Non-uniform Fast Fourier Transform of Type 1

David Krantz, mail@dkrantz.se

April 22, 2020

1 Introduction

The Fourier transform, and the Fast Fourier Transform (FFT) algorithm, have been around for many years and have proved their usefulness in many fields of science, e.g. in signal processing. The FFT however depends on the data being equidistant, which in many applications might not be the case. Real world data can consist of outliers and missing data points, resulting in non-uniform data. A fast algorithm for spectral analysis of non-uniform data is therefore of interest.

We now describe an algorithm that computes the non-uniform fast Fourier transform (NUFFT) of type 1. The main idea of the algorithm is to make the data uniform by means of interpolation and then make use of the standard FFT. The theory described below is almost completely based on the work done by Greengard and Lee (2004) [1].

2 The Non-uniform Fast Fourier Transform of Type 1

The NUFFT can be described in several dimensions, but for simplicity reasons only the 1-dimensional case will be considered in this report. Consider N non-uniform data points $x_j \in [0, 2\pi]$, $j = 0, 1, \ldots, N-1$, with corresponding strengths $f_j \in \mathbb{C}$, and let M denote the number of Fourier modes of interest. Then, the type 1 NUFFT is defined as

$$F(k) = \frac{1}{N} \sum_{j=0}^{N-1} f_j e^{-ikx_j}, \tag{1}$$

for the frequencies $k = -M/2, -M/2+1, \ldots, M/2-1$.

2.1 The Algorithm

The Fourier transform convention used in this section is

$$\begin{aligned} \Phi(k) &= \frac{1}{2\pi} \int_{-\infty}^{\infty} \phi(x) e^{-ikx} dx \\ \phi(x) &= \int_{-\infty}^{\infty} \Phi(k) e^{ikx} dx, \end{aligned} \tag{2}$$

where $\Phi(k)$ is the Fourier transform of $\phi(x)$.

Eq. (1) can be viewed as computing the Fourier series coefficients of the function

$$f(x) = \sum_{j=0}^{N-1} f_j \delta(x - x_j), \tag{3}$$

defined periodically on $[0, 2\pi]$, where $\delta(x)$ denotes the Dirac-delta function. Note that $f(x)$ is a continuous function, which we could sample on an uniform grid and then apply standard FFT to. However, since $f(x)$ is not well-resolved on an uniform grid the result would most likely not be satisfactory. Thus, we want to find a function that describes $f(x)$ well on an uniform grid and perform FFT on that function instead. One way of doing this is to interpolate the strengths f_j to nearby points on the uniform grid. This "spreading" is in this case achieved by convolving f with the 1-dimensional periodic Gaussian kernel g_τ as

$$f_\tau(x) = f * g_\tau(x), \tag{4}$$

where

$$g_\tau(x) = \sum_{l=-\infty}^{\infty} e^{-(x-2l\pi)^2/(4\tau)}. \tag{5}$$

The parameter $\tau \in \mathbb{R}$ controls the width of the kernel, which if chosen properly will result in f_τ being well-resolved on an oversampled uniform grid.

Define $R > 1$ as the upsampling factor and let $M_r = RN$ denote the number of points on the oversampled uniform grid $\xi \in [0, 2\pi]$ with the grid spacing $h = 2\pi/M_r$ and grid points $\xi_m = hm$, $m = 0, 1, \ldots, M_r - 1$. The Fourier coefficients of Eq. (4) can then be approximated using the standard FFT as

$$F_\tau(k) \approx \frac{1}{M_r} \sum_{m=0}^{M_r-1} f_\tau(\xi_m) e^{-ik\xi_m}, \tag{6}$$

where

$$f_\tau(\xi_m) = \sum_{j=0}^{N-1} f_j g_\tau(\xi_m - x_j). \tag{7}$$

Note that the approximation introduced in Eq. (6) originates from the fact that we are approximating the integral that exactly computes the Fourier coefficients of f_τ using a finite sum.

Taking the Fourier transform of Eq. (4) gives

$$F_\tau(k) = F(k) G_\tau(k), \tag{8}$$

where $G_\tau(k) = \sqrt{\tau/\pi} e^{-k^2\tau}$ denotes the Fourier transform of g_τ. From this we can simply compute the wanted $F(k)$ as

$$F(k) = \sqrt{\frac{\pi}{\tau}} e^{k^2\tau} F_\tau(k). \tag{9}$$

Worth mentioning is that in practice g_τ is not evaluated at all points on the grid. This is because the Gaussian centred at x_j can be seen as non-negligible only at nearby grid points due to its sharpness and width, which are tuned by τ. We denote the number of non-negliable neighbouring points in each direction of x_j as the spreading parameter M_{sp}.

