Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at *PeerJ Computer Science* **2020**; doi:10.7717/peerj-cs.257

Discrete Two Dimensional Fourier Transform in Polar Coordinates Part II: Numerical Computation and Approximation of the Continuous Transform

Xueyang Yao[1], Natalie Baddour[2]

[1]Department of Systems Design, University of Waterloo, 200 University Avenue West, Waterloo, Ontario, N2L 3G1, Canada

[2]Department of Mechanical Engineering, University of Ottawa, 161 Louis Pasteur, Ottawa, Ontario, K1N 6N5, Canada

[1]xueyang.yao@uwaterloo.ca

[2]nbaddour@uottawa.ca (corresponding author)

Abstract

The theory of the continuous two-dimensional (2D) Fourier Transform in polar coordinates has been recently developed but no discrete counterpart exists to date. In the first part of this two-paper series, we proposed and evaluated the theory of the 2D discrete Fourier Transform (DFT) in polar coordinates. The theory of the actual manipulated quantities was shown, including the standard set of shift, modulation, multiplication, and convolution rules. In this second part of the series, we address the computational aspects of the 2D DFT in polar coordinates. Specifically, we demonstrate how the decomposition of the 2D DFT as a DFT, Discrete Hankel Transform (DHT) and inverse DFT sequence can be exploited for efficient code. We also demonstrate how the proposed 2D DFT can be used to approximate the continuous forward and inverse Fourier transform in polar coordinates in the same manner that the 1D DFT can be used to approximate its continuous counterpart.

Keywords: Fourier Theory, DFT in polar coordinates, polar coordinates, multidimensional DFT, Discrete Hankel Transform, Discrete Fourier Transform, Orthogonality.

1 Introduction

The Fourier transform is a powerful analytical tool and has proved to be invaluable in many disciplines such as physics, mathematics and engineering. The development of the Fast Fourier Transform (FFT) algorithm [1], which computes the discrete Fourier transform with a fast algorithm, firmly established the Fourier transform as a practical tool in diverse areas, most notably signal and image processing.

In two dimensions, the FFT can still be used to compute the discrete Fourier transform in Cartesian coordinates. However, in many applications such as photoacoustics [2] and tomography ([3], [4], [5]), it is often necessary to compute the Fourier transform in polar coordinates. Moreover, for functions that are naturally described in polar coordinates, a discrete version of the 2D Fourier transform in polar coordinates is needed. There have been some attempts to calculate the Fourier transform in polar coordinates, most notably through the Hankel transform, since the zeroth order Hankel transform is known to be a 2D Fourier transform in polar coordinates for rotationally symmetric functions. However, prior work has focused on

© 2019 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

numerically approximating the continuous transform. This stands in contrast to the Fourier transform, where the Discrete Fourier Transform (DFT) can stand alone as an orthogonal transform, independent of the existence of its continuous counterpart.

In the first part of this two-part paper series, we proposed an independent discrete 2D Fourier transform in polar coordinates, which has been defined to be discrete from first principles [6]. Standard operational 'rules' of shift, modulation and convolution rules for this 2D DFT in polar coordinates were also demonstrated. The operational rules were demonstrated via the key properties of the proposed discrete kernel of the transform. However, using the discrete kernel may not be the most effective way to compute the transform. Furthermore, while the 2D DFT in polar coordinates was demonstrated to have properties and rules as a standalone transform independent of its relationship to any continuous transform, an obvious application of the proposed discrete transform is to approximate its continuous counterpart.

Hence, the goal of this second part of this two-part paper series is to propose computationally efficient approaches to the computation of the previously proposed 2D DFT in polar coordinates and also to validate its effectiveness to approximate the continuous 2D Fourier transform in polar coordinates. Moreover, since computing speed is one of the most important criteria for a discrete transform, computational improvements to the transform are also proposed.

The outline of the paper is as follows. Section 2 introduces the proposed definition of the discrete 2D Fourier transform in polar coordinates. The motivation of this definition and the transform rules (multiplication, convolution, shift etc) are given in the first part of this two-part paper. Section 3 introduces the discretization scheme where we show the connection between discrete samples of the continuous functions and the discrete transform. Here, the connection between using the proposed 2D DFT and sampled vales of the continuous functions is explained. The proposed 2D DFT relies on a specific sampling scheme (introduced in section 3) which can be plotted and analyzed for 'grid coverage' – how much of the 2D plane is covered and at which density. Thus, Section 4 analyzes the proposed discretization points and their implication of the sampling grid for density and coverage of the grid. The insights gained from this section will be useful in interpreting the results of approximating the continuous transform with the discrete transform. Section 5 introduces numerical computation schemes whereby the interpretation of the proposed 2D transform as a sequence of 1D DFT, 1D DHT and 1D IDFT is exploited. Approaches to exploit some of Matlab's (Mathworks 2018) built-in functions for ease of coding and computational speed are also demonstrated. Section 6 then investigates the ability of the proposed 2D DFT to approximate the continuous transform in terms of precision and accuracy. Three test functions for which closed-form continuous transforms are known are analyzed. Section 7 discusses the computational time of the transform and approaches that were successfully undertaken to drastically improve the computation time. Finally, section 8 summarizes and concludes the paper. Sample Matlab code is included in the appendix of the paper.

2 Definition of the Discrete 2D Fourier Transform in Polar Coordinates

The 2D-Discrete Fourier Transform in polar coordinates has been defined in the first part of this two-paper series as the discrete transform that takes the matrix (or double-subscripted series) f_{pk} to the matrix (double-subscripted series) F_{ql} such that $f_{pk} \to F_{qm}$ is given by

$$F_{qm} = \mathbb{F}\left(f_{pk}\right) = \sum_{k=1}^{N_1-1} \sum_{p=-M}^{M} f_{pk} E^{-}_{qm;pk} \qquad (1)$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

where p,k,q,m,n , N_1, and N_2 are integers such that $-M \le n \le M$, where $2M+1=N_2$ $1 \le m,k, \le N_1 - 1$ and $-M \le p,q \le M$. Similarly, for the inverse transform we propose

$$f_{pk} = \mathbb{F}^{-1}\left(F_{qm}\right) = \sum_{m=1}^{N_1-1} \sum_{q=-M}^{M} F_{qm} E^{+}_{qm;pk} \tag{2}$$

In equations (1) and (2), $E^{\pm}_{qm;pk}$ are the kernels of the transformation. These can be chosen as the 'non-symmetric' form given by

$$\begin{aligned} E^{-}_{qm;pk} &= \frac{1}{N_2} \sum_{n=-M}^{M} \frac{J_n\left(\frac{j_{nk} j_{nm}}{j_{nN_1}}\right)}{j^2_{nN_1} J^2_{n+1}\left(j_{nk}\right)} 2i^{-n} e^{-in\frac{2\pi p}{N_2}} e^{+in\frac{2\pi q}{N_2}} \\ E^{+}_{qm;pk} &= \frac{1}{N_2} \sum_{n=-M}^{M} \frac{J_n\left(\frac{j_{nm} j_{nk}}{j_{nN_1}}\right)}{J^2_{n+1}\left(j_{nm}\right)} 2i^{+n} e^{+i\frac{2\pi np}{N_2}} e^{-i\frac{2\pi nq}{N_2}} \end{aligned} \tag{3}$$

Here, $J_n(z)$ is the *n*th order Bessel function of the first kind and j_{nk} denotes the *k*th zero of the *n*th Bessel function. The subscript (+ or -) indicated the sign on the $i^{\pm}$ and on the exponent containing the *p* variable; the *q* variable exponent then takes the opposite sign. From a matrix point of view, both f_{pk} and F_{ql} are $N_2 \times \left(N_1 - 1\right)$ sized matrices. Another possible form of the kernel is the 'symmetric' form where the kernels are complex conjugates of each other. This form is simply a matter of redistributing the factors of j_{nN_1} in the denominators so that

$$\begin{aligned} E^{(s)-}_{qm;pk} &= \frac{1}{N_2} \sum_{n=-M}^{M} \frac{J_n\left(\frac{j_{nk} j_{nm}}{j_{nN_1}}\right)}{j_{nN_1} J^2_{n+1}\left(j_{nk}\right)} 2i^{-n} e^{-in\frac{2\pi p}{N_2}} e^{+in\frac{2\pi q}{N_2}} \\ E^{(s)+}_{qm;pk} &= \frac{1}{N_2} \sum_{n=-M}^{M} \frac{J_n\left(\frac{j_{nm} j_{nk}}{j_{nN_1}}\right)}{j_{nN_1} J^2_{n+1}\left(j_{nm}\right)} 2i^{n} e^{+i\frac{2\pi np}{N_2}} e^{-i\frac{2\pi nq}{N_2}} \end{aligned} \tag{4}$$

As before, p,k,q,m,n , N_1, and N_2 are integers such that $-M \le n \le M$, where $2M+1=N_2$ $1 \le m,k, \le N_1 - 1$ and $-M \le p,q \le M$. In equation (4), $E^{(s)+}_{qm;pk}$ is now the complex conjugate of $E^{(s)-}_{qm;pk}$. Both sets of kernels in equations (3) and (4) lead to the same set of modulation, shift,

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

convolution, multiplication rules. The form in equation (3) arises naturally from discretization of the continuous transform, but does not lead to the expected Parseval relationship. The form in equation (4) is less suited to approximating the continuous transform but does lead to the expected Parseval relationship. The Parseval relationships were presented and shown in the first part of this two-part paper.

3 Discrete Transform to approximate the continuous transform

In this section, relationships between discretely sampled values of the function and its continuous 2D FT are presented in the case of a space-limited or band-limited function. These relationships were derived in the first part of the paper and are repeated here since they will form the basis for the using the discrete transform to approximate the continuous transform at specified sampling points

Consider a function in the space domain $f(r,\theta)$ which is space limited to $r \in [0,R]$. This implies that the function is zero outside of the circle bounded by $r \in [0,R]$. An approximate relationship between sampled values of the continuous function and sampled values of its continuous forward 2D transform $F(\rho,\psi)$ has been derived in the first part of the two-part paper as

$$F\left(\frac{j_{qm}}{R},\frac{2\pi q}{N_2}\right) \approx 2\pi R^2 \sum_{k=1}^{N_1-1} \sum_{p=-M}^{M} f\left(\frac{j_{pk}R}{j_{pN_1}},\frac{2\pi p}{N_2}\right) \frac{1}{N_2} \sum_{n=-M}^{M} \frac{2i^{-n} J_n\left(\frac{j_{nk}\, j_{nm}}{j_{nN_1}}\right)}{j_{nN_1}^2 J_{n+1}^2(j_{nk})} e^{-i\frac{2\pi np}{N_2}} e^{+i\frac{2\pi nq}{N_2}} \quad (5)$$

Similarly, an approximate relationship between sampled values of the continuous forward transform $F(\rho,\psi)$ and sampled values of the continuous original function $f(r,\theta)$ was shown to be given by

$$f\left(\frac{j_{pk}R}{j_{pN_1}},\frac{2\pi p}{N_2}\right) \approx \frac{1}{2\pi R^2} \sum_{m=1}^{N_1-1} \sum_{q=-M}^{M} F\left(\frac{j_{qm}}{R},\frac{2\pi q}{N_2}\right) \frac{1}{N_2} \sum_{n=-M}^{M} \frac{2i^{n} J_n\left(\frac{j_{nm}\, j_{nk}}{j_{nN_1}}\right)}{J_{n+1}^2(j_{nm})} e^{+i\frac{2\pi np}{N_2}} e^{-i\frac{2\pi nq}{N_2}} \quad (6)$$

In equations (5) and (6), $f(r,\theta)$ is the original function in 2D space and $F(\rho,\psi)$ is the 2D Fourier transform of the function in polar coordinates. The values of the sampled functions given in equations and can be considered to be the discrete quantities denoted by

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

$$f_{pk} = f\left(\frac{j_{pk}R}{j_{pN1}}, \frac{2\pi p}{N_2}\right)$$
$$F_{qm} = F\left(\frac{j_{qm}}{R}, \frac{2\pi q}{N_2}\right) \quad (7)$$

Now consider functions in the frequency domain $F(\rho,\psi)$ with an effective band limit $\rho \in [0, W_\rho]$. That is, we suppose that the 2D Fourier transform $F(\rho,\psi)$ of $f(r,\theta)$ is band-limited, meaning that $F(\rho,\psi)$ is zero for $\rho \geq W_\rho = 2\pi W$. The variable W_ρ is written in this form since W would typically be quoted in units of Hz (cycles per second) if using temporal units or cycles per meter if using spatial units. Therefore, the multiplication by 2π ensures that the final units are in s^{-1} or m^{-1}. The approximate relationship between the discrete transform $F(\rho,\psi)$ and the sampled values of the continuous transform $f(r,\theta)$ was derived in the first part of the paper and is given by

$$F\left(\frac{j_{qm}W_p}{j_{qN_1}}, \frac{2\pi q}{N_2}\right) \approx \frac{2\pi}{W_\rho^2} \sum_{k=1}^{N_1-1} \sum_{p=-M}^{M} f\left(\frac{j_{pk}}{W_\rho}, \frac{2\pi p}{N_2}\right) \frac{1}{N_2} \sum_{n=-M}^{M} \frac{2i^{-n} J_n\left(\frac{j_{nm}j_{nk}}{j_{nN_1}}\right)}{J_{n+1}^2(j_{nk})} e^{-i\frac{2\pi np}{N_2}} e^{+i\frac{2\pi nq}{N_2}} \quad (8)$$

The inverse transform is given by

$$f\left(\frac{j_{pk}}{W_\rho}, \frac{2\pi p}{N_2}\right) \approx \frac{W_\rho^2}{2\pi} \sum_{m=1}^{N_1-1} \sum_{q=-M}^{M} F\left(\frac{j_{qm}W_p}{j_{qN_1}}, \frac{2\pi q}{N_2}\right) \frac{1}{N_2} \sum_{n=-M}^{M} \frac{2i^{n} J_n\left(\frac{j_{nk}j_{nm}}{j_{nN_1}}\right)}{j_{nN_1}^2 J_{n+1}^2(j_{nm})} e^{-i\frac{2\pi nq}{N_2}} e^{+i\frac{2\pi np}{N_2}} \quad (9)$$

As before, the relationships in equations (8) and (9) give relationships between the sampled values of the original function

$$f_{pk} = f\left(\frac{j_{pk}}{W_p}, \frac{2\pi p}{N_2}\right) \quad (10)$$

and sampled values of its continuous 2D transform

$$F_{qm} = F\left(\frac{j_{qm}W_\rho}{j_{qN_1}}, \frac{2\pi q}{N_2}\right) \quad (11)$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

The relationships given by equation (5),(6) ,(8) and (9), were the motivating definition of a 2D Discrete Fourier transform in polar coordinates, defined in the first part of this two-part paper. In the context of this second part of the two-part paper, they are also the relationships that permit the use of the discrete transform to approximate the continuous transform at the specified sampling points.

4 Discretization Points and Sampling Grid

The transforms defined in equations (1) and (2) can be applied to any matrix f_{pk} to yield its forward transform F_{qm}, which can then be transformed backwards by using the inverse transform. However, if these same discrete transforms are to be used for the purpose of approximating a continuous 2D Fourier transform, then these transforms need to be applied to the specific sampled values of the continuous functions in both space and frequency domains, as shown in equations (7) and (11). The relationships in (7) and (11) define the sampling points that need to be used and it is noted that the points are defined differently based on whether we start with the assumption of a space or band limited function. These specific sampling points as given in equation (7) and (11) imply a specific sampling grid for the function. In this section, the sampling grid (its coverage and density in 2D) is analyzed.

