 N) & \text{for direct call,} \\ \mathcal{O}((\log_2 N)^2) & \text{for repeated subvector scaling and recursive bisection,} \\ \mathcal{O}(N \log_2 N) & \text{for repeated multiplication, stabilized forward recursion and logarithmic recursion,} \\ \mathcal{O}(N^2 \log_2 N) & \text{for forward recursion.} \end{cases}$$

We consider now the DST. By observing

$$\mathbf{S}_{N-1}^I \;=\; \frac{1}{2}\,\mathrm{Re}\left(\tilde{\mathbf{R}}\,\mathbf{F}_{2N}\,\tilde{\mathbf{R}}^{\mathrm{T}}\right)$$

with

$$\tilde{\mathbf{R}} := \begin{pmatrix} 0 & 1 & 0 & \cdots & & \cdots & 0 & 1 \\ \vdots & & \ddots & & & & \iddots & \\ 0 & & & 1 & 0 & 1 & & 0 \end{pmatrix} \in \mathbb{R}^{N-1,2N}$$

we obtain a complex factorization. Fast algorithms for the remaining DST–II, DST–III and DST–IV follow in a simple way by intertwining relations (see [11]): Let $\Sigma_N := \mathrm{diag}\,((-1)^k)_{k=0}^{N-1}$. Then

$$\mathbf{S}_N^{II} = \mathbf{J}_N\,\mathbf{C}_N^{II}\,\Sigma_N\,, \quad \mathbf{S}_N^{III} = \Sigma_N\,\mathbf{C}_N^{III}\,\mathbf{J}_N\,, \quad \mathbf{S}_N^{IV} = \Sigma_N\,\mathbf{C}_N^{IV}\,\mathbf{J}_N\,.$$

Since multiplications with $\mathbf{J}_N$ and Σ_N contain no further floating point operations, we obtain the numerical stability immediately.
There exist algorithms for the fast multiplication of the above sine and cosine matrices with an arbitrary vector which are based on direct factorizations of the corresponding matrices into sparse (but unfortunately not orthogonal) matrices [21, 23, 3]. These algorithms avoid arithmetic with complex numbers and have a lower arithmetical complexity than Algorithm 4.2. However, in [2] it was proved that the numerical stability of the FFT–based Algorithm 4.2 is better than the numerical stability of these real fast algorithms.

5 Robustness of NDFT

In the last section, we are interested in discrete Fourier transforms for nonequispaced data (NDFT). Let $N \in \mathbb{N}$ be a power of two. We want to evaluate the 1–periodic trigonometric polynomial

$$f(v) := \sum_{k=-N/2}^{N/2-1} f_k\,\mathrm{e}^{-2\pi\mathrm{i}kv} \tag{5.1}$$

at arbitrary nodes $w_j \in [-\frac{1}{2}, \frac{1}{2})$ $(j = -N/2, \ldots, N/2-1)$. Again, it will be useful to rewrite (5.1) in matrix–vector notation

$$\hat{\mathbf{f}} = \mathbf{A}_N \mathbf{f}\,,$$

where

$$\hat{\mathbf{f}} := (f(w_j))_{j=-N/2}^{N/2-1}\,, \quad \mathbf{f} := (f_k)_{k=-N/2}^{N/2-1}\,, \quad \mathbf{A}_N := \left(\mathrm{e}^{-2\pi \mathrm{i} k w_j}\right)_{j,k=-N/2}^{N/2-1}\,.$$

For the efficient realization of (5.1) we use the following approach [22]: We introduce the *oversampling factor* $\alpha > 1$ and set $M := \alpha N$. Let φ be a 1–periodic function with uniformly convergent Fourier series. We approximate f by

$$s_1(v) := \sum_{l=-N/2}^{N/2-1} g_l\, \varphi(v - \frac{l}{M}). \tag{5.2}$$

Switching to the frequency domain, we obtain

$$\begin{aligned} s_1(v) &= \sum_{k\in\mathbb{Z}} \hat{g}_k\, c_k(\varphi)\, \mathrm{e}^{-2\pi \mathrm{i} k v} \qquad (5.3)\\ &= \sum_{k=-M/2}^{M/2-1} \hat{g}_k\, c_k(\varphi)\, \mathrm{e}^{-2\pi \mathrm{i} k v} + \sum_{r\in\mathbb{Z}\setminus\{0\}} \sum_{k=-M/2}^{M/2-1} \hat{g}_k\, c_{k+Mr}(\varphi)\, \mathrm{e}^{-2\pi \mathrm{i}(k+Mr)v} \end{aligned}$$

with

$$\hat{g}_k := \sum_{l=-M/2}^{M/2-1} g_l\, \mathrm{e}^{2\pi \mathrm{i} k l/M}, \tag{5.4}$$

$$c_k(\varphi) := \int_{-1/2}^{1/2} \varphi(v)\, \mathrm{e}^{2\pi \mathrm{i} k v}\, \mathrm{d}v \quad (k \in \mathbb{Z}).$$