A summarized procedure of this section can be found in Alg. 1.

Algorithm 1.

1. Construct an oversampled grid $\xi \in [0, 2\pi]$ with step size $h = 2\pi/M_r$
2. Find the points $\tilde{\xi}_j = hm_j$ in ξ that are the closest to x_j for $j = 0, 1, \ldots, N-1$, where the distance is measured by the absolute value
3. Add $f_j e^{-(x_j - \tilde{x}_j - hl)^2/(4\tau)}$ to $f_\tau(m_j + l)$ for $j = 0, 1, \ldots, N-1$ and $-M_{sp} \leq l \leq M_{sp}$
4. Calculate F_τ using the standard FFT (of size M_r) of f_τ
5. Set $F(k) = \sqrt{\pi/\tau} e^{k^2\tau} F_\tau(k)$ for $M/2 \leq k < M/2$

3 Result and Discussion

The three methods used to compute Eq. (1) discussed in this section are the previously described NUFFT algorithm, FFT and direct summation (DS). We will present validations of the NUFFT algorithm and the implementation as well as discussing its accuracy and the parameters controlling it. The NUFFT algorithm was implemented in MATLAB where the built-in function `fft` was used to compute the standard FFT.

We measure the errors r of the different methods using the following relative L_2 norm

$$r = \frac{\|\hat{F} - F\|_2}{\|F\|_2}, \tag{10}$$

where F is the result of DS and $\hat{F}$ the result of the computations from NUFFT or FFT.

3.1 Uniform Data

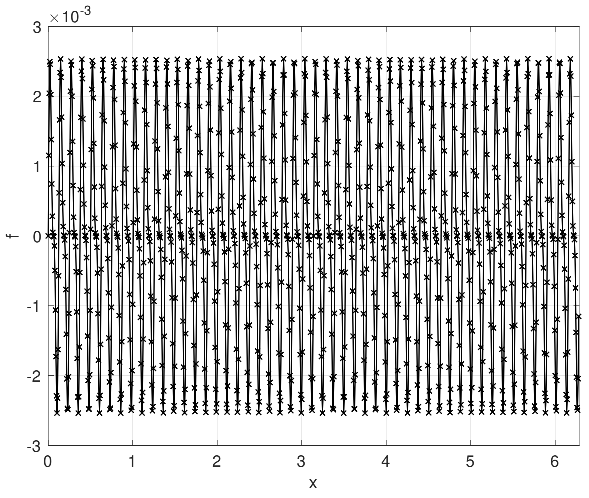
In order to validate the implementation we begin with considering uniform data and comparing the results with performing the direct summation and standard FFT. A signal consisting of a sum of two sine waves with frequencies 50 Hz and 100 Hz and amplitudes 2 and 1 respectively was constructed. We expect the Fourier transform of this signal to be zero for all frequencies except the two ones in the signal. We also expect the magnitude of the peak at 50 Hz to be two times the one at 100 Hz. We set $N = M = 2^{10}$, $R = 8$, $M_{sp} = 24$, and alike Greengard and Lee (2004) [1], set τ according to