4.1 Sampling points

For a space-limited function, we assume that the original function of interest is defined over continuous (r,θ) space where $0 \le r \le R$ and $0 \le \theta \le 2\pi$. The discrete sampling spaces used for radial and angular sampling points in regular $\vec{r}$ space (r,θ) and $\vec{\omega}$ frequency (ρ,ψ) space are defined as

$$r_{pk} = \frac{j_{pk} R}{j_{pN_1}} \qquad \theta_p = \frac{p2\pi}{N_2} \tag{12}$$

and

$$\rho_{qm} = \frac{j_{qm}}{R} \qquad \psi_q = \frac{q2\pi}{N_2} \tag{13}$$

For a band limited function, the function is assume band-limited to $0 \le \rho \le W_\rho, 0 \le \psi \le 2\pi$. The sampling space used for radial and angular sampling points in regular $\vec{\omega}$ frequency space (ρ,ψ) and $\vec{r}$ space (r,θ) for a bandlimited function is defined as

$$r_{pk} = \frac{j_{pk}}{W_\rho} \qquad \theta_p = \frac{p2\pi}{N_2} \tag{14}$$

and

$$\rho_{qm} = \frac{j_{qm} W_\rho}{j_{qN_1}} \qquad \psi_q = \frac{q2\pi}{N_2} \tag{15}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

As above, p,k,q,m,n, N_1, and N_2 are integers such that $-M \le n \le M$, where $2M+1=N_2$, $1 \le m,k, \le N_1 - 1$ and $-M \le p,q \le M$. Clearly, the density of the sampling points depends on the numbers of points chosen, that is on N_1 and N_2. Also clear is the fact that the grid is not equispaced in the radial variable. The sampling grid for a space-limited function are plotted below to enable visualization. In the first instance, the polar grids are plotted for the case $R=1$, $N_1=16$ and $N_2=15$. These are shown in space (***r*** space) and frequency (***ρ*** space) in Figures 1 and 2 respectively.

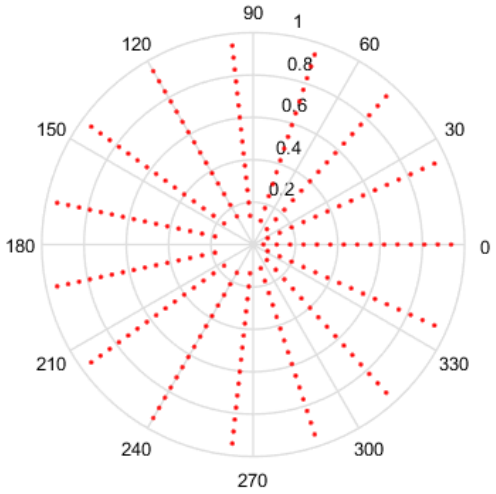

Figure 1 Sampling grid in space domain of a space limited function for R=1, N1=16 and N2 = 15

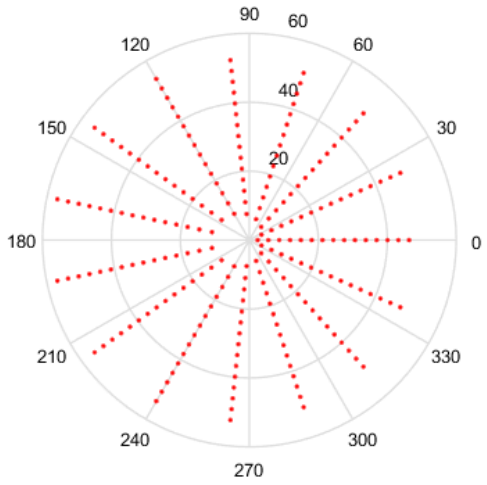

Figure 2 Sampling grid in frequency domain of a space limited function for R=1, N1=16 and N2 = 15

Clearly, the grids in Figure 1and Figure 2 are fairly sparse, but the low values of N_2 and N_1 have been chosen so that the structure of the sampling points can be easily seen. It can be observed that there is a hole at the center area in both domains which is caused by the special sampling points. For higher values of the N_2 and N_1, the grid becomes fairly dense, obtaining good coverage of both spaces, but details are harder to observe. To demonstrate, the polar grids are plotted for the case R =1, $N_1=96$ and $N_2=95$. These are shown in Figures 3 and 4 respectively.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

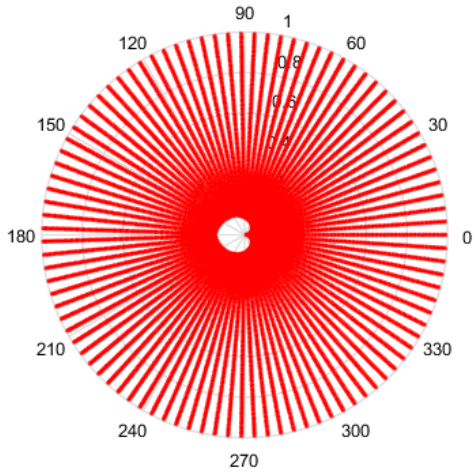

Figure 3 Sampling grid in space domain of a space limited function for R=1, N_1 = 96 and N_2 = 95

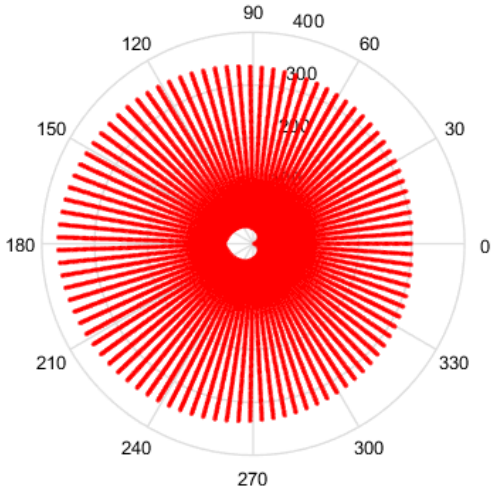

Figure 4 Sampling grid in frequency domain of a space limited function for R=1, N_1 = 96 and N_2 = 95

From Figure 3 and Figure 4, by choosing higher values of N_1 and N_2, the sampling grid becomes denser, however there is still a gap in the center area. The sampling grids for band-limited functions are not plotted here since the sample grid for a band-limited function has the same shape as with space limited function but the domains are reversed.

4.2 Sample Grid Analysis

From the first part of the paper, it was shown that the 2D-Fourier Transform can be interpreted as a Discrete Fourier Transform in the angular direction, a Discrete Hankel Transform in the radial direction and then an inverse Discrete Fourier Transform in the angular direction. Hence, the sample size in the angular direction could have been decided by the Nyquist sampling theorem [7], which states that

$$f_s > 2 f_{\max} \tag{16}$$

where f_s is the sample frequency and $f_{\max}$ is the highest frequency or band limit.

In the radial direction, the necessary relationship for the Discrete Hankel Transform is given by [8]

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

$$W_\rho R = j_{nN_1} \tag{17}$$

where W_ρ is the effective band-limit, R is the effective space limit and j_{nN} is the N^{th} zero of $J_n(r)$. For the 2D Fourier Transform, since $-M \le p \le M$, the order of the Bessel zero ranges from $-M$ to M, the relationship needed becomes

$$min(j_{pN_1}) \ge W_\rho R \tag{18}$$

The relationships $j_{nN} = j_{-nN}$ and $j_{0N_1} < j_{\pm 1N_1} < j_{\pm 2N_1} < \ldots < j_{\pm MN_1}$ are valid [9], hence equation (18) can be written as

$$j_{0N_1} \ge W_\rho R \tag{19}$$

It is pointed out in [10], [11] that the zeros of $J_n(z)$ are almost evenly spaced at intervals of π and that the spacing becomes exactly π in the limit as $z \to \infty$. In fact, it is shown in [10] that a simple asymptotic form for the Bessel function is given by

$$J_n(z) \approx \sqrt{\frac{2}{\pi z}} \cos\left[z - \left(n + \frac{1}{2} \right) \frac{\pi}{2} \right] \tag{20}$$

Therefore, an approximation to the Bessel zero, j_{nk} is given by

$$j_{nk} \approx \left(2k + n - \frac{1}{2} \right) \frac{\pi}{2} \tag{21}$$

Hence, equation (19) can be written to choose N_1 approximately as

$$\begin{aligned} & N_1 \pi \ge W_\rho R = 2\pi WR \\ & \Rightarrow N_1 \ge 2WR \end{aligned} \tag{22}$$

where the reader is reminded that the units of W is m^{-1} (the space equivalent of Hz). N_1 / R is the spatial sampling frequency and we see that equation (22) effectively makes the same statement as equation (16).

Intuitively, more sample points lead to more information captured, which gives an expectation that increasing N_1 or N_2 individually will give a better sampling grid coverage. However, it can be seen from Figure 1 to Figure 4 that there is a gap in the center of the sample grid. From equation (12) and (13), the area of the gap in the center is related to the ranges of p and k, that is N_2 and N_1. In the sections below, it is assumed that the sampling theorems are already satisfied (that is, an appropriate space and band limit is selected) and the relationship between N_2, N_1 and the size of the gap will be discussed.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

4.2.1 Space-limited function

In this section, it is assumed that the function is a space limited function, defined in $r \in [0, R]$. The sampling points are defined as equation (12) in the space domain and (13) in the frequency domain. In the following, a relationship between N_2 , N_1 and the area of the gap in both domains is discussed.

4.2.1.1 Sample grid in the space domain

In the space domain, the effective limit in the space domain, *R*, is fixed. To analyze how the values of N_2 and N_1 affect the coverage of the grid in space domain, consider the following definition of 'grid coverage'

$$A_r = \frac{\pi R^2 - \pi \bar{r}^2}{\pi R^2} \cdot 100 \tag{23}$$

where $\bar{r}$ denotes the average radius of the gap (the hole in the middle of the grid). A_r as defined in equation (23) is a measure of the 'grid coverage' since it gives a percentage of how much of the original space limited domain area is captured by the discrete grid. For example, if the average radius of the center gap is zero, then A_r would be 100%, that is, complete coverage. Based on the observation of Figure 1 and Figure 3, the relationship $r_{01} < r_{\pm 11} < r_{\pm 21} < r_{\pm M1}$ is valid. Therefore, from equation (12), the average radius of the gap is given by

$$\bar{r} = \frac{(r_{01} + r_{M1})}{2} = \frac{1}{2}\left(\frac{j_{01}}{j_{0N_1}} R + \frac{j_{M1}}{j_{MN_1}} R \right) \tag{24}$$

Hence, equation (23) can be written as

$$A_r = \left[1 - \frac{1}{4}\left(\frac{j_{01}}{j_{0N_1}} + \frac{j_{M1}}{j_{MN_1}} \right)^2 \right] \cdot 100 \tag{25}$$

Table 1 shows the different values of A_r as the values of N_1 and N_2 are changed.

Table 1 Spatial grid coverage, A_r, with respect to different values of N_1 and N_2 (R is fixed)

N1 / N2	15	75	150	300
15	$A_r = 98.48\%$	$A_r = 99.92\%$	$A_r = 99.98\%$	$A_r = 99.99\%$
75	$A_r = 93.78\%$	$A_r = 99.36\%$	$A_r = 99.81\%$	$A_r = 99.95\%$
151	$A_r = 90.14\%$	$A_r = 98.42\%$	$A_r = 99.46\%$	$A_r = 99.84\%$
301	$A_r = 86.17\%$	$A_r = 96.58\%$	$A_r = 98.59\%$	$A_r = 99.51\%$

From Table 1, it can be seen that increasing N_1 (sample size in the radial direction) tends to increase the grid coverage. Since the effective space limit R is fixed, from equation (19) it follows that increasing N_1 actually increases the effective band limit. However, increasing N_2

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

(sample size in angular direction) will result in a bigger gap in the center of the grid, which then decreases the coverage.

4.2.1.2 *Sample grid in the frequency domain*

Similarly, coverage of the grid in the frequency domain is defined as

$$A_\rho = \frac{\pi W_\rho^2 - \pi \bar{\rho}^2}{\pi W_\rho^2} \cdot 100 \tag{26}$$

where $\bar{\rho}$ denotes the average radius of the gap. Since

$$\bar{\rho} = \frac{(\rho_{01} + \rho_{M1})}{2} = \frac{(j_{01} + j_{M1})}{2R} \tag{27}$$

Then, it follows that equation (26) can be written as

$$A_\rho = \left[1 - \frac{(j_{01} + j_{M1})^2}{4R^2 W_\rho^2}\right] \cdot 100\% \tag{28}$$

From equation (28), it can be observed that the sample grid coverage in the frequency domain is affected by R , W_ρ and M . Since $N_2 = 2M + 1$, in order to get a better grid coverage with a fixed W_ρ, R and N_2 can be adjusted. Table 2 shows the grid coverage A_ρ for different values of R and N_2 .

Table 2 Frequency grid coverage, A_ρ, with respect to different values of R and N_2 (W_ρ is fixed)

R / N_2	15	75	150	300
15	$A_\rho = 99.80\%$	$A_\rho = 99.99\%$	$A_\rho = 100.00\%$	$A_\rho = 100.00\%$
75	$A_\rho = 97.66\%$	$A_\rho = 99.91\%$	$A_\rho = 99.98\%$	$A_\rho = 99.99\%$
151	$A_\rho = 91.88\%$	$A_\rho = 99.68\%$	$A_\rho = 99.92\%$	$A_\rho = 99.98\%$
301	$A_\rho = 70.67\%$	$A_\rho = 98.83\%$	$A_\rho = 99.71\%$	$A_\rho = 99.93\%$

From Table 2, the conclusion for the frequency domain is that when the effective band limit is fixed, increasing R (effective space limit) tends to increase the coverage in the frequency domain, while increasing N_2 (sample size in the angular direction) decreases the coverage. However, from equation (19) it should be noted that to satisfy the sampling theorem, increasing R with fixed W_ρ requires an increase in N_1 at the same time.

4.2.2 Band-limited function

In this section, we suppose that the function is an effectively band limited function, defined on $\rho \in [0, W_p]$. The sampling points are defined as in equation (14) in the space domain and as in

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

(15) in the frequency domain. In this subsection, the relationship between N_2 , N_1 and the area of the gap in both domains is discussed.

4.2.2.1 Sampling Grid in the space domain

The same definition of grid coverage in the space domain will be used as in equation (23). Since the sampling points of a band-limited function are given by equations (14) and(15), the average radius of the gap can be defined as

$$\bar{r} = \frac{(r_{01} + r_{M1})}{2} = \frac{1}{2}\left(\frac{j_{01}}{W_\rho} + \frac{j_{M1}}{W_\rho} \right) \tag{29}$$

Therefore, the coverage of the grid in space domain can be written as

$$A_r = \left[1 - \frac{(j_{01} + j_{M1})^2}{4W_\rho^2 R^2} \right] \cdot 100 \tag{30}$$

It can be observed that the grid coverage in the space domain of a band-limited function is the same as the grid coverage in the frequency domain of space limited function.