If the Fourier coefficients $c_k(\varphi)$ become sufficiently small for large $|k|$ and if $c_k(\varphi) \neq 0$ for $k = -N/2, \ldots, N/2-1$, then we suggest by comparing (5.1) with (5.3) to set

$$\hat{g}_k := \begin{cases} f_k / c_k(\varphi) & k = -N/2, \ldots, N/2-1, \\ 0 & k = -M/2, \ldots, -N/2-1; N/2, \ldots, M/2-1. \end{cases} \tag{5.5}$$

Now the values g_l can be obtained from (5.4) by the reduced inverse FFT of size M. If φ is also well–localized in time domain such that it can be approximated by a 1–periodic function ψ with $\operatorname{supp}\psi \cap [-\frac{1}{2}, \frac{1}{2}) \subseteq [-\frac{m}{M}, \frac{m}{M})$ $(2m \ll M)$, then

$$f(w_j) \approx s_1(w_j) \approx s(w_j) = \sum_{l\in I_{M,m}(w_j)} g_l\, \psi(w_j - \frac{l}{M}) \tag{5.6}$$

with $I_{M,m}(w_j) := \{l = -N/2, \ldots, N/2-1 : Mw_j - m \le l \le Mw_j + m\}$. For fixed $w_j \in [-1/2, 1/2)$, the above sum contains at most $2m+2$ nonzero summands.
In summary, we obtain the following algorithm for the fast computation of (5.1) with $\mathcal{O}(\alpha N \log(\alpha N))$ arithmetical operations:

Algorithm 5.1 (Fast computation of NDFT (5.1))

Input*:* $N \in \mathbb{N}$, $\alpha > 1$, $M := \alpha N$, $w_j \in [-\frac{1}{2}, \frac{1}{2})$, $f_k \in \mathbb{C}$ $(j, k = -N/2, \ldots, N/2-1)$.

0. *Precompute* $c_k(\varphi)$ $(k = -N/2, \ldots, N/2-1)$, $\psi(w_j - \frac{l}{M})$ $(j = -N/2, \ldots, N/2-1$*;* $l \in I_{M,m}(w_j))$.
1. *Form* $\hat{g}_k := f_k / c_k(\varphi) \quad (k = -N/2, \ldots, N/2-1)$.
2. *Compute by a modified Cooley–Tukey Algorithm* 3.2

$$g_l := M^{-1} \sum_{k=-N/2}^{N/2-1} \hat{g}_k \, \mathrm{e}^{-2\pi \mathrm{i} k l / M} \quad (l = -M/2, \ldots, M/2-1).$$

3. *Set*

$$s(w_j) := \sum_{l \in I_{M,m}(w_j)} g_l \, \psi(w_j - \frac{l}{M}) \quad (j = -N/2, \ldots, N/2-1) \, .$$

Output: $s(w_j)$ *approximate value of* $f(w_j)$ $(j = -N/2, \ldots, N/2-1)$.

Suitable functions φ are dilated, periodized Gaussian bells (see [10])

$$\varphi(v) := (\pi b)^{-1/2} \sum_{r \in \mathbb{Z}} \mathrm{e}^{-(M(v+r))^2/b} \, , \tag{5.7}$$

dilated, periodized, centered cardinal B–splines of order $2m$ (see [4])

$$\varphi(v) := \sum_{r \in \mathbb{Z}} \mathrm{M}_{2m}(M(v+r)) \quad (m \ge 1) \tag{5.8}$$

or other window functions as prolate spheroidal functions and Kaiser–Bessel functions [14], Gaussian kernels tapered with a Hanning window [9], Gaussian kernels combined with sinc kernels [16] or special optimized windows [14, 9]. If φ has not "local support", then one can use a truncated version of φ as ψ, for example with respect to (5.7)

$$\psi(v) := (\pi b)^{-1/2} \sum_{r \in \mathbb{Z}} \mathrm{e}^{-(M(v+r))^2/b} \, \chi_{[-m,m]}(M(v+r)) \, . \tag{5.9}$$