$$\tau = \frac{1}{M^2} \frac{\pi}{R(R-0.5)} M_{sp}. \tag{11}$$

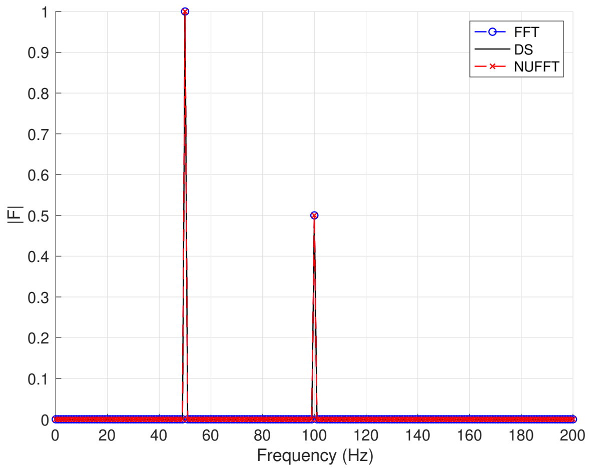
Fig. 1 shows the generated signal and single-sided spectrum of the its Fourier transform. Firstly, we see that all three methods seem to overlap each other. Secondly, as expected, there are two peaks at the frequencies 50 Hz and 100 Hz with the predicted magnitudes. The Fourier coefficients at the other frequencies are less than 10^{-14} and are thus considered to be zero. The errors and computational times is shown in Tab. 1. We can with certainty validate the NUFFT algorithm and its implementation when using uniform data as it has an error comparable with the one of FFT. Worth mentioning is that the achieved accuracy comes with the price of it being relatively computationally expensive. More specifically, it is about 300 times slower than the standard FFT algorithm but, in this particular case, approximately twice as fast as the DS method. The only conclusion we can draw from this single observation is that the NUFFT implementation is significantly slower than the FFT function implemented in MATLAB.

Table 1: *Errors of the methods according to Eq.* (10) *as well as the time it took them to evaluate Eq.* (1).

Method	NUFFT	FFT	DS
r	$7.65 \cdot 10^{-14}$	$7.63 \cdot 10^{-14}$	0
Elapsed time (s)	0.12	0.00038	0.28



(a)



(b)

Figure 1: *Single-sided discrete Fourier transform (1b) of the signal in 1a computed using NUFFT (dashed red line with crosses), DS (solid black line) and FFT (dashed blue line with circles).*

3.2 Non-uniform Data

We now consider the same parameters and data as in Sect. 3.1 but sampled at random points in $[0, 2\pi]$. More specifically, we generate $N/2$ samples in $[0, \pi]$ and $N/2$ samples in $[\pi + 1, 2\pi]$ from an uniform distribution. This will make the data highly non-uniform as well as having a large gap in the middle, see Fig. 2a. Since the only difference between this setup and the one in Sect. 3.1 is that the data points now are non-uniform, we expect the result to have two peaks as in Fig. 1b. However, since the sine functions are sampled at randomly distributed points, additional frequencies can be introduced in the data through the effect of aliasing. We do however expect the power of these frequencies to be small compared to the ones at 50 Hz and 100 Hz. We therefore also expect the result to have relatively small, but non-zero, Fourier coefficients spread across the spectrum.

The generated signal and its Fourier transform is shown in Fig. 2. Note that the result from the FFT algorithm is not included due to it not supporting non-uniform data. As predicted, the result is noisy but has two relatively distinct peaks at 50 Hz and 100 Hz. The relative L_2 error between DS and NUFFT read $r = 6.20 \cdot 10^{-14}$. Comparing this error with its equivalent in Tab. 1, we can conclude the NUFFT algorithm handles both uniform and non-uniform data well.

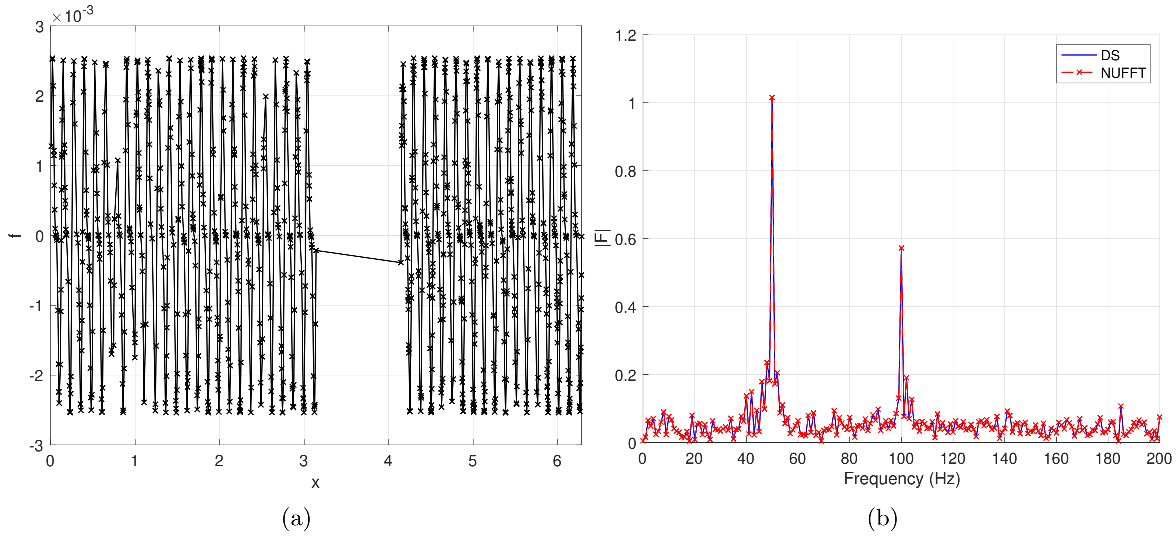


Figure 2: *Single-sided discrete Fourier transform (2b) of the signal in 2a computed using NUFFT (dashed red line with crosses) and DS (solid black line).*