4.2.2.2 Sample Grid in frequency domain

The coverage of the grid in the frequency domain of a band limited function is defined by equation (26). With sampling points defined in equation (15) , the average radius of the gap can be defined as

$$\bar{\rho} = \frac{(\rho_{01} + \rho_{M1})}{2} = \frac{1}{2}\left(\frac{j_{01}}{j_{0N_1}} W_\rho + \frac{j_{M1}}{j_{MN_1}} W_\rho \right) \tag{31}$$

The coverage of the grid in frequency domain can be written as

$$A_\rho = \left[1 - \frac{1}{4}\left(\frac{j_{01}}{j_{0N_1}} + \frac{j_{M1}}{j_{MN_1}} \right)^2 \right] \cdot 100 \tag{32}$$

It can be observed that the grid coverage in the frequency domain of a band-limited function is the same as the grid coverage in the space domain of a space limited function.

4.3 Conclusion

Based on the discussion above, the following conclusions can be made:

1. Increasing N_2 (angular direction) tends to decrease the sampling grid coverage in both domains. Increasing N_1 (radial direction) tends to increase the sampling coverage in the space domain for a space-limited function and in the frequency domain for a frequency-limited function. So, if a signal changes sharply in the angular direction such that large values of N_2 are needed, a large value of N_1 is also needed to compensate for the effect of increasing N_2 on the grid coverage.

2. For a space-limited function, if there is a lot of energy at the origin in the space domain, a larger value of N_1 will be required to ensure that the sampling grid gets as close to the origin as possible in the space domain. If the function has a lot of energy at the origin in

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

the frequency domain, a large value for both N_1 and R will be required to ensure adequate grid coverage.

3. For a band-limited function, if there is a lot of energy at the origin in the frequency domain, a large value of N_1 will be needed to ensure that the sample grid gets as close to the origin as possible in the frequency domain. If the function has a lot of energy at the origin in the space domain, large values for both N_1 and W_ρ are required.

5 Numerical Computation of the Transform

We have already demonstrated in the first part of the paper that the discrete 2D Fourier transform in polar coordinates can be interpreted as a DFT, DHT and then inverse DFT. This interpretation is quite helpful in coding the transform and in exploiting the speed of the FFT (Fast Fourier Transform) in implementing the computations. In this section, we explain how the speed of Matlab's (Mathworks 2018) built-in code (or similar software) can be exploited to implement the 2D FT in polar coordinates.

5.1 Forward transform

To transform $f_{pk} \to F_{qm}$, the operation is performed in steps. The first step is a forward 1D DFT transforming $f_{pk} \to \bar{f}_{nk}$ where the p subscript is transformed to the n subscript:

$$\bar{f}_{nk} = \sum_{p=-M}^{M} f_{pk} e^{-in\frac{2\pi p}{N_2}} \qquad \text{for } n = -M..M,\ k = 1..N_1 - 1 \tag{33}$$

The overbar is used to indicate a standard 1D DFT. In matrix operations, this is equivalent to stating that each *column* of f_{pk} is DFT'ed to yield $\bar{f}_{nk}$. The second step is a discrete Hankel transform of order n, transforming $\bar{f}_{nk} \to \hat{\bar{f}}_{nm}$ so that the k subscript is Hankel transformed to the m subscript:

$$\hat{\bar{f}}_{nm} = \sum_{k=1}^{N_1-1} \frac{2J_n\left(\frac{j_{nk}j_{nm}}{j_{nN_1}}\right)}{j_{nN_1}J_{n+1}^2(j_{nk})} \bar{f}_{nk} = \sum_{k=1}^{N_1-1} Y_{m,k}^{nN_1} \bar{f}_{nk} \qquad \text{for } n = -M..M,\ m = 1..N_1 - 1 \tag{34}$$

Here, the overhat is used to indicate a Discrete Hankel Transform (DHT), with the DHT as defined in [8] via the transformation matrix

$$Y_{m,k}^{nN_1} = \frac{2}{j_{nN_1}J_{n+1}^2(j_{nk})} J_n\left(\frac{j_{nm}j_{nk}}{j_{nN_1}}\right) \qquad 1 \le m,k \le N_1 - 1 \tag{35}$$

In matrix operations, this states that each *row* of $\bar{f}_{nk}$ is DHT'ed to yield $\hat{\bar{f}}_{nm}$. These are now scaled to give the Fourier coefficients of the 2D DFT $\hat{\bar{f}}_{nm} \to \bar{F}_{nm}$ such that

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

$$\bar{F}_{nm} = \frac{2\pi R^2 i^{-n}}{j_{nN_1}} \hat{\bar{f}}_{nm} = \frac{2\pi R^2 i^{-n}}{j_{nN_1}} \sum_{k=1}^{N_1-1} Y_{m,k}^{nN_1} \bar{f}_{nk} \qquad \text{for } n = -M..M,\ \ m = 1..N_1 - 1 \quad (36)$$

It is noted that the step in equation (36) exactly parallels the continuous form equivalent step where $F_n(\rho) = 2\pi\, i^{-n} \mathbb{H}_n\{f_n(r)\}$, see [12], [13].

The final step of the forward 2D DFT in polar coordinates is then a standard *inverse* 1D DFT, which transforms each *column* of $\bar{F}_{nm} \to F_{qm}$ so that the n subscript is (inverse) DFT transformed to the q subscript via

$$F_{qm} = \frac{1}{N_2} \sum_{n=-M}^{M} \bar{F}_{nm} e^{+in\frac{2\pi q}{N_2}} \qquad \text{for } q = 0..N_2 - 1,\ \ m = 1..N_1 - 1 \quad (37)$$

This last step is a 1D Inverse Discrete Fourier Transform (IDFT) for each *column* of $\bar{F}_{nm}$ to obtain F_{qm}. The inverse 2D DFT can be similarly interpreted, shown in the next subsection.

5.2 Inverse Transform

The steps of the inverse 2D DFT are the reverse of the steps outlined above for the forward 2D DFT. First, $F_{qm} \to \bar{F}_{nm}$ via a forward 1D DFT

$$\bar{F}_{nm} = \sum_{q=M}^{M} F_{qm} e^{-i\frac{2\pi nq}{N_2}} \qquad n = -M..M,\ \ m = 1..N_1 - 1 \quad (38)$$

Then, a discrete Hankel transform is performed to transform $\bar{F}_{nm} \to \hat{\bar{F}}_{nk}$ such that

$$\hat{\bar{F}}_{nk} = \sum_{m=1}^{N_1-1} \frac{2J_n\left(\frac{j_{nm} j_{nk}}{j_{nN_1}}\right)}{j_{nN_1} J_{n+1}^2(j_{nm})} \bar{F}_{nm} = \sum_{m=1}^{N_1-1} Y_{k,m}^{nN_1} \bar{F}_{nm} \qquad \text{for } n = -M..M,\ \ k = 1..N_1 - 1 \quad (39)$$

This is followed by a scaling operation to obtain $\hat{\bar{F}}_{nk} \to \bar{f}_{nk}$ from

$$\bar{f}_{nk} = \frac{j_{nN_1} i^{+n}}{2\pi R^2} \hat{\bar{F}}_{nk} \qquad \text{for } n = -M..M,\ \ k = 1..N_1 - 1 \quad (40)$$

Finally, an inverse 1D IDFT is applied to obtain $\bar{f}_{nk} \to f_{pk}$ from

$$f_{pk} = \frac{1}{N_2} \sum_{n=-M}^{M} \bar{f}_{nk} e^{+i\frac{2\pi np}{N_2}} \qquad \text{for } p = -M..M,\ \ k = 1..N_1 - 1 \quad (41)$$

As previously mentioned, this parallels the steps taken for the continuous case, with each continuous operation (Fourier series, Hankel transform) replaced by its discrete counterpart (DFT, DHT).

Therefore, for both forward and inverse 2D-DFT, the sequence of operations is a DFT of each column of the starting matrix, followed by a DHT of each row, a term-by-term scaling,

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

followed by an IDFT of each column. This is a significant computational improvement because by interpreting the transform this way, the Fast Fourier Transform (FFT) can be used, which reduces the computational time quite significantly.

5.3 Interpretation of the sampled forward transform in Matlab terms

To use the built-in Matlab function fft, a few operations are required. First, we define matlab-friendly indices $p' = p + (M+1)$ and $n' = n + (M+1)$ so that $p, n = -M..M$ become $p', n' = 1..2M+1 = 1..N_2$ (since $2M+1 = N_2$), that is the primed variables range from $1..2M$ rather than $-M..M$. Hence, if the matrix $\mathbf{f}$ with entries $f_{p'k}$ is defined, where $p' = 1..N_2$, $k = 1..N_1 - 1$, then equation (33) can be written as the Matlab-defined DFT as

$$\bar{f}_{n'k} = \sum_{p'=1}^{N_2} f_{pk} e^{\frac{-2\pi i(p'-1-M)(n'-1-M)}{N_2}} \tag{42}$$

The definition of DFT in Matlab is actually given by the relationship

$$\bar{f}_{n'k} = \sum_{p'=1}^{N_2} f_{p'k} e^{\frac{-2\pi i(p'-1)(n'-1)}{N_2}} \tag{43}$$

Since the relationship $\sum_{p'=1}^{N_2} f_{p'k} e^{\frac{-2\pi i(p'-1)(n'-1-M)}{N2}} = \sum_{p'=1}^{N_2} f_{pk} e^{\frac{-2\pi i(p'-1-M)(n'-1-M)}{N2}}$ is valid, we can sample the original function to obtain the discrete f_{pk} values, put them in the matrix $f_{p'k}$ then shift the matrix $f_{p'k}$ by $M+1$ along the column direction. In Matlab, the function $circshift(A, K, dim)$ can be used, which circularly shifts the values in array A by K positions along dimension dim. Inputs K and dim must be scalars. Specifically, dim = 1 indicates the columns of matrix A and dim = 2 indicates the rows of matrix A. Hence, equation (42) can be written as

$$\bar{f}_{n'k} = fft\left(circshift\left(f_{p'k}, M+1, 1\right), N_2, 1\right) \tag{44}$$

In matrix operations, this is equivalent to stating that each *column* of $f_{p'k}$ is DFT'ed to yield $\bar{f}_{n'k}$.

The second step (equation (34)) is a discrete Hankel transform of order n, transforming $\bar{f}_{n'k} \to \hat{\bar{f}}_{n'l}$ so that the k subscript is Hankel transformed to the l subscript. In order to relate the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

order n to the index n', we need to shift $\bar{f}_{n'k}$ by $-(M+1)$ along column direction so that the order ranges from –M to M.

$$\hat{\bar{f}}_{n'l} = \sum_{k=1}^{N_1-1} \frac{2J_n\left(\frac{j_{nk}\, j_{nl}}{j_{nN_1}}\right)}{j_{nN_1} J_{n+1}^2\left(j_{nk}\right)} circshift\left(\bar{f}_{n'k}, -(M+1), 1\right) \qquad \begin{cases} \text{for } n' = 1..N_2, \;\; l = 1..N_1 - 1 \\ \text{where } n = n' - \left(M+1\right) \end{cases} \quad (45)$$

By using the Hankel transform matrix defined in [8], equation (45) can be rewritten as

$$\hat{\bar{f}}_{n'l} = circshift\left(\bar{f}_{n'k}, -(M+1), 1\right) \left(Y_{l,k}^{nN_1}\right)^T \qquad \begin{cases} \text{for } n' = 1..N_2, \;\; l = 1..N_1 - 1 \\ \text{where } n = n' - M - 1 \end{cases} \quad (46)$$

In matrix operations, this states that each *row* of $\bar{f}_{n'k}$ is DHT'ed to yield $\hat{\bar{f}}_{n'l}$. These are now scaled to give the Fourier coefficients of the 2D DFT $\hat{\bar{f}}_{n'l} \rightarrow \bar{F}_{n'l}$. In order to proceed to an inverse DFT in the next step, it is necessary to shift the matrix by $M+1$ along the column direction after scaling

$$\bar{F}_{n'l} = circshift\left(\frac{2\pi R^2}{j_{nN_1}} i^{-n} \hat{\bar{f}}_{n'l}, M+1, 1\right) \qquad \begin{cases} \text{for } n' = 1..N_2, \;\; l = 1..N_1 - 1 \\ \text{where } n = n' - \left(M+1\right) \end{cases} \quad (47)$$

This last step is a 1D IDFT for each *column* of $\bar{F}_{n'l}$ to obtain F_{ql}. Using $2M+1 = N_2$, and $q' = q+1+M$, equation (37) can be written as

$$\begin{aligned} F_{q'l} &= \frac{1}{N_2} \sum_{n'=1}^{N_2} \bar{F}_{nl} e^{+i(n'-M-1)\frac{2\pi(q'-1-M)}{N_2}} \qquad \text{for } q' = 1..N_2, \;\; l = 1..N_1 - 1 \\ &= \frac{1}{N_2} \sum_{n'=1}^{N_2} \bar{F}_{n'l} e^{+i(n'-1)\frac{2\pi(q'-1-M)}{N_2}} \\ &= circshift\left(ifft\left(\bar{F}_{n'l}, N_2, 1\right), -\left(M+1\right), 1\right) \end{aligned} \quad (48)$$

5.4 Interpretation of the sampled inverse transform in Matlab terms

Similarly, matlab-friendly indices $q' = q + (M+1)$ and $n' = n + (M+1)$ are also defined. Hence, if the matrix F with entries $F_{q'l}$ is defined, where $q' = 1..N_2$, $l = 1..N_1 - 1$, it then follows that equation (38) can be written as the matlab-defined DFT as

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

$$\begin{aligned}\bar{F}_{n'l} &= \sum_{q'=1}^{N_2} F_{ql} e^{-i(n'-M-1)\frac{2\pi(q'-1-M)}{N_2}} \qquad for\ n'=1..N_2,\ \ l=1..N_1-1 \\ &= \sum_{q'=1}^{N_2} F_{q'l} e^{-i(n'-M-1)\frac{2\pi(q'-1)}{N_2}}\end{aligned} \tag{49}$$

If the original function can be sampled as F_{ql} and then put into matrix $F_{q'l}$, then we need an *circshift* operation. So equation (49) can be written as

$$\bar{F}_{n'l} = fft\left(circshift(F_{q'l}, M+1,1), N_2, 1\right) \tag{50}$$

Subsequently, a discrete Hankel transform of order n is required, transforming $\bar{F}_{n'l} \to \hat{\bar{F}}_{n'l}$ so that the l subscript is Hankel transformed to the k subscript. To achieve this, *circshift* is also needed here.

$$\hat{\bar{F}}_{n'k} = circshift\left(\bar{F}_{n'l}, -(M+1),1\right)\left(Y_{k,l}^{nN_1}\right)^T \qquad \begin{cases}\text{for } n'=1..N_2,\ \ l=1..N_1-1 \\ \text{where } n=n'-M-1\end{cases} \tag{51}$$

This is followed by a scaling operation to obtain $\hat{\bar{F}}_{n'k} \to \bar{f}_{n'k}$ and then a *circshift* by $(M+1)$ so that

$$\bar{f}_{n'k} = circshift\left(\frac{j_{nN_1}}{2\pi R^2} i^{+n} \hat{\bar{F}}_{n'k}, (M+1),1\right) \qquad \begin{cases}\text{for } n'=1..N_2,\ \ k=1..N_1-1 \\ \text{where } n=n'-\left(M+1\right)\end{cases} \tag{52}$$

This last step is a 1D IDFT for each *column* of $\bar{f}_{n'k}$ to get $f_{p'k}$. Using $2M+1=N_2$, and $p'=p+1$, equation (41) can be written as

$$\begin{aligned}f_{p'k} &= \frac{1}{N_2}\sum_{n'=1}^{N_2} \bar{f}_{nk} e^{+i(n'-M-1)\frac{2\pi(p'-1-M)}{N_2}} \qquad \text{for } p'=1..N_2,\ \ k=1..N_1-1 \\ &= \frac{1}{N_2}\sum_{n'=1}^{N_2} \bar{f}_{n'k} e^{+i\frac{2\pi(n'-1)(p'-1-M)}{N_2}} \\ &= circshift\left(ifft\left(\bar{f}_{n'k}, N_2, 1\right), -(M+1), 1\right)\end{aligned} \tag{53}$$

In conclusion, in this section, by using the interpretation of the kernel as sequential DFT, DHT and IDFT operations, Matlab (or similar software) built-in code can be used to efficiently implement the 2D DFT algorithm in polar coordinates.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

6 Numerical evaluation of the 2D DFT in polar coordinates to approximate the continuous FT

In this section, the 2D discrete Fourier transform is evaluated for its ability to estimate the continuous Fourier transform at the selected special sampling points in the spatial and frequency domains.