Here $\chi_{[-m,m]}$ denotes the characteristic function of $[-m, m]$. Estimates of the approximation error

$$\max\{|f(w_j) - s(w_j)| : j = -N/2, \ldots, N/2-1\}$$

for the window functions (5.7) and (5.8) are given in [10, 4, 9, 22].
We are interested in stability properties of Algorithm 5.1. Let

$$\tilde{\mathbf{F}}_M := \frac{1}{\sqrt{M}} \left(\mathrm{e}^{-2\pi i j k/M} \right)_{j,k=-M/2}^{M/2-1} = (\mathbf{J}_2 \otimes \mathbf{I}_{M/2}) \mathbf{F}_M (\mathbf{J}_2 \otimes \mathbf{I}_{M/2}) \,.$$

Clearly, the multiplication of $\tilde{\mathbf{F}}_M$ with a vector can be realized by FFT with the same stability estimates as in Section 3. Now Algorithm 5.1 reads in matrix–vector notation as

$$\mathbf{A}_N \mathbf{f} \approx \sqrt{M}\, \mathbf{B} \tilde{\mathbf{F}}_M \mathbf{D} \mathbf{f} \,,$$

where $\mathbf{B}$ denotes the sparse matrix

$$\mathbf{B} := \left(\psi(w_j - \frac{l}{M}) \right)_{j=-N/2, l=-M/2}^{N/2-1, M/2-1} \tag{5.10}$$

and

$$\mathbf{D} := \left(\mathbf{O} \,|\, \mathrm{diag}(1/(M\, c_k(\varphi)))_{k=-N/2}^{N/2-1} \,|\, \mathbf{O} \right)^{\mathrm{T}} \tag{5.11}$$

with $(N,(M-N)/2)$–zero matrices $\mathbf{O}$.

Theorem 5.2 *Let $m, N \in \mathbb{N}$ and let $M := \alpha N$ $(\alpha > 1)$ be a power of 2 with $2m \ll M$. Let h be a nonnegative even function, which decreases monotonically in $[0, \infty)$, and let*

$$\varphi(x) := \sum_{r \in \mathbb{Z}} h(M(x+r)) \,, \quad \psi(x) := \sum_{r \in \mathbb{Z}} (\chi_{[-m,m]} h)(M(x+r)) \,.$$

Suppose that φ has a uniform convergent Fourier expansion with monotone decreasing absolute values of Fourier coefficients

$$c_k(\varphi) = \frac{1}{M}\, \hat{h}(\frac{2\pi k}{M}) \quad (k \in \mathbb{Z}) \,.$$

Let the nodes $w_j \in [-\frac{1}{2}, \frac{1}{2})$, $w_j \pm 1$ $(j \in -N/2, \ldots, N/2-1)$ be distributed such that each "window" $\left[-\frac{m}{M} + \frac{l}{M}, \frac{m}{M} + \frac{l}{M}\right)$ $(l = -M/2, \ldots, M/2-1)$ contains at most γ/α nodes. If (5.1) is computed by Algorithm 5.1 with the above functions φ, ψ, i.e.,

$$\tilde{\mathbf{f}} := \sqrt{M}\, \mathbf{B}\, \tilde{\mathbf{F}}_M\, \mathbf{D}\, \mathbf{f} \quad (\mathbf{f} \in \mathbb{R}^N) \,,$$

where $\mathbf{D} \in \mathbb{C}^{M,N}$ and $\mathbf{B} \in \mathbb{R}^{N,M}$ are determined by (5.10) – (5.11), then the roundoff error of Algorithm 5.1 can be estimated by

$$\| \mathrm{fl}(\tilde{\mathbf{f}}) - \tilde{\mathbf{f}} \|_2 \le \left(\beta \sqrt{\gamma\, N}\, (k_M + 2m)\, u + \mathcal{O}(u^2) \right) \| \mathbf{f} \|_2 \,,$$

where k_M is defined by (3.6) and

$$\beta := (h^2(0) + ||h||_{L_2}^2)^{1/2}\, |\hat{h}(\pi/\alpha)|^{-1} \,.$$

Note that $\gamma \approx 2m$ for "uniformly distributed" nodes w_j.