3.3 Error Controlling Parameters

The accuracy of the NUFFT algorithm is controlled by the Gaussian kernel parameter τ, oversampling factor R and spreading parameter M_{sp}. More specifically, they affect the quality of the interpolation of the data from a non-uniform to an uniform grid. We now present two results highlighting the role of these parameters, as well as how they relate to each other.

3.3.1 Spreading Parameter M_{sp} and Oversampling Factor R

For a given data point, the spreading parameter M_{sp} controls the number of adjacent points the Gaussian kernel g_τ should affect. Since the goal is to interpolate each $f(x_j)$ onto the uniform oversampled grid, we expect that larger values of M_{sp} will improve the accuracy due to it will result in the values $f(x_j)$ being spread onto points further away. To accurately perform this spreading we want to centre g_τ as close to each x_j as possible. The precision of which we may perform this is furthermore tuned by the upsampling factor R. Since it controls the number of data points in the fine grid, a larger R results in a smaller step size h. The effect of this is that we are able to perform step 2 in Alg. 1 better. In summary, we expect the accuracy of the NUFFT algorithm to improve as M_{sp} and R are increased.

We now consider non-uniform data points as constructed in Sect. 3.2 and values $f(x_j)$ generated from an uniform distribution on the open interval $(0, 1)$. We set $N = M = 2^{10}$, τ according to Eq. (11), and sweep M_{sp} from 1 to 24 and R from 2 to 8.

The result from this experiment is shown in Fig. 3. There are a few interesting things to be said about this result. Firstly, we see a clear trend that larger M_{sp} yields lower error. This result is well in line with the expectations. Secondly, the error seems to plateau around 10^{-13}. Moreover, the larger R, the smaller M_{sp} we need to reach the plateau. Fig. 3 also reveals that the error plateaus at smaller values of r for larger R, which also was expected. Another interesting observation is the slope of the lines with different values of R. The orange dashed line with $R = 3$ has a slope of around -1 (see black solid line for reference). By increasing R we see that the slope becomes steeper as it approaches the solid magenta line of slope -2. This tells us that the algorithm convergences faster when increasing M_{sp} for larger values of R.

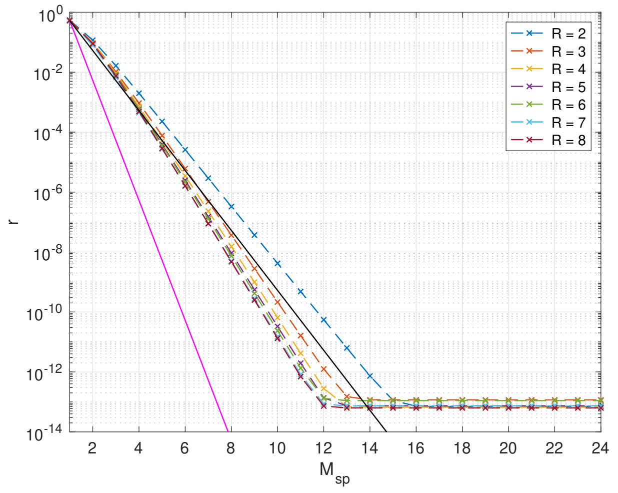


Figure 3: *Relative error r for different values of M_{sp} and R.* The solid black and magenta lines are two reference lines with slope -1 and -2 respectively.

One explanation why the error plateaus could be due to the set values of M_{sp}, R and τ. If $R = 8$ the curve plateaus around $M_{sp} = 12$. This gives $M_r = 2^{13}$ and $h = 7.7 \cdot 10^{-4}$. This means that all points within $M_{sp}h = 0.0092$ of each point is considered as non-negligible when convolving f with g_τ. We now put this distance into the context of the full width half maximum (FWHM) of g_τ. It turns out that we have $\text{FWHM} = 4\sqrt{\ln(2) \cdot \tau} = 4\sqrt{\ln(2) \cdot 6.0 \cdot 10^{-7}} = 0.0026$, resulting in $M_{sp}h \approx 3.5 \cdot \text{FWHM}$. One could thus imagine that increasing M_{sp} even more would not yield any significant change in the result.