6.1 Method for testing the Algorithm

6.1.1 Accuracy

In order to test accuracy of the 2D-DFT and 2D-IDFT to calculate approximate the continuous counterpart, the dynamic error is proposed as a metric. The dynamic error is defined as [11]

$$E(\mathrm{v}) = 20\log_{10}\left[\frac{|C(\mathrm{v}) - D(\mathrm{v})|}{\max|D(\mathrm{v})|}\right] \tag{54}$$

where $C(\mathrm{v})$ is the continuous forward or inverse 2D-Fourier transform and $D(\mathrm{v})$ is the value obtained from the discrete counterpart. The dynamic error is defined as the ratio of the absolute error to the maximum amplitude of the discrete function, calculated on a log scale. Therefore, a large negative value represents an accurate discrete transform. The dynamic error is used instead of the percentage error in order to avoid division by zero.

6.1.2 Precision

The precision of the algorithm is an important evaluation criterion, which is tested by sequentially performing a pair of forward and inverse transforms and comparing the result to the original function. High precision indicates that numerical evaluation of the transform does not add much error. An average of the absolute error between the original function and the calculated counterpart at each sample point is used to measure the precision. It is given by

$$\varepsilon = \frac{1}{(N_1 - 1)\cdot N_2}\sum_{n=1}^{(N_1-1)\cdot N_2}\left|f - f^*\right| \tag{55}$$

where f is the original function and f^* is the calculated counterpart. An ideal precision would result in the absolute error being zero.

6.2 Test functions

In this section, three test functions are chosen to evaluate the ability of the discrete transform to approximate the continuous counterpart. The first test case is the circularly symmetric Gaussian function. Given that it is circularly symmetric and that the Gaussian is continuous and smooth, the proposed DFT is expected to perform well. The second test case is “Four-term sinusoid and Sinc” function, which is not symmetric in the angular direction and suffers a discontinuity in the radial direction. The third test function presents a more challenging test function, the “Four-term sinusoid and Modified exponential” function. In this case, the test function is not circularly symmetric and it explodes at the origin (approaches infinity at the origin). Given that as shown above, the sampling grid cannot cover the area around the origin very well, a function that explodes at the origin should give more error and should provide a reasonable test case for evaluating the performance of the discrete transform.

6.2.1 Gaussian

The first function chosen for evaluation is a circular symmetric function which is Gaussian in the radial direction. Specifically, the function in the space domain is given by

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

$$f(r,\theta) = e^{-a^2 r^2} \tag{56}$$

where a is some real constant. Since the function is circularly symmetric, the 2D-DFT is a zeroth-order Hankel Transform [14] and is given by

$$F(\rho,\psi) = \frac{\pi}{a^2} e^{\frac{-\rho^2}{4a^2}} \tag{57}$$

The graphs for the original function and its continuous 2D-DFT (which is also a Gaussian) are plotted with $a = 1$ and shown in Figure 5.

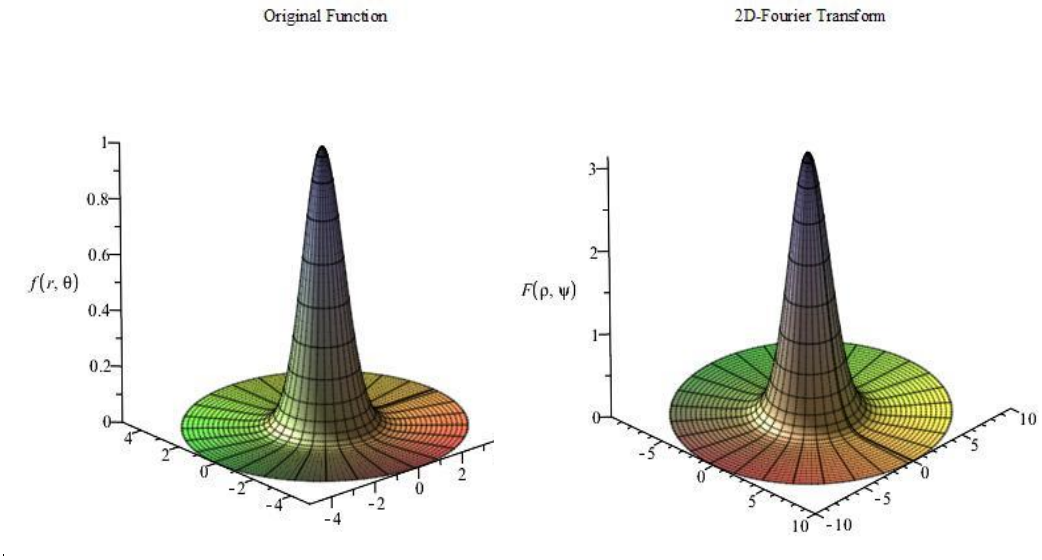

Figure 5 the original Gaussian function and its 2D-Fourier Transform

From Figure 5, the function is circular symmetric and fairly smooth in the radial direction. Moreover, the function can be considered as either an effectively space-limited function or an effectively band-limited function. For the purposes of testing it, it shall be considered as a space-limited function and equations (12) and (13) will be used to proceed with the forward and inverse transform in sequence.

To perform the transform, the following variables need to be chosen: N_2, R and N_1. In the angular direction, since the function in the spatial domain is circularly symmetric, N_2 can be chosen to be small. Thus, $N_2 = 15$ is chosen.

In the radial direction, from plotting the function, it can be seen that the effective space limit can be taken to be $R = 5$ and the effective band limit can be taken to be $W_\rho = 10$. From equation (19), $j_{0N_1} \geq R \cdot W_\rho = 50$. Therefore, $N_1 = 17$ is chosen (we could also have obtained a rough estimate of N_1 from equation (22)). However, most of the energy of the function in both the space and frequency domains is located in the center near the origin. Based on the discussion in Section 4.3, relatively large values of R and W_ρ are needed. The effective space limit $R = 40$ and effective band-limit $W_p = 30$ are thus chosen, which gives $j_{0N_1} \geq R \cdot W_\rho = 1200$. Therefore $N_1 = 383$ is chosen in order to satisfy this constraint. Both cases discussed here ($N_1 = 17$ and $N_1 = 383$) are tested in following.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

6.2.1.1 *Forward Transform*

Test results with $R = 5$, $N_1 = 17$ are shown in Figure 6 and Figure 7. Figure 6 shows the sampled continuous forward transform and the discrete forward transform. Figure 7 shows the error between the sampled values of the continuous transform and the discretely calculated values.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

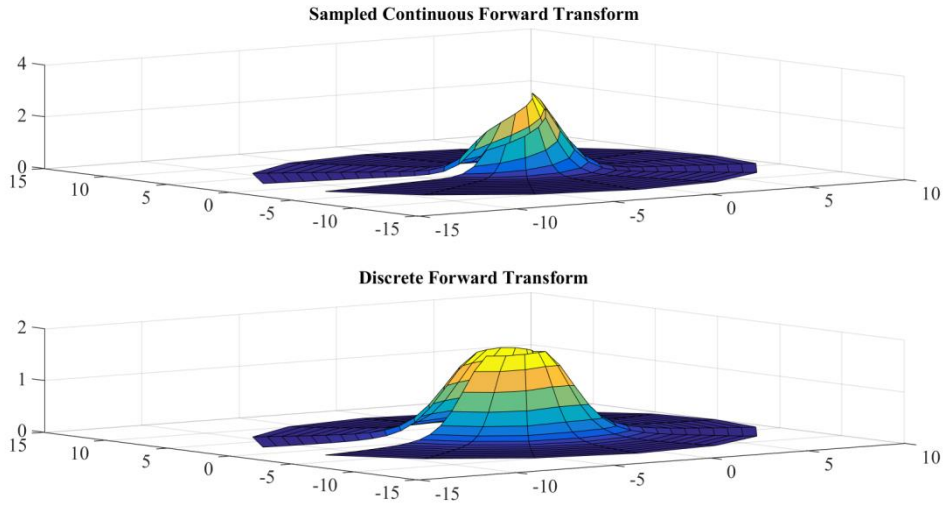

Figure 6 Sampled continuous forward transform and discrete forward transform of the Gaussian function with R=5, N_2=15, N_1=17

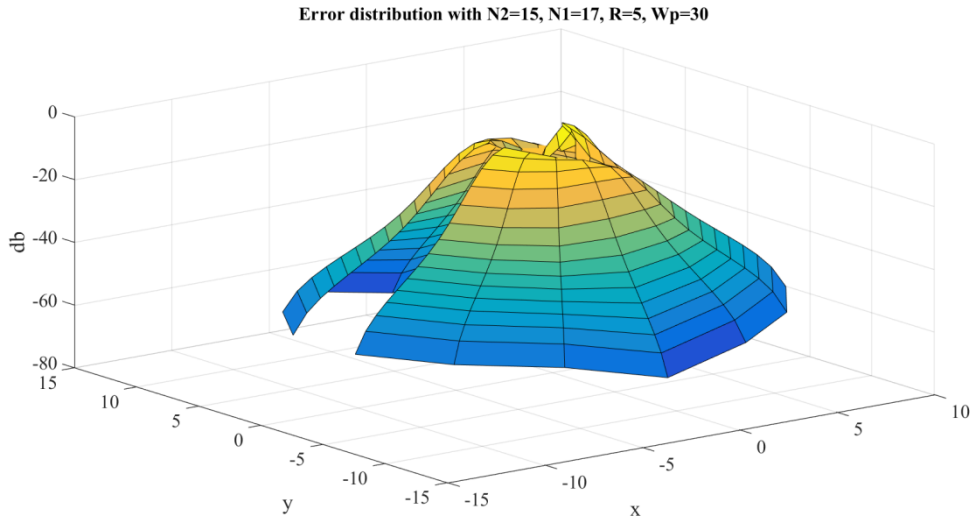

Figure 7 The error distribution of the forward transform of Gaussian Function with R=5, N_2=15, N_1=17

From Figure 7, it can be observed that the error gets bigger at the center, which is as expected because the sampling grid shows that the sampling points can never attain the origin. The maximum value of the error is $E_{\max} = -0.9115\,dB$ and this occurs at the center. The average error is $E_{avg.} = -30.4446dB$.

Test results with $R = 40$, $N_1 = 383$ are shown in Figure 8 and Figure 9. Similar to the previous case, the error gets larger at the center, as expected. However, the maximum value of the error is $E_{\max} = -8.3842\,dB$ and this occurs at the center. The average value of the error is $E_{avg.} = -63.8031dB$. Clearly, the test with $R = 40$, $N_1 = 383$ gives a better approximation, which verifies the discussion in Section 4.3.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

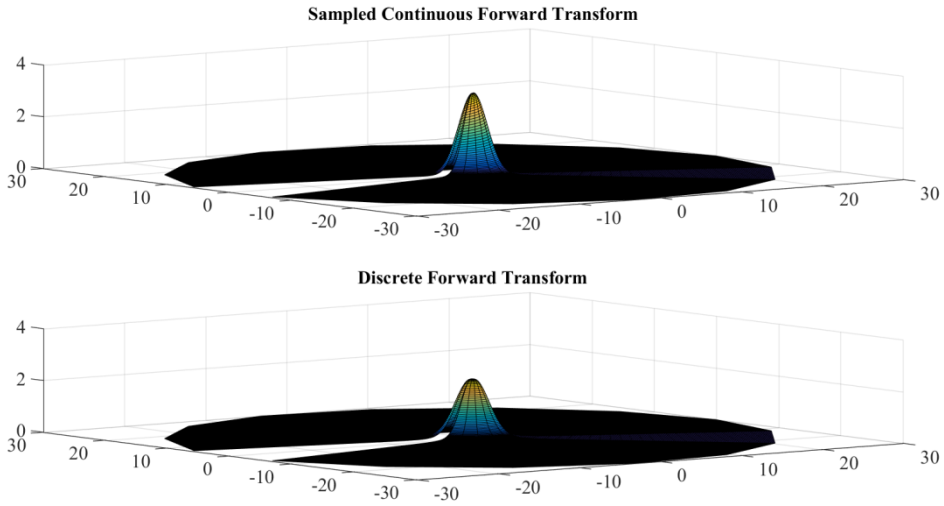

Figure 8 Sampled Continuous forward transform and Discrete forward transform of Gaussian Function with R=40, N_2=15, N_1=383

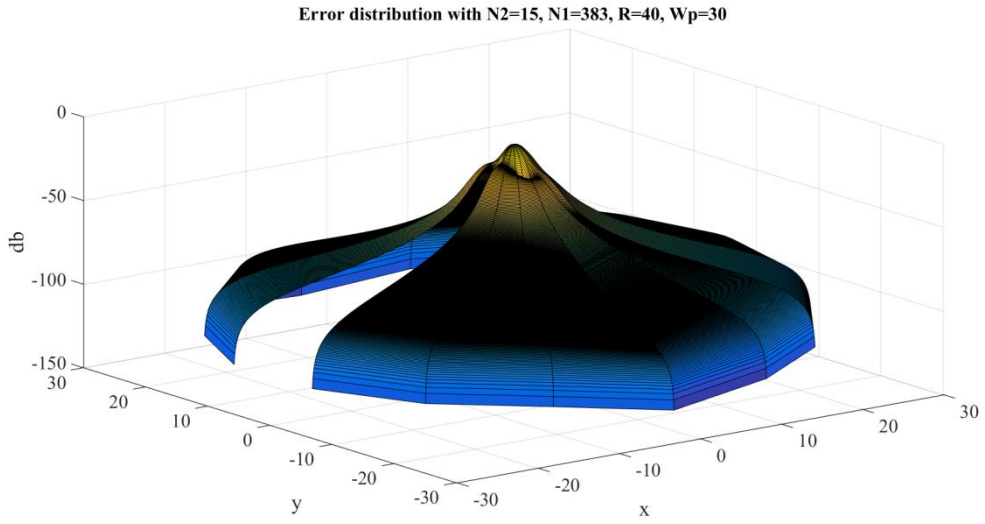

Figure 9 The error distribution of the forward transform of Gaussian Function with R=40, N_2=15, N_1=383

With $R = 40$, Table 3 shows the errors (max and average error) with respect to different value of N_1 and N_2. The trends as functions of N_1 and N_2 are shown as plots in Figure 10 and Figure 11.