Proof. 1. First, we estimate the spectral norms of $\mathbf{D}$ and $\mathbf{B}$ ([13, p. 120]). By assumption and by (5.11), we see immediately that

$$\|\mathbf{D}\|_2 = \max_{k=-N/2,\dots,N/2-1} |M\, c_k(\varphi)|^{-1} = (M\,|c_{N/2}(\varphi)|)^{-1} = |\hat{h}(\pi/\alpha)|^{-1}. \quad (5.12)$$

Since ψ is even and monotonically decreasing in $[0,\infty)$, it is easy to check the integral estimate

$$\frac{1}{M} \sum_{l=-M/2}^{M/2-1} \psi^2(w_j - \frac{l}{M}) \le \frac{1}{M}\,\psi^2(0) + \int_{-\frac{1}{2}}^{\frac{1}{2}} \psi^2(x)\,\mathrm{d}x\,.$$

Then it follows by definition of ψ that

$$\begin{aligned} \sum_{l=-M/2}^{M/2-1} \psi^2(w_j - \frac{l}{M}) &\le h^2(0) + M \int_{-m/M}^{m/M} h^2(Mx)\,\mathrm{d}x \\ &\le h^2(0) + ||h||_{L_2}^2\,. \end{aligned} \quad (5.13)$$

By definition (5.10) of the sparse matrix

$$\mathbf{B} = (b_{j,k})_{j=-N/2,k=-M/2}^{N/2-1,M/2-1}\,, \quad b_{j,k} := \psi(w_j - \frac{k}{M})\,,$$

we get for the j–th component $(\mathbf{By})_j$ of $\mathbf{By}$ ($\mathbf{y} = (y_k)_{k=-M/2}^{M/2-1} \in \mathbb{C}^M$) that

$$\begin{aligned} |(\mathbf{By})_j|^2 &\le \left(\sum_{r=1}^{2m} |b_{j,k_r}|\,|y_{k_r}| \right)^2 \quad (b_{j,k_r} > 0,\ k_r \in \{-M/2,\dots,M/2-1\}) \\ &\le \left(\sum_{r=1}^{2m} b_{j,k_r}^2 \right) \left(\sum_{r=1}^{2m} |y_{k_r}|^2 \right). \end{aligned}$$

By (5.13), we have

$$\sum_{r=1}^{2m} b_{j,k_r}^2 \le \sum_{k=-M/2}^{M/2-1} \psi(w_j - \frac{k}{M})^2 \le h^2(0) + ||h||_{L_2}^2$$

such that

$$|(\mathbf{By})_j|^2 \le (h^2(0) + ||h||_{L_2}^2) \sum_{r=1}^{2m} |y_{k_r}|^2\,. \quad (5.14)$$

By assumption, each "window" $[-\frac{m}{M}+\frac{l}{M}, \frac{m}{M}+\frac{l}{M})$ $(l = -M/2, \dots, M/2-1)$ contains at most γ/α nodes $w_j, w_j \pm 1$. Therefore each column of $\mathbf{B}$ contains at most γ/α nonzero entries such that by (5.14)

$$\|\mathbf{B}\mathbf{y}\|_2^2 = \sum_{j=-N/2}^{N/2-1} |(\mathbf{B}\mathbf{y})_j|^2 \le \frac{\gamma}{\alpha} \left(h^2(0) + ||h||_{L_2}^2\right) \|\mathbf{y}\|_2^2$$

and consequently

$$\|\mathbf{B}\|_2 \le \sqrt{\frac{\gamma}{\alpha}} \left(h^2(0) + ||h||_{L_2}^2\right)^{1/2} =: \tilde{\beta} \,. \tag{5.15}$$

2. Next, it is easy to check that by (2.2) and (5.12)

$$\|\mathrm{fl}(\mathbf{D}\mathbf{f}) - \mathbf{D}\mathbf{f}\|_2 \le u \, |\hat{h}(\pi/\alpha)|^{-1} \, \|\mathbf{f}\|_2 \,. \tag{5.16}$$

Now from (5.12) it follows that

$$\|\mathrm{fl}(\mathbf{D}\mathbf{f})\|_2 \le \|\mathrm{fl}(\mathbf{D}\mathbf{f}) - \mathbf{D}\mathbf{f}\|_2 + \|\mathbf{D}\mathbf{f}\|_2 \le |\hat{h}(\pi/\alpha)|^{-1}(u+1) \, \|\mathbf{f}\|_2 \,. \tag{5.17}$$