However, a more probable explanation is that the plateau is due to the FFT algorithm. Since it is used in the NUFFT algorithm, the accuracy of the NUFFT algorithm is limited by the on in the FFT. This argument is strengthened by the fact that the order of the error of the NUFFT and FFT method found in Tab. 1 agrees well with the error level on which the NUFFT method plateaus.

3.3.2 Gaussian Kernel Parameter τ

We now study how the choice of τ affects the error of the algorithm by sweeping it from being around 10^{-7} to 10^{-5}. Consider the same data as in Sect. 3.3.1 and set $R = 4$ and $M_{sp} = 12$.

The result from the sweep of τ is shown in Fig. 4. The error decreases rapidly in the beginning and reaches its minimum of approximately $r = 6.1 \cdot 10^{-14}$ at $\tau = 2.34 \cdot 10^{-6}$. From this point, the error keeps increasing with τ until it eventually will be infinite. Evidently, there is a relatively small interval in which τ should be chosen. Worth mentioning is that τ computed according to Eq. (11), as illustrated by the blue solid vertical line in Fig. 4, is close to the values that achieved the lowest error in this sweep.

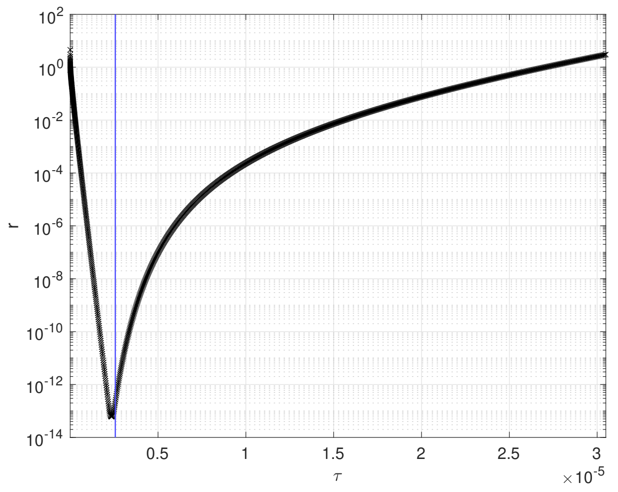


Figure 4: *Relative error r for different values of τ. The vertical blue solid line is where τ is computed according to Eq.* (11)*.*

The behaviour in Fig. 4 can partly be explained by how τ relates to the width of the Gaussian g_τ. If τ become small, it will yield a narrow Gaussian, resulting in the values $f(x_j)$ not being spread far. This makes f_τ a bad representation of f the uniform grid, and hence the error grows. In contrast, if τ become large, the values $f(x_j)$ will be spread far. The outcome of this is however that we lose the resolution of the signal, and thus the error will grow. The width of g_τ, i.e. the parameter τ, should thus, in relation to $M_{sp}h$, not be chosen too large and not too small.

4 Conclusion

In conclusion, we have validated the NUFFT algorithm and the implementation of it by being able to achieve an accuracy comparable with the one of FFT ($r \approx 10^{-14}$). The accuracy of the NUFFT algorithm is controlled by the three parameters M_{sp}, R and τ. We can conclude that the larger values of M_{sp} and R, the higher the accuracy of the algorithm. However, it simultaneously becomes more computationally expensive as it will require more memory and additional computations. Thus, one should choose these parameters based on the needed accuracy. τ controls the shape of the Gaussian kernel g_τ and should be chosen such that the Gaussian, in relation to M_{sp} and R, does not become to narrow and not to wide. Finally, we have numerically validated that the choice of τ used by Greengard and Lee (2004) [1] in Eq. (11) is an appropriate choice.

References

[1] Leslie Greengard and June-Yub Lee. "Accelerating the Nonuniform Fast Fourier Transform". In: *SIAM Review* 46.3 (2004), pp. 443–454. DOI: `10.1137/S003614450343200X`. eprint: `https://doi.org/10.1137/S003614450343200X`. URL: `https://doi.org/10.1137/S003614450343200X`.