Table 3 Error (dB) of forward transform of Gaussian Function with R=40, different value of N_1 and N_2

N2 \ N1	283	333	383	433	483

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at *PeerJ Computer Science* **2020**; doi:10.7717/peerj-cs.257

3	$E_{max.} = -21.6$ $E_{avg.} = -71.3$	$E_{max.} = -23.0$ $E_{avg.} = -76.9$	$E_{max.} = -24.3$ $E_{avg.} = -81.8$	$E_{max.} = -25.4$ $E_{avg.} = -86.0$	$E_{max.} = -26.3$ $E_{avg.} = -89.8$
7	$E_{max.} = -12.9$ $E_{avg.} = -62.6$	$E_{max.} = -14.4$ $E_{avg.} = -68.3$	$E_{max.} = -15.7$ $E_{avg.} = -73.2$	$E_{max.} = -16.9$ $E_{avg.} = -77.5$	$E_{max.} = -17.8$ $E_{avg.} = -81.4$
15	$E_{max.} = -5.4$ $E_{avg.} = -53.1$	$E_{max.} = -7.0$ $E_{avg.} = -58.9$	$E_{max.} = -8.4$ $E_{avg.} = -63.8$	$E_{max.} = -9.6$ $E_{avg.} = -68.1$	$E_{max.} = -10.6$ $E_{avg.} = -72.0$
31	$E_{max.} = 2.3$ $E_{avg.} = -42.0$	$E_{max.} = 0.5$ $E_{avg.} = -47.6$	$E_{max.} = -1.0$ $E_{avg.} = -52.5$	$E_{max.} = -2.3$ $E_{avg.} = -56.9$	$E_{max.} = -3.4$ $E_{avg.} = -60.7$
61	$E_{max.} = 9.7$ $E_{avg.} = -32.5$	$E_{max.} = 7.9$ $E_{avg.} = -37.5$	$E_{max.} = 6.4$ $E_{avg.} = -42.0$	$E_{max.} = 5.0$ $E_{avg.} = -46.1$	$E_{max.} = 3.8$ $E_{avg.} = -49.8$

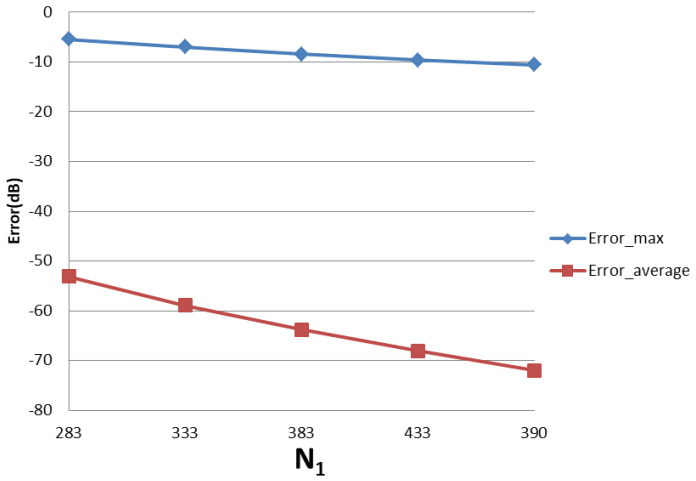

Figure 10 Error of forward transform of Gaussian Function with fixed N_2 (15) and varying N_1

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

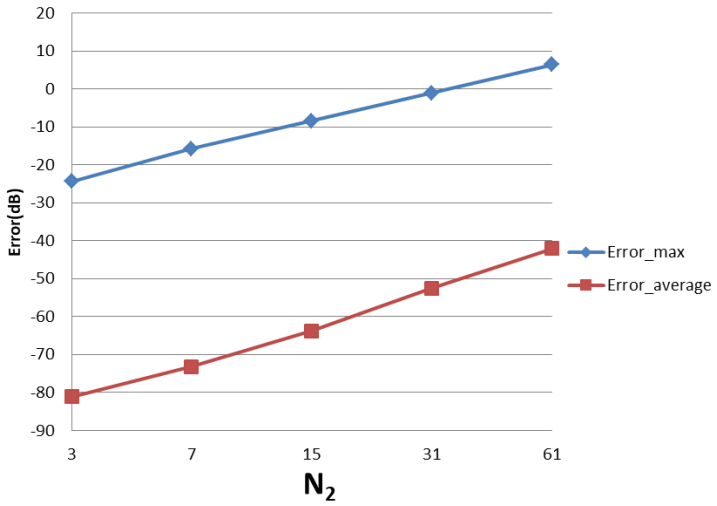

Figure 11 Error of forward transform of Gaussian Function with fixed N_1 (383) and varying N_2

From Figure 10, it can be seen that when N_1 individually (N_2 is fixed at N_2 =15) is less than the minimum of 383 obtained from the sampling theorem, increasing N_1 will lead to smaller errors, as expected. When N_1 is bigger than the sampling-theorem threshold of 383, increasing N_1 still decreases the error which verifies the discussion about sample grid coverage in Section 4.3. Increasing N_1 tends to increase the sample grid coverage and capture more information at the center area and thus leads to smaller errors.

From Figure 11, increasing N_2 alone (that is, without a corresponding increase in N_1) leads to larger errors, both $Error_{\max}$ and $Error_{average}$. Although at first counterintuitive, this result is actually reasonable because the function is radially symmetric which implies that $N_2 = 1$ should be sufficient based on the sampling theorem for the angular direction. Therefore, increasing N_2 will not lead to a better approximation. Moreover, from the discussion of the sample grid coverage in Section 4.3, the sampling grid coverage in both domains gets worse when N_2 gets bigger because more information from the center is lost. This problem can be solved by increasing N_1 at the same time, but it could be computationally time consuming. Therefore, choosing N_2 properly is very important from the standpoint of accuracy and computational efficiency.

6.2.1.2 Inverse Transform

Test results for the inverse transform with $R = 5$, $N_1 = 17$ are shown in Figure 12 and Figure 13. Figure 12 shows the sampled continuous inverse transform and discrete inverse transform and Figure 13 shows the error between the sampled continuous and discretely calculated values.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

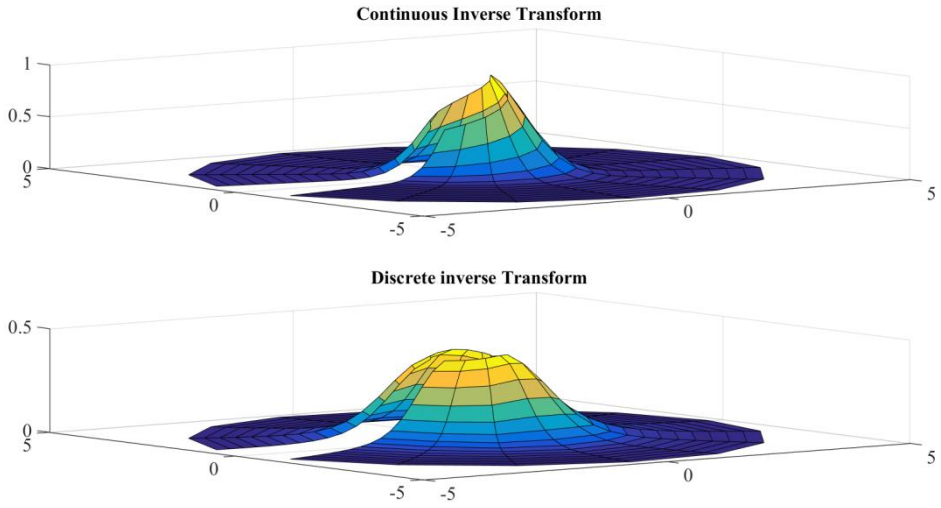

Figure 12 Sampled continuous inverse transform and discrete inverse transform of the Gaussian function with R=5, N_2=15, N_1=17

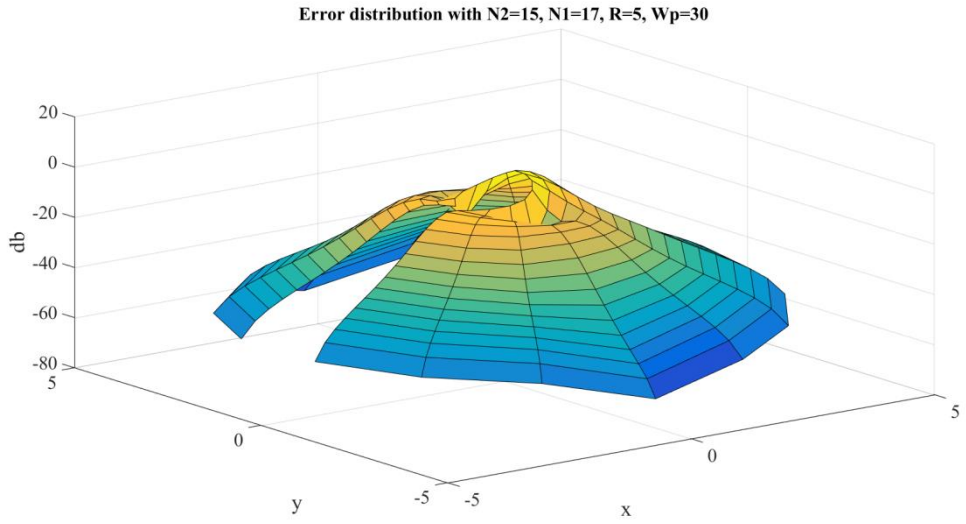

Figure 13 The error distribution of the inverse transform of Gaussian Function with R=5, N_2=15, N_1=17

Similar to the case for the forward transform, the error gets larger at the center, which is as expected because the sampling grid shows that the sampling points never attain the center. The maximum value of the error is $E_{\max} = 3.1954\,dB$ and this occurs at the center. The average of the error is $E_{avg.} = -25.7799 dB$.

Test results for the inverse transform with $R = 40$, $N_1 = 383$ are shown in Figure 14 and Figure 15.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

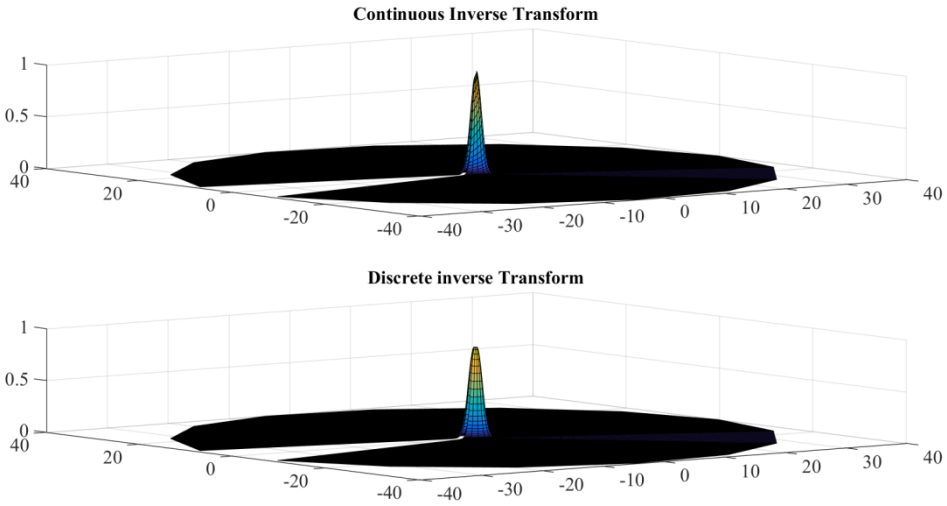

Figure 14 Sampled continuous inverse transform and discrete inverse transform of the Gaussian function with R=40, N_2=15, N_1=383

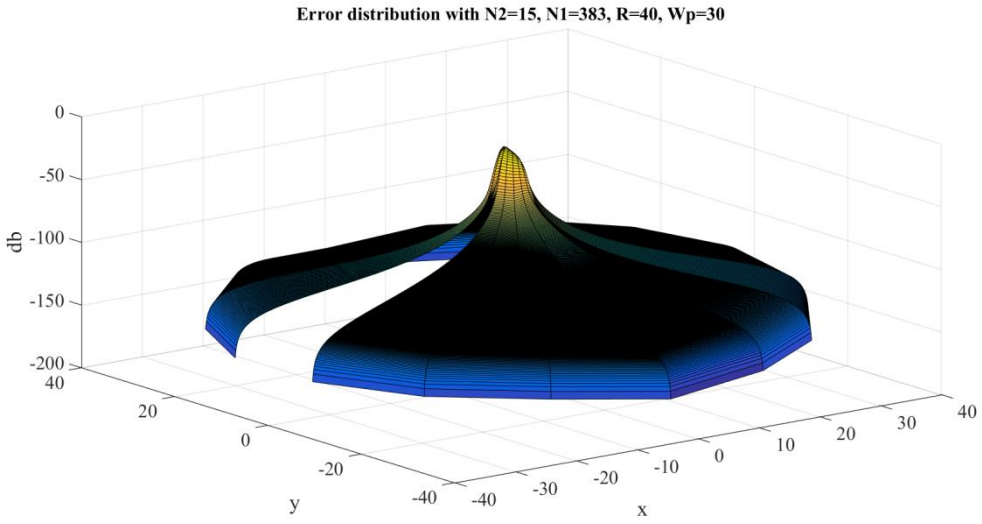

Figure 15 The error distribution of the inverse transform of Gaussian Function with R=40, N_2=15, N_1=383

In this case, the maximum value of the error is $E_{\max} = -12.2602\,dB$ and this occurs at the center. The average of the error is $E_{avg.} = -98.0316 dB$. Table 4 shows the errors with respect to different value of N_1 and N_2, from which Figure 16 and Figure 17 demonstrate the trend.