3. Set $\hat{\mathbf{y}} := \mathrm{fl}(\sqrt{M}\, \tilde{\mathbf{F}}_M(\mathrm{fl}(\mathbf{D}\mathbf{f}))$ and $\mathbf{y} := \sqrt{M}\, \tilde{\mathbf{F}}_M \mathbf{D}\mathbf{f}$. Then we can estimate

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_2 \le \|\hat{\mathbf{y}} - \sqrt{M}\, \tilde{\mathbf{F}}_M(\mathrm{fl}(\mathbf{D}\mathbf{f}))\|_2 + \|\sqrt{M}\, \tilde{\mathbf{F}}_M(\mathrm{fl}(\mathbf{D}\mathbf{f})) - \sqrt{M}\, \tilde{\mathbf{F}}_M \mathbf{D}\mathbf{f}\|_2$$

such that by Theorem 3.3, (5.16) and (5.17)

$$\begin{aligned} \|\hat{\mathbf{y}} - \mathbf{y}\|_2 &\le \left(\sqrt{M}\,(k_M - 1)\, u + \mathcal{O}(u^2)\right) \|\mathrm{fl}(\mathbf{D}\mathbf{f})\|_2 \\ &\quad + \sqrt{M}\, \|\mathrm{fl}(\mathbf{D}\mathbf{f}) - \mathbf{D}\mathbf{f}\|_2 \\ &\le |\hat{h}(\pi/\alpha)|^{-1} \left(\sqrt{M}\, k_M \, u + \mathcal{O}(u^2)\right) \|\mathbf{f}\|_2. \end{aligned} \tag{5.18}$$

4. Finally, we consider the error between $\mathrm{fl}(\tilde{\mathbf{f}}) := \mathrm{fl}(\mathbf{B}\hat{\mathbf{y}})$ and $\tilde{\mathbf{f}} := \mathbf{B}\mathbf{y}$. By (5.15) and (5.18), we obtain

$$\begin{aligned} \|\mathrm{fl}(\tilde{\mathbf{f}}) - \tilde{\mathbf{f}}\|_2 &\le \|\mathrm{fl}(\mathbf{B}\hat{\mathbf{y}}) - \mathbf{B}\hat{\mathbf{y}}\|_2 + \|\mathbf{B}(\hat{\mathbf{y}} - \mathbf{y})\|_2 \\ &\le \|\mathrm{fl}(\mathbf{B}\hat{\mathbf{y}}) - \mathbf{B}\hat{\mathbf{y}}\|_2 + \tilde{\beta}\, |\hat{h}(\pi/\alpha)|^{-1} \left(\sqrt{M}\, k_M \, u + \mathcal{O}(u^2)\right) \|\mathbf{f}\|_2 \end{aligned} \tag{5.19}$$

By (2.1), (2.2) and (5.10), it follows from [13, p. 76], that

$$|\mathrm{fl}(\mathbf{B}\hat{\mathbf{y}}) - \mathbf{B}\hat{\mathbf{y}}| \le \frac{2mu}{1-2mu} \, \mathbf{B}\, |\hat{\mathbf{y}}|$$

and consequently by (5.15) that

$$\begin{aligned} \|\mathrm{fl}(\mathbf{B}\hat{\mathbf{y}}) - \mathbf{B}\hat{\mathbf{y}}\|_2 &\le \frac{2mu}{1-2mu} \, \|\mathbf{B}\|_2 \, \|\hat{\mathbf{y}}\|_2 \\ &\le \left(2m\tilde{\beta}\, u + \mathcal{O}(u^2)\right) \|\hat{\mathbf{y}}\|_2 \,. \end{aligned}$$

By (5.17) and (5.12), we obtain

$$\begin{aligned} \|\hat{\mathbf{y}}\|_2 &\leq \|\hat{\mathbf{y}} - \mathbf{y}\|_2 + \|\mathbf{y}\|_2 = \|\sqrt{M}\tilde{\mathbf{F}}_M \mathbf{D}\mathbf{f}\|_2 + \mathcal{O}(u)\, \|\mathbf{f}\|_2 \\ &\leq \left(\sqrt{M}\, |\hat{h}(\pi/\alpha)|^{-1} + \mathcal{O}(u)\right) \|\mathbf{f}\|_2 . \end{aligned}$$