Table 4 Error (dB) of inverse transform of Gaussian Function with R=40, different value of N_1 and N_2

N2 \ N1	283	333	383	433	483

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at *PeerJ Computer Science* **2020**; doi:10.7717/peerj-cs.257

3	$E_{max.} = -25.9$ $E_{avg.} = -115.3$	$E_{max.} = -27.5$ $E_{avg.} = -115.4$	$E_{max.} = -28.9$ $E_{avg.} = -115.4$	$E_{max.} = -30.2$ $E_{avg.} = -115.5$	$E_{max.} = -31.3$ $E_{avg.} = -115.5$
7	$E_{max.} = -16.5$ $E_{avg.} = -107.0$	$E_{max.} = -18.1$ $E_{avg.} = -107.1$	$E_{max.} = -19.4$ $E_{avg.} = -107.2$	$E_{max.} = -20.5$ $E_{avg.} = -107.2$	$E_{max.} = -21.6$ $E_{avg.} = -107.2$
15	$E_{max.} = -9.7$ $E_{avg.} = -97.9$	$E_{max.} = -11.0$ $E_{avg.} = -98.0$	$E_{max.} = -12.3$ $E_{avg.} = -98.0$	$E_{max.} = -13.4$ $E_{avg.} = -98.1$	$E_{max.} = -14.4$ $E_{avg.} = -98.1$
34	$E_{max.} = -4.4$ $E_{avg.} = -86.9$	$E_{max.} = -5.5$ $E_{avg.} = -86.9$	$E_{max.} = -6.5$ $E_{avg.} = -87.0$	$E_{max.} = -7.5$ $E_{avg.} = -87.0$	$E_{max.} = -8.3$ $E_{avg.} = -87.0$
61	$E_{max.} = -1.1$ $E_{avg.} = -75.6$	$E_{max.} = -1.7$ $E_{avg.} = -75.6$	$E_{max.} = -2.4$ $E_{avg.} = -75.6$	$E_{max.} = -3.0$ $E_{avg.} = -75.6$	$E_{max.} = -3.7$ $E_{avg.} = -75.7$

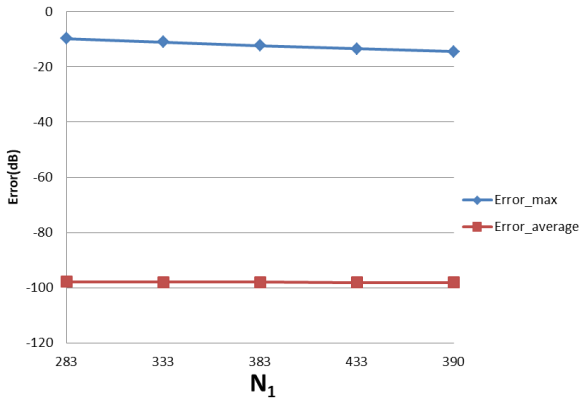

Figure 16 Error of inverse transform of Gaussian Function with fixed N_2 (15) and varying N_1

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

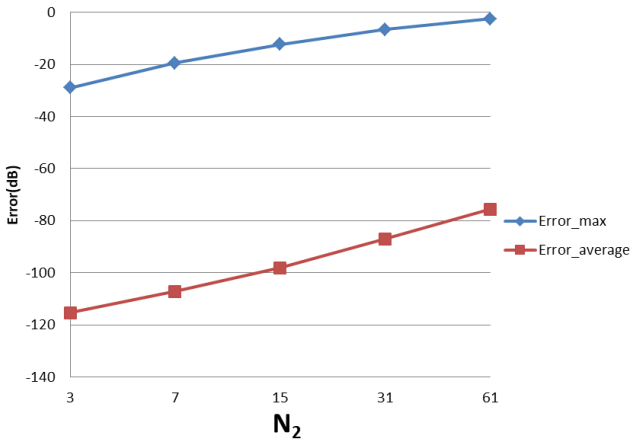

Figure 17 Error of inverse transform of Gaussian Function with fixed N_1 (383) and varying N_2

From Figure 16 it can be observed that increasing N_1 tends to improve the result but not significantly. This could be explained by the discussion for $R = 40$, $N_1 = 383$ that with fixed R and W_ρ, increasing N_1 will not allow the sampling grid in the frequency domain to get any closer to the origin to capture more information. From Figure 17, increasing N_2 (with fixed $N_1 = 383$) leads to a worse approximation which verifies the discussion for $R = 40$, $N_1 = 383$.

Performing sequential 2D-DFT and 2D-IDFT results in $\varepsilon = 4.1656 \times e^{-17}$ where ε is calculated with equation (55) . Therefore, performing sequential forward and inverse transforms does not add much error.

6.2.2 Four-term sinusoid & Sinc Function

The second function chosen for evaluation is given by

$$f(r,\theta) = \frac{\sin(ar)}{ar}[3\sin(\theta) + \sin(3\theta) + 4\cos(10\theta) + 12\sin(15\theta)] \qquad (58)$$

which is a sinc function in the radial direction and a four-term sinusoid in the angular direction. The continuous 2D-FT can be calculated from [12]

$$F(\rho,\psi) = \sum_{n=-\infty}^{\infty} 2\pi i^{-n} e^{in\psi} \int_0^{\infty} f_n(r) J_n(\rho r) r dr \qquad (59)$$

where $f_n(r)$ is the Fourier series of $f(r,\theta)$ and can be written as

$$f_n(r) = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(r,\theta) e^{-in\theta} d\theta \qquad (60)$$

From the sampling theorem for the angular direction, the highest angular frequency in equation (58) results in N_2 being *at least* 31 required to reconstruct the signal. Therefore, at least 31 terms are required to calculate the continuous 2D-FT, which can be written as

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

$$F(\rho,\psi)=\begin{cases}\dfrac{8\pi\cos(10\psi)\rho^{10}}{a\sqrt{a^2-\rho^2}\left(a+\sqrt{a^2-\rho^2}\right)^{10}}, & \rho<a\\ -\dfrac{6\pi i\sin(\psi)}{a\rho\sqrt{\rho^2+a^2}}+\dfrac{2\pi i\sin\left(3\arcsin\left(\dfrac{a}{\rho}\right)\right)\sin(3\psi)}{\sqrt{\rho^2+a^2}}-\dfrac{8\pi\sin\left(10\arcsin\left(\dfrac{a}{\rho}\right)\right)\cos(10\psi)}{\sqrt{\rho^2+a^2}}\\ +\dfrac{24\pi i\sin\left(15\arcsin\left(\dfrac{a}{\rho}\right)\right)\sin(15\psi)}{\sqrt{\rho^2+a^2}}, & \rho>a\end{cases} \quad (61)$$

The graphs for the original function and the magnitude of its continuous 2D-FT with $a=5$ are shown in Figure 18

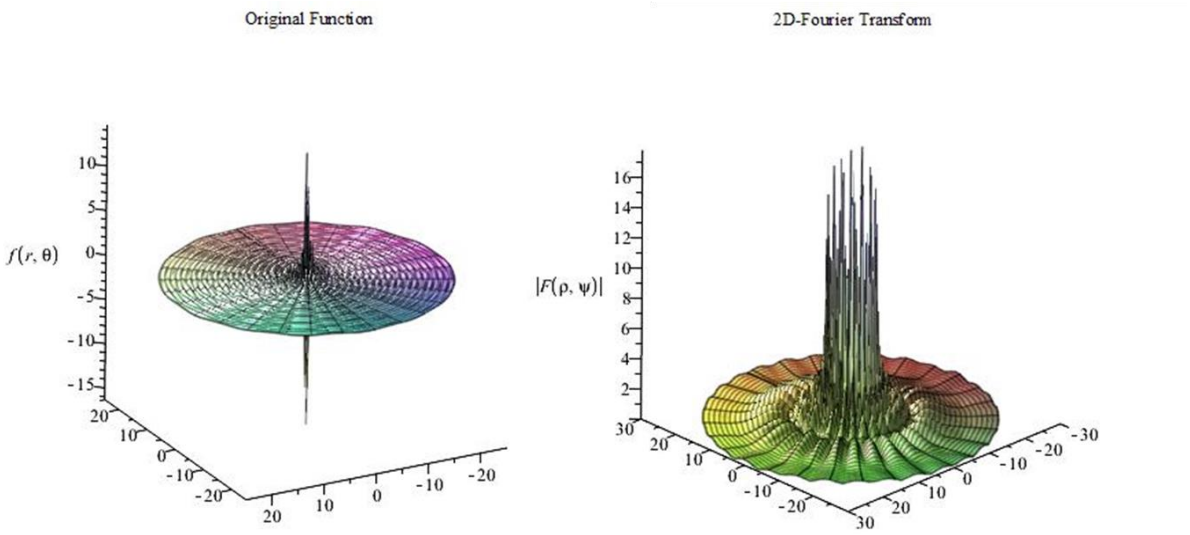

Figure 18 The original function and its 2D-Fourier Transform of 'Four-term sinusoid & Sinc' Function

From Figure 18, the function can be considered as a band-limited function. Therefore equation (14) and (15) were used to implement the forward and inverse transform.

In the angular direction, the highest frequency term in the function in the space domain is $12\sin(15\theta)$. From the sampling theorem, the sampling frequency should be at least twice that of the highest frequency present in the signal. Thus, $N_2=41$ is chosen in order to go a little past the minimum requirement of 31. In the radial direction, from the graphs of the original function and its 2D-FT, it can be assumed that $f(r,\theta)$ is space-limited at $R=15$ and band-limited at $W_\rho=30$. However, since most of the energy in the space domain is located at the origin, a relatively large band limit should be chosen based on the discussion in Section 4.3. Therefore, $W_\rho=90$, $N_1=430$ are chosen.

6.2.2.1 *Forward Transform*

The results for the forward 2D-DFT of Four-term sinusoid & Sinc function with $W_\rho=90$, $N_1=430$ are shown in Figure 19 and Figure 20.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

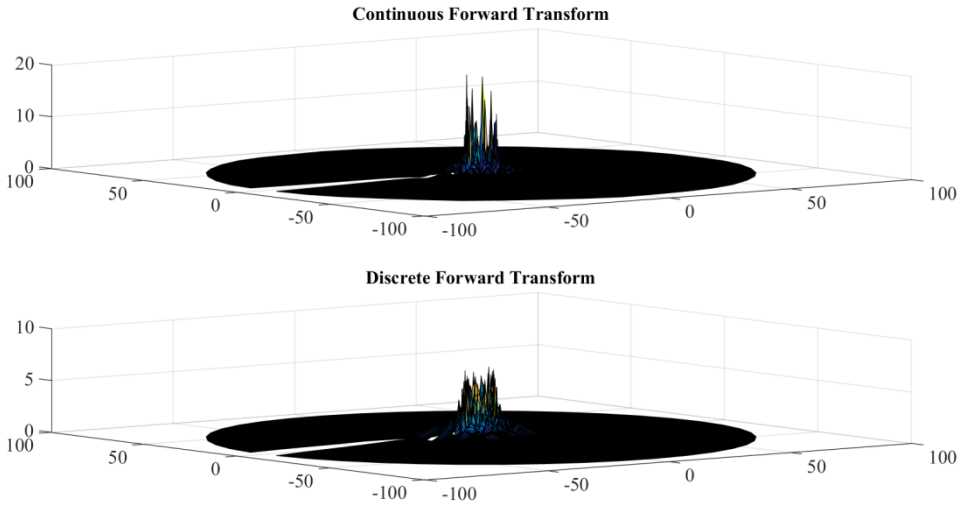

Figure 19 Sampled continuous forward transform and discrete forward transform of 'Four-term sinusoid & Sinc' Function with W_ρ =90,N_1=430, N_2=41

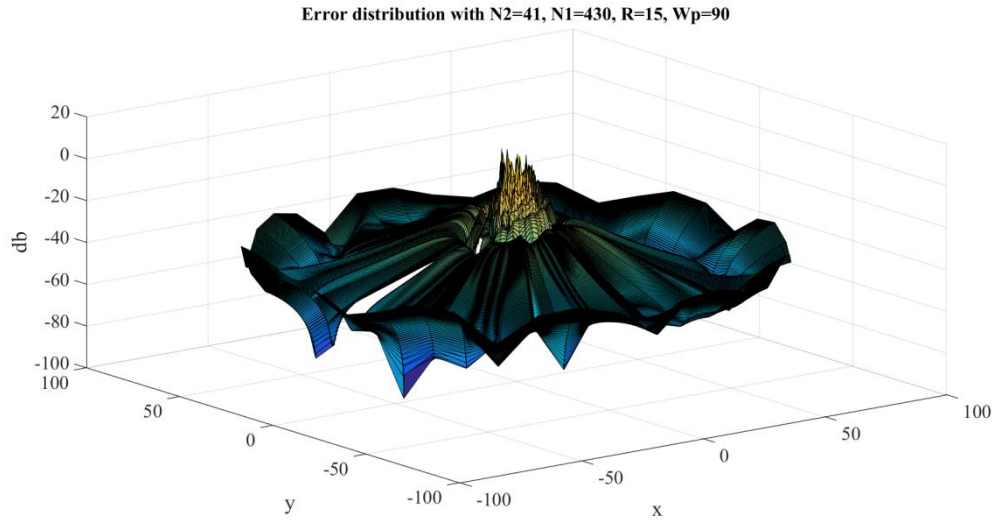

Figure 20 The error distribution of the forward transform of 'Four-term sinusoid & Sinc' Function with W_ρ =90,N_1=430, N_2=41

From Figure 19, the discrete transform does not approximate the continuous transform very well. This is expected because the function in the frequency domain is discontinuous and the sampling points close to the discontinuity will result in a very large error. The maximum value of the error is $Error_{\max} = 10.6535dB$ and this occurs where the discontinuities are located. The average of the error is $Error_{average} = -38.7831dB$.

With $W_\rho = 90$, $N_1 = 430$,Table 5 shows the errors with respect to different value of N_1 and N_2, from which Figure 21 and Figure 22 show the trend.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at *PeerJ Computer Science* **2020**; doi:10.7717/peerj-cs.257

Table 5 Error (dB) of the forward transform of 'Four-term sinusoid & Sinc' Function with different value of N_1 and N_2 of forward transform

N2 \ N1	330	380	430	480	530
11	$E_{max.} = 4.6$ $E_{avg.} = -33.6$	$E_{max.} = 7.1$ $E_{avg.} = -33.4$	$E_{max.} = 3.4$ $E_{avg.} = -33.5$	$E_{max.} = 9.0$ $E_{avg.} = -35.1$	$E_{max.} = 2.8$ $E_{avg.} = -35.5$
21	$E_{max.} = 6.7$ $E_{avg.} = -33.9$	$E_{max.} = 10.5$ $E_{avg.} = -34.6$	$E_{max.} = 3.2$ $E_{avg.} = -37.2$	$E_{max.} = 6.9$ $E_{avg.} = -38.0$	$E_{max.} = 3.6$ $E_{avg.} = -38.1$
41	$E_{max.} = 8.5$ $E_{avg.} = -38.7$	$E_{max.} = 35.1$ $E_{avg.} = -38.9$	$E_{max.} = 10.7$ $E_{avg.} = -38.8$	$E_{max.} = 14.6$ $E_{avg.} = -39.8$	$E_{max.} = 11.1$ $E_{avg.} = -41.3$
81	$E_{max.} = 9.7$ $E_{avg.} = -34.3$	$E_{max.} = 32.7$ $E_{avg.} = 35.5$	$E_{max.} = 14.8$ $E_{avg.} = -36.2$	$E_{max.} = 22.6$ $E_{avg.} = -37.3$	$E_{max.} = 14.5$ $E_{avg.} = -37.5$
161	$E_{max.} = 19.9$ $E_{avg.} = -29.4$	$E_{max.} = 30.2$ $E_{avg.} = -30.7$	$E_{max.} = 22.5$ $E_{avg.} = -31.1$	$E_{max.} = 22.5$ $E_{avg.} = -32.2$	$E_{max.} = 16.1$ $E_{avg.} = -32.8$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

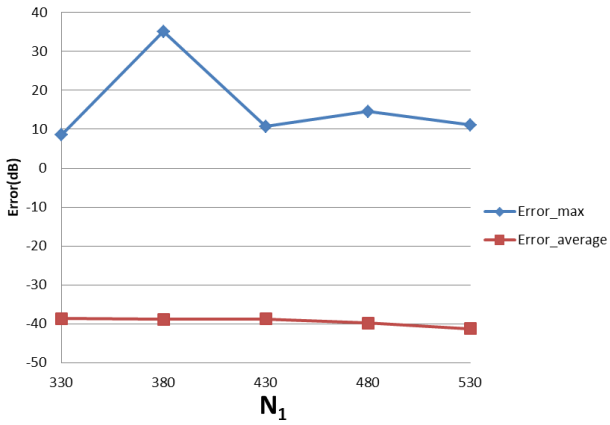

Figure 21 Error of the forward transform of 'Four-term sinusoid & Sinc' Function with fixed N_2 (41) and varying N_1

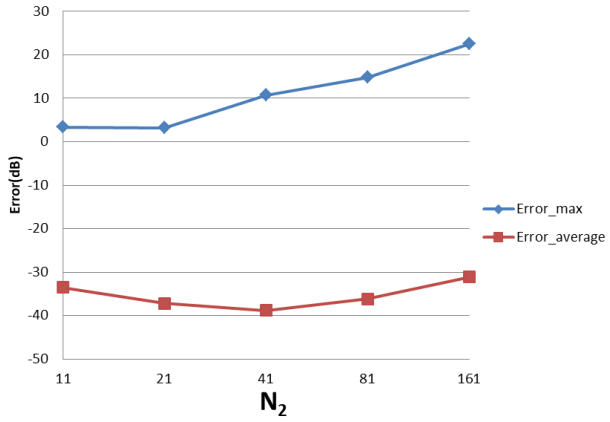

Figure 22 Error of the forward transform of 'Four-term sinusoid & Sinc' Function with fixed N_1 (512) and varying N_2

From Figure 21, increasing N_1 alone tends improve the average error. The maximum error does not change with N_1, which is reasonable because of the discontinuity of the function in the frequency domain.

From Figure 22, increasing N_2 leads to $Error_{\max}$ and $Error_{average}$ first improving and then worsening. This is reasonable because when N_2 is less than the minimum requirement of 31 from sampling theorem, the test result is actually affected by both sampling point density (from the sampling theorem) and grid coverage (discussed in Section 4.3). Increasing N_2 should give better results from the point of view of the sampling theorem but worse grid coverage. The result from the combined effects is dependent on the function properties. In the specific case of this function, when N_2 is bigger than 31, thereby implying that the angular sampling theorem has been satisfied - the results get worse with increasing N_2.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

6.2.2.2 Inverse Transform

The results for the 2D-IDFT of Four-term sinusoid & Sinc function with $W_\rho = 90$, $N_1 = 430$ are shown in Figure 23 and Figure 24.

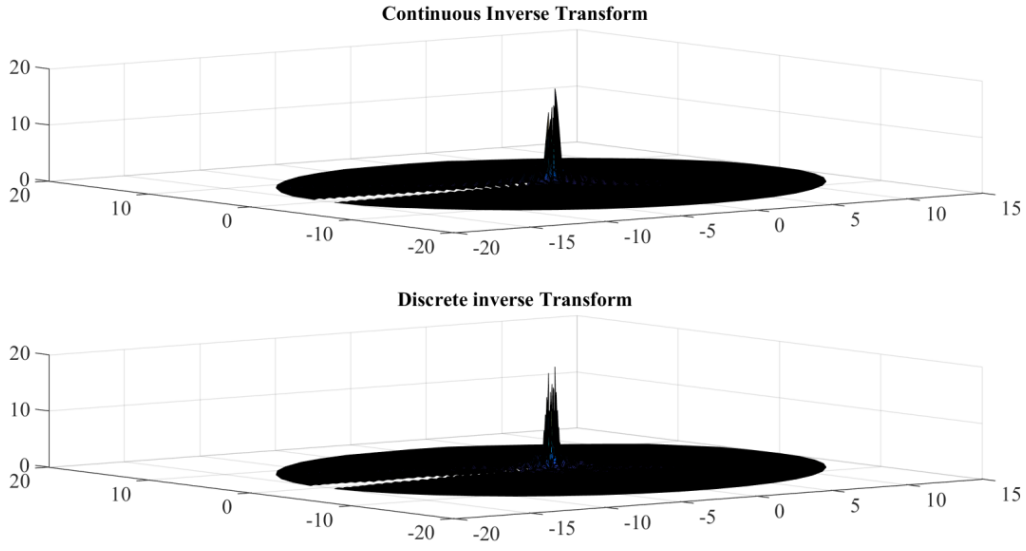

Figure 23 Sampled continuous inverse transform and discrete inverse transform of 'Four-term sinusoid & Sinc' Function with W_ρ =90,N_1=430, N_2=41

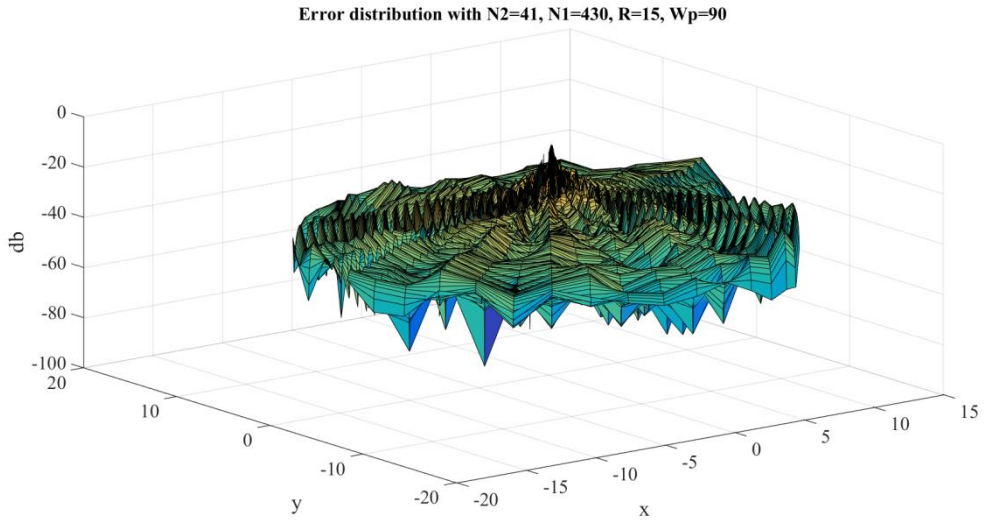

Figure 24 The error distribution of the inverse transform of 'Four-term sinusoid & Sinc' Function with W_ρ =90,N_1=430, N_2=41

The maximum value of the error is $Error_{\max} = -8.6734dB$ .The average of the error is $Error_{average} = -37.8119dB$. With $W_\rho = 90$, $N_1 = 430$, Table 6 shows the errors with respect to different value of N_1 and N_2, from which Figure 25 and Figure 26 show the trend.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at *PeerJ Computer Science* **2020**; doi:10.7717/peerj-cs.257

Table 6 Error (dB) of inverse transform of 'Four-term sinusoid & Sinc' Function with different value of N_1 and N_2

N2 \ N1	330	380	430	480	530
11	$E_{\max.} = 0.1$ $E_{avg.} = -43.7$	$E_{\max.} = 0.1$ $E_{avg.} = -43.7$	$E_{\max.} = 0.1$ $E_{avg.} = -46.6$	$E_{\max.} = 0.1$ $E_{avg.} = -45.6$	$E_{\max.} = 0.1$ $E_{avg.} = -48.1$
21	$E_{\max.} = 0.7$ $E_{avg.} = -38.3$	$E_{\max.} = 0.7$ $E_{avg.} = -38.0$	$E_{\max.} = 0.6$ $E_{avg.} = -40.4$	$E_{\max.} = 0.6$ $E_{avg.} = -40.6$	$E_{\max.} = 0.7$ $E_{avg.} = -42.2$
41	$E_{\max.} = -9.0$ $E_{avg.} = -35.9$	$E_{\max.} = -8.5$ $E_{avg.} = -24.7$	$E_{\max.} = -8.7$ $E_{avg.} = -37.8$	$E_{\max.} = -8.8$ $E_{avg.} = -38.2$	$E_{\max.} = -8.6$ $E_{avg.} = -39.0$
81	$E_{\max.} = -4.5$ $E_{avg.} = -35.7$	$E_{\max.} = -4.7$ $E_{avg.} = -26.5$	$E_{\max.} = -4.5$ $E_{avg.} = -37.5$	$E_{\max.} = -4.6$ $E_{avg.} = -36.2$	$E_{\max.} = -4.5$ $E_{avg.} = -39.0$
161	$E_{\max.} = 0.8$ $E_{avg.} = -35.6$	$E_{\max.} = 0.7$ $E_{avg.} = -32.5$	$E_{\max.} = 0.7$ $E_{avg.} = -36.6$	$E_{\max.} = 0.7$ $E_{avg.} = -37.2$	$E_{\max.} = 0.7$ $E_{avg.} = -39.2$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

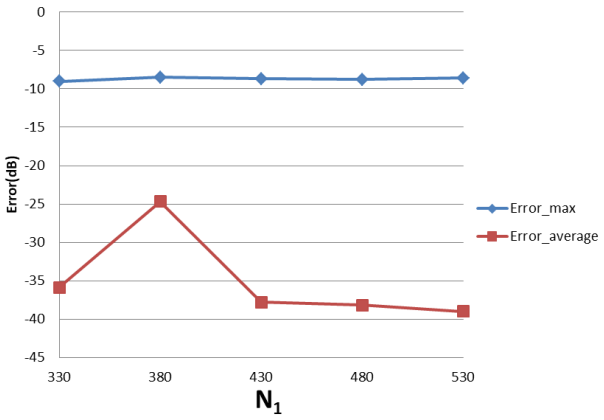

Figure 25 Error of inverse transform of 'Four-term sinusoid & Sinc' Function with fixed N_2 (41) and varying N_1

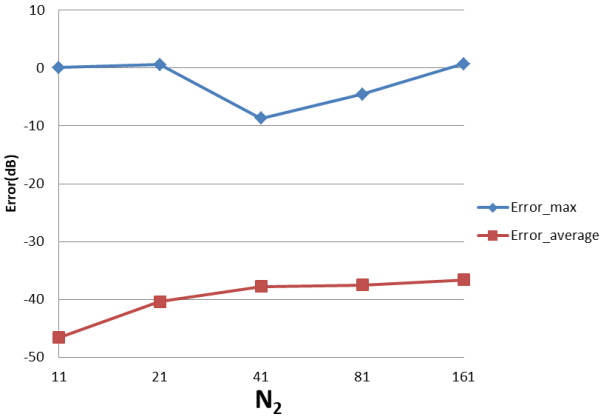

Figure 26 Error of inverse transform of 'Four-term sinusoid & Sinc' Function with fixed N_1 (512) and varying N_2

From Figure 25, it can be observed that the increasing N_1 alone improves the average error, as was expected. However, $N_1 = 380$ gives an apparently worse average error than the other points. This could be caused by the discontinuity of the function in the frequency domain. Changing to $N_1 = 381$, the average error becomes -37.0 dB which proves that the large error is caused by the discontinuity.

From Figure 26, increasing N_2 does not lead to worse results, which is different from previous cases. However, from Figure 18 it can be seen that the function in the frequency domain does not have much information in the center area. So, even though increasing N_2 causes a larger hole in the center as discussed in Section 4.3, it does not lead to losing much energy which explains why Figure 26 shows a different trend from the previous cases.

Performing 2D-DFT and 2D-IDFT sequentially results in $\varepsilon = 1.3117 \times e^{-12}$ where ε is calculated by equation (55) .

6.2.3 Four-term sinusoid and modified exponential

For the next test function, the function is given by

$$f(r,\theta) = \frac{e^{-ar}}{r}[3\sin(\theta) + \sin(3\theta) + 4\cos(10\theta) + 12\sin(15\theta)] \qquad (62)$$

Its continuous 2D-FT can be calculated as

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

$$F(\rho,\psi) = -6\pi i \sin(\psi)\frac{\sqrt{\rho^2+a^2}-a}{\rho\sqrt{\rho^2+a^2}} + 2\pi i \sin(3\psi)\frac{(\sqrt{\rho^2+a^2}-a)^3}{\rho^3\sqrt{\rho^2+a^2}} - 8\pi\cos(10\psi)\frac{(\sqrt{\rho^2+a^2}-a)^{10}}{\rho^{10}\sqrt{\rho^2+a^2}} + 24\pi i \sin(15\psi)\frac{(\sqrt{\rho^2+a^2}-a)^{15}}{\rho^{15}\sqrt{\rho^2+a^2}} \quad (63)$$

The graphs for the original function and the magnitude of its continuous 2D-FT with a=0.1 are shown in Figure 27.

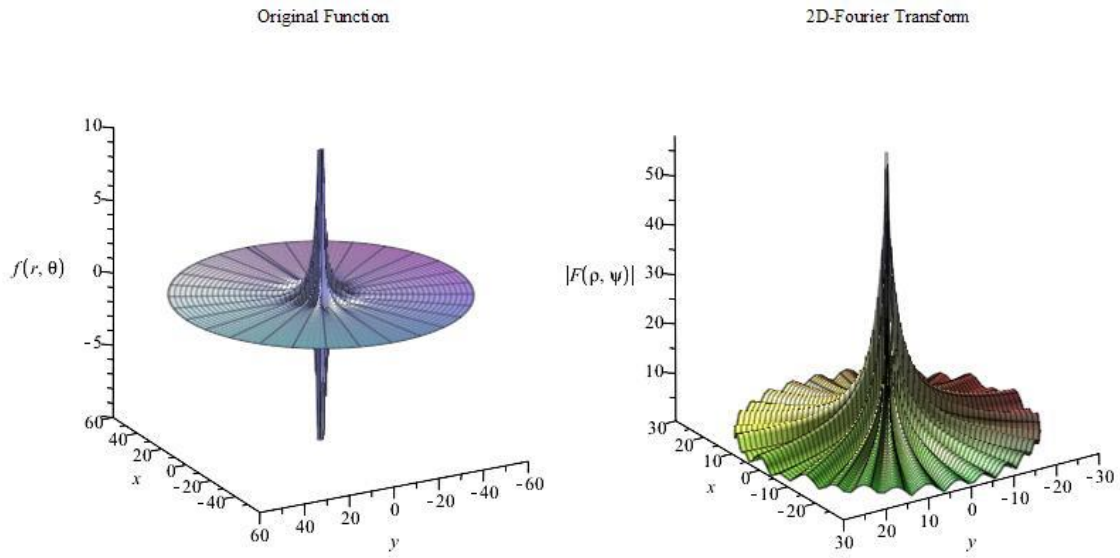

Figure 27The original function and 2D-Fourier Transform of 'Four-term Sinusoid & Modified Exponential' function

From Figure 27, it can be observed that the function has a spike in both domains, which is a more difficult scenario based on the discussion in Section 4.3. In this case, the function can be assumed as space-limited or band-limited. This function will be tested as being space-limited.

From graph of the original function and its 2D-DFT, it can be assumed that $f(r,\theta)$ is effectively space-limited with $R=20$, and $F(\rho,\psi)$ is effectively band-limited with $W_\rho=15$, which gives $j_{0N_1}\approx 300$. This results in $N_1=96$. However, since the function explodes at the center area in both domains, relatively large values of R and W_ρ should give a better approximation. Therefore, another case with $R=40$, $W_\rho=30$ is tested. In this case, $N_1=383$ is chosen.

In the angular direction, the highest frequency term is $12\sin(15\theta)$. From the sampling theorem, the sampling frequency should be at least twice of the highest frequency of signal. Thus, $N_2=41$ is chosen.

6.2.3.1 Forward Transform

Here, the function is tested as a space limited function and equation (12) and (13) are used to proceed with the forward and inverse transform in sequence.