Thus

$$\|\mathrm{fl}(\mathbf{B}\hat{\mathbf{y}}) - \mathbf{B}\hat{\mathbf{y}}\|_2 \leq \left(2m\, \tilde{\beta}\, \sqrt{M}\, |\hat{h}(\pi/\alpha)|^{-1}\, u + \mathcal{O}(u^2)\right) \|\mathbf{f}\|_{\mathbf{2}} . \tag{5.20}$$

Together with (5.19) this yields the assertion. ∎

Finally, we confirm our theoretical results by numerical experiments. Let $N = 2^n$. As example we consider the computation of

$$f(w_j) = \sum_{k=0}^{2^n-1} \mathrm{e}^{-2\pi \mathrm{i} k w_j} \quad (j = -2^{n-1}, \ldots, 2^{n-1} - 1) \tag{5.21}$$

with uniformly distributed random nodes $w_j \in \left[-\frac{1}{2}, \frac{1}{2}\right)$.
The exact vector $\hat{\mathbf{f}} = (f(w_j))_{j=-2^{n-1}}^{2^{n-1}-1}$ is given by

$$f(w_j) := \frac{\mathrm{e}^{-2\pi \mathrm{i} w_j (2^n-1)} - \mathrm{e}^{2\pi \mathrm{i} w_j}}{1 - \mathrm{e}^{2\pi \mathrm{i} w_j}} .$$

Further, let $\tilde{\mathbf{f}}_{\mathrm{C}} \in \mathbb{C}^{2^n}$ denote the vector, which was evaluated by cascade summation of the right–hand side of (5.21), and let

$$E_{\mathrm{C}}(n) := \log_{10}(\|\tilde{\mathbf{f}}_{\mathrm{C}} - \hat{\mathbf{f}}\|_2)/\|\hat{\mathbf{f}}\|_2) .$$

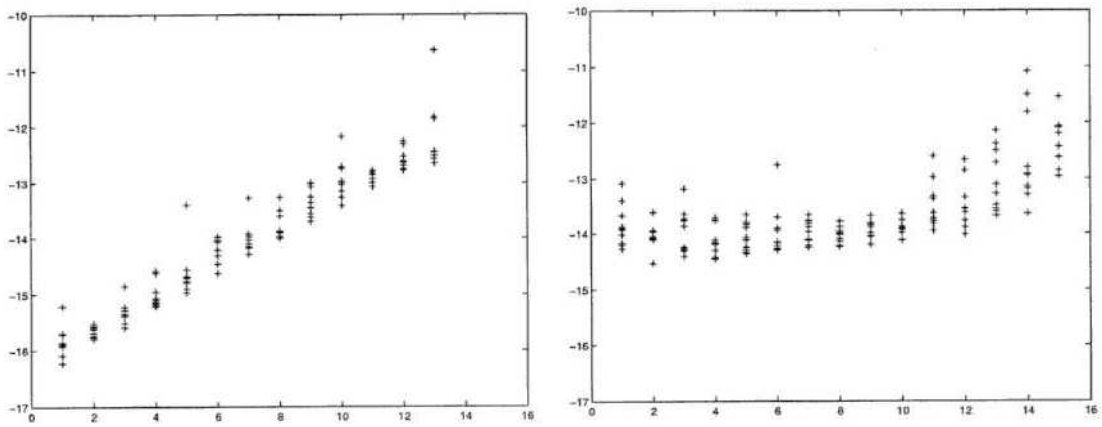

Figure 3. Left: $(n, E_{\mathrm{C}}(n))$ for $n = 1, \ldots, 13$. Right: $(n, E_{\mathrm{NDFT}}(n))$ for $n = 1, \ldots, 15$.

Figure 3 (left) shows the error $E_{\mathrm{C}}(n)$ for 10 numerical tests with various random nodes w_j as function of $n = \log_2 N$. For comparison, Figure 3 (right) presents the corresponding error

$$E_{\mathrm{NDFT}}(n) := \log_{10}(\|\tilde{\mathbf{f}} - \hat{\mathbf{f}}\|_2)/\|\hat{\mathbf{f}}\|_2)$$

introduced by Algorithm 5.1, where φ and ψ are defined by (5.7) and (5.9), respectively with $\alpha := 2$, $m := 15$ and $b := 20/\pi$. The algorithms were implemented in C and tested on a Sun SPARCstation 20 in double precision.

Acknowledgement

The authors wish to thank G. Baszenski for numerical experiments in Section 3.

References

[1] M. Arioli, H. Munthe–Kaas, and L. Valdettaro, Componentwise error analysis for FFTs with applications to fast Helmholtz solvers, *Numer. Algorithms* 12,

65 – 88 (1996).

[2] G. Baszenski, U. Schreiber, and M. Tasche, Numerical stability of fast cosine transforms, *Numer. Funct. Anal. Optim.* 21, 25 – 46 (2000).

[3] G. Baszenski and M. Tasche, Fast polynomial multiplication and convolution related to the discrete cosine transform, *Linear Algebra Appl.* 252, 1 – 25 (1997).

[4] G. Beylkin, On the fast Fourier transform of functions with singularities, *Appl. Comput. Harmon. Anal.* 2, 363 – 381 (1995).

[5] W. L. Briggs and V. E. Henson, *The* DFT, SIAM, Philadelphia, 1995.

[6] O. Buneman, Stable on–line creation of sines and cosines of successive angles, *Proc.* IEEE 75, 1434 – 1435 (1987).

[7] C. Y. Chu, The fast Fourier transform on the hypercube parallel computers, PhD thesis, Cornell University, Ithaca, 1988.

[8] P. Deuflhard and A. Hohmann, *Numerische Mathematik*, de Gruyter, Berlin, 1991.

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

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

[11] G. Heinig and K. Rost, DFT representations of Toeplitz–plus–Hankel Bezoutians with application to fast matrix–vector multiplication, *Linear Algebra Appl.* 284, 157 – 176 (1998).

[12] P. Henrici, *Applied and Computational Complex Analysis*, Vol. 3, J. Wiley & Sons, New York, 1993.

[13] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, SIAM, Philadephia, 1996.

[14] J. I. Jackson, Selection of a convolution function for Fourier inversion using gridding, IEEE *Trans. Medical Imaging* 10, 473 – 478, 1991.

[15] C. Van Loan, *Computational Frameworks for the Fast Fourier Transform*, SIAM, Philadelphia, 1992.

[16] J. Pelt, Fast computation of trigonometric sums with applications to frequency analysis of astronomical data, in *Astronomical Time Series* (D. Maoz, A. Sternberg, and E. M. Leibowitz, eds.), Kluwer, Dordrecht, 1997, pp. 179 – 182.

[17] D. Potts and G. Steidl, Preconditioners for ill–conditioned Toeplitz matrices, *BIT* 39, 513 – 533 (1999).

[18] G. U. Ramos, Roundoff error analysis of the fast Fourier transform. *Math. Comp.* 25, 757 – 768 (1971).

[19] K. R. Rao and P. Yip, *Discrete Cosine Transforms*, Academic Press, Boston, 1990.

[20] J. C. Schatzman, Accuracy of the discrete Fourier transform and the fast Fourier transform, SIAM *J. Sci. Comput.* 17, 1150 – 1166 (1996).

[21] G. Steidl, Fast radix–p discrete cosine transform, *Appl. Algebra Engrg. Comm. Comput.* 3, 39 – 46 (1992).

[22] G. Steidl, A note on fast Fourier transforms for nonequispaced grids, *Adv. Comput. Math.* 9, 337 – 353 (1998).

[23] G. Steidl and M. Tasche, A polynomial approach to fast algorithms for discrete Fourier–cosine and Fourier–sine transforms, *Math. Comp.* 56, 281 – 296 (1991).

[24] M. Tasche and H. Zeuner, Roundoff error analysis for fast trigonometric transforms, in *Handbook of Analytic–Computational Methods in Applied Mathematics* (G. A. Anastassiou, ed.), CRC Press, Boca Raton, 2000, pp. 357 – 406.

[25] M. Tasche and H. Zeuner, Worst and average case roundoff error analysis for FFT, *BIT* 41, 563 – 581 (2001).

[26] M. Tasche and H. Zeuner, Improved roundoff error analysis for precomputed twiddle factors, *J. Comput. Anal. Appl.* 4, 1 – 18 (2002).

[27] Z. Wang, Fast algorithms for the discrete W transform and for the discrete-Fourier transform, IEEE *Trans. Acoust. Speech Signal Process.* 32, 803 – 816 (1984).

[28] P. Y. Yalamov, Improvements of some bounds on the stability of fast Helmholtz solvers, *Numer. Algorithms* 26, 11 – 20 (2001).