The results with $R=40, W_\rho=30, N_1=383$ are shown in Figure 28 and Figure 29.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

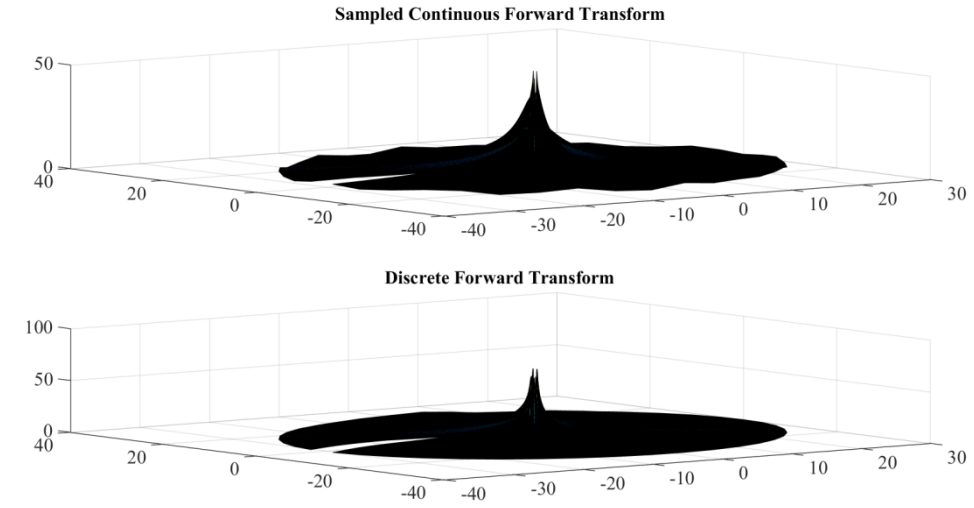

Figure 28 Continuous forward transform and discrete forward transform of 'Four-term Sinusoid & Modified Exponential' function with R=40, W_ρ =30, N_1=383, N_2=41 (test as a space limited function)

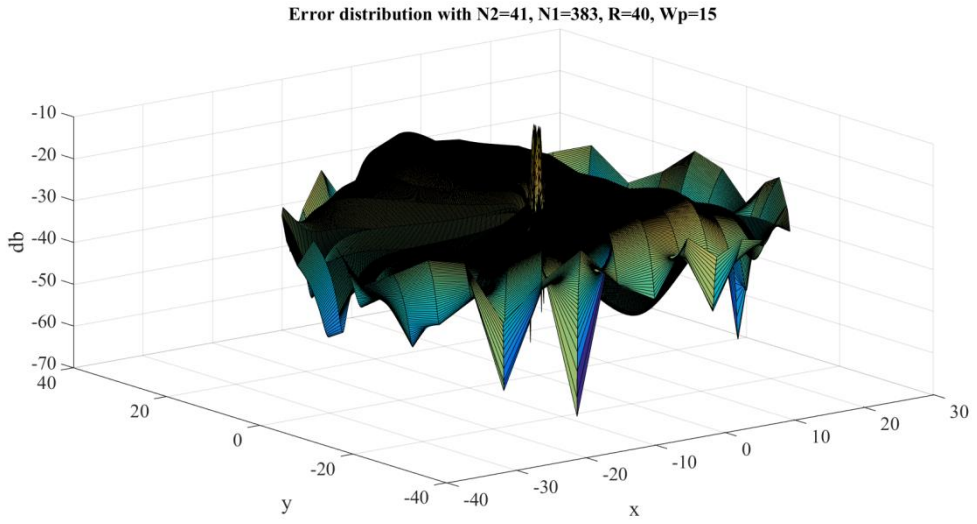

Figure 29 The error distribution of the forward transform of 'Four-term Sinusoid & Modified Exponential' function with R=40, W_ρ =30, N_1=383, N_2=41 (test as a space limited function)

The maximum value of the error is $Error_{\max} = -10.1535dB$ and this occurs at the center area. The average of the error is $Error_{average} = -32.7619dB$. This demonstrates that the discrete function approximates the sampled values of the continuous function quite well.

6.2.3.2 Inverse Transform

The results with $R = 40, W_\rho = 30, N_1 = 383$ are shown in Figure 30 and Figure 31.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

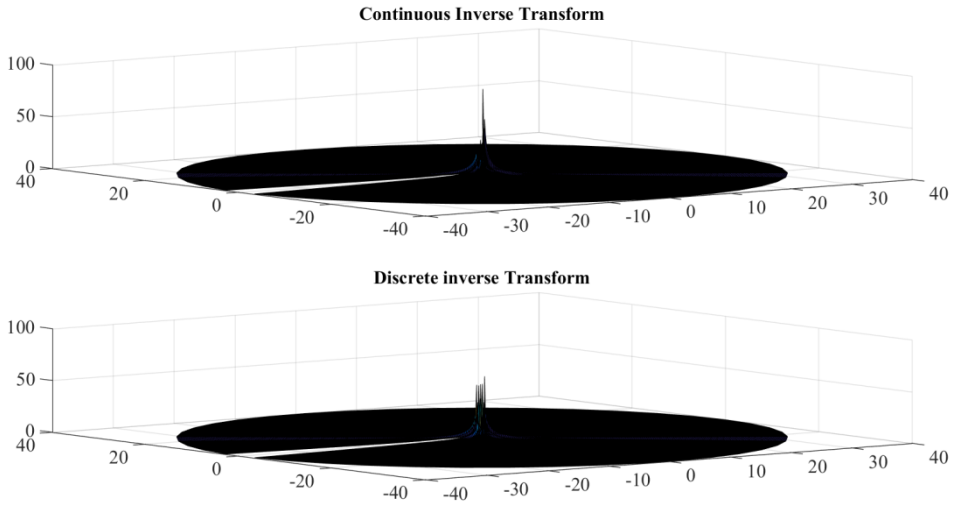

Figure 30 Continuous inverse transform and discrete forward transform of 'Four-term Sinusoid & Modified Exponential' function with R=40, W_ρ =30, N_1=383, N_2=41 (tested as a space limited function)

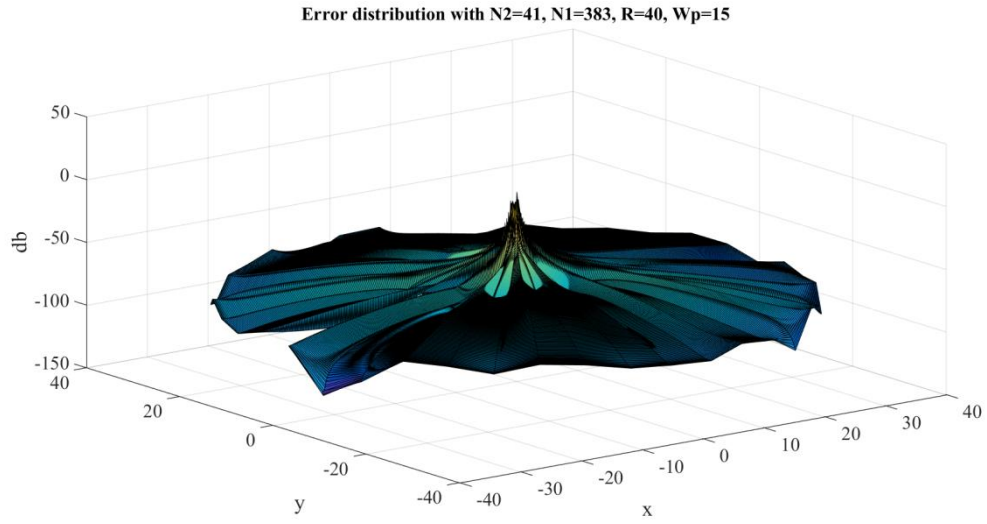

Figure 31 The error distribution of the forward transform of 'Four-term Sinusoid & Modified Exponential' function with R=40, W_ρ =30, N_1=383, N_2=41 (tested as a space limited function)

The maximum value of the error is $Error_{\max} = 0.5579dB$ and this occurs at the center. The average of the error is $Error_{average} = -68.7317dB$.

Performing 2D-DFT and 2D-IDFT results in $\varepsilon = 1.421 \times e^{-12}$, where ε is calculated by equation (55) .

It can be observed that even for functions with the worst properties, the proposed transform can still be used to approximate the continuous Fourier transform with fairly small errors, as long as the function is sampled properly.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

7 Improving the computing time of the transform

One of the advantages of the traditional Fourier transform is that the computing speed is fast by using the now well-established *fft* algorithm. To reduce the computing time of the 2D DFT in polar coordinates, the following steps are taken:

1. Interpreting the transform as three sequential operations (DFT, DHT, IDFT) instead of a single four-dimensional matrix.
2. Pre-calculating and saving the Bessel zeros.

7.1 Reducing computing time by interpreting the transform as three operations in sequence

As explained above, the essence of the transform is that the matrix f_{pk} is transformed into the matrix F_{ql}. The intuitive way to interpret the transform kernel is to think of it as a four-dimensional matrix or matrices in a matrix. However, interpreting the transform as a 1D-DFT of each column, a 1D-DHT of each row and then a 1D-IDFT of each column makes it possible to use the Matlab built in functions *fft* and *ifft*, which significantly reduced the computational time.

7.2 Reduce computing time by pre-calculating the Bessel Zeros

After defining the transform as three operations in sequence and using the matrix for the discrete Hankel transform defined in [8], it was found that a lot of computational time was used to calculate the Bessel zeros for every different test case, even though the Bessel zeros are the same in every case. Pre-calculating the Bessel zeros and storing the results for large numbers of N_1 and N_2 saves a lot of time.

Table 7 shows the computing time of a forward transform on the same computer (Processor: Intel(R) Core(TM) i7-4710HQ CPU, RAM:12GB) for three cases:

1. Evaluate the transform as matrices in a matrix without pre-calculating the Bessel zeros.
2. Evaluate the transform as a DFT, DHT and IDFT in sequence without pre-calculating the Bessel zeros.
3. Evaluate the transform as a DFT, DHT and IDFT in sequence with pre-calculating the Bessel zeros.

The Gaussian function was used as the test function therefore N_1=383 and N_2=15.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

Table 7 Computing time of three cases: Case1: Run the transform as matrixes in matrix without pre-calculating the Bessel zeros; Case2: Run the transform as DFT,DHT and IDFT in sequence without pre-calculating the Bessel zeros; Case3: Run the transform as DFT,DHT and IDFT in sequence with pre-calculating the Bessel zeros

Test cases	Total computing time(seconds)
Case 1	3346.5
Case 2	321.1
Case 3	14.3

From Table 7, it can be clearly observed that the computing time by running the transform as matrices in a matrix costs 3346.5s, which is not acceptable or the transform to be useful. Testing the transform as three operations turns 3346.5 seconds into 321.1 seconds. This is much better. Finally, pre-calculating the Bessel Zeros makes the transform much faster and applicable.

8 Summary and Conclusion

8.1 Accuracy and Precision of the transform

The proposed discrete 2D-Fourier Transform in polar coordinates demonstrates an acceptable accuracy in providing discrete estimates to the continuous Fourier transform in polar coordinates. In [8],[11]and[15], the one dimensional Hankel transform of a sinc function showed similar dynamic error, which could be used as a comparative measure. Since the discrete Hankel transform is one step of the proposed discrete 2D-Fourier Transform, and the definition of the Hankel transform is based on [8], a similar dynamic error should be expected.

The test cases showed that the transform introduced very small errors ($\varepsilon = 1.4004 \times e^{-12}$ for worst case) by performing a forward transform and an inverse transform sequentially, which demonstrates that the algorithm shows good precision.

8.2 Guidelines of choosing sample size

As discussed in Section 4.3 and proved by test cases, the sample size N_1 (sample size in the radial direction) and N_2 (sample size in the angular direction) do not have to be of the same order. For functions with different properties, sample size in the different directions could be very different. To approximate the continuous Fourier transform properly, sample size should be chosen based on the discussion in Section 4.3.

8.3 Interpretation of the transform

By interpreting the transform as a 1D Fourier transform, 1D Hankel transform and 1D inverse Fourier transform, the computing time of the transform is improved to a useful level.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

References

[1] J. W. Cooley and J. W. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series," *Math. Comput.*, vol. 19, no. 90, pp. 297–301, 1965.

[2] Y. Xu, D. Feng, and L. V. Wang, "Exact frequency-domain reconstruction for thermoacoustic tomography. I. Planar geometry," *Med. Imaging IEEE Trans. On*, vol. 21, no. 7, pp. 823–828, 2002.

[3] M. C. Scott *et al.*, "Electron tomography at 2.4-ångström resolution," *Nature*, vol. 483, no. 7390, p. 444, Mar. 2012.

[4] B. P. Fahimian *et al.*, "Radiation dose reduction in medical x-ray CT via Fourier-based iterative reconstruction," *Med. Phys.*, vol. 40, no. 3, Mar. 2013.

[5] E. Lee *et al.*, "Radiation dose reduction and image enhancement in biological imaging through equally-sloped tomography," *J. Struct. Biol.*, vol. 164, no. 2, pp. 221–227, 2008.

[6] N. Baddour, "Discrete Two Dimensional Fourier Transform in Polar Coordinates Part I: Theory and Operational Rules," *Preprints*, vol. 2019, no. 2019070151, Jul. 2019.

[7] C. E. Shannon, "Communication in the presence of noise," *Proc. IEEE*, vol. 72, no. 9, pp. 1192–1201, 1984.

[8] N. Baddour and U. Chouinard, "Theory and operational rules for the discrete Hankel transform," *JOSA A*, vol. 32, no. 4, pp. 611–622, Apr. 2015.

[9] D. W. Lozier, "NIST Digital Library of Mathematical Functions," *Ann. Math. Artif. Intell.*, vol. 38, no. 1–3, pp. 105–119, 2003.

[10] N. Baddour, "The Discrete Hankel Transform," in *Fourier Transforms - Century of Digitalization and Increasing Expectations*, London, UK: IntechOpen, 2019.

[11] M. Guizar-Sicairos and J. C. Gutiérrez-Vega, "Computation of quasi-discrete Hankel transforms of integer order for propagating optical wave fields," *JOSA A*, vol. 21, no. 1, pp. 53–58, Jan. 2004.

[12] N. Baddour, "Two-Dimensional Fourier Transforms in Polar Coordinates," *Adv. Imaging Electron Phys.*, vol. 165, pp. 1–45, Jan. 2011.

[13] N. Baddour, "Operational and convolution properties of two-dimensional Fourier transforms in polar coordinates," *J. Opt. Soc. Am. A*, vol. 26, no. 8, pp. 1767–1777, Aug. 2009.

[14] A. D. Poularikas, *Transforms and Applications Handbook, Third Edition*. CRC Press, 2010.

[15] W. Higgins and Jr. Munson D., "An algorithm for computing general integer-order Hankel transforms," *Acoust. Speech Signal Process. IEEE Trans. On*, vol. 35, no. 1, pp. 86–97, 1987.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

Appendix A –Matlab Code

A-1. Theta matrix for space limited function

```
% N1 sample size in radial direction
% N2 sample size in angular direction
function theta=thetamatrix_SpaceLimited(N2,N1)
theta=zeros(N2,N1-1);
M=(N2-1)/2;
for ii=1:N2;
    p=ii-1-M;
    for k=1:N1-1;
        theta(ii,k)=(p/N2)*2*pi;
    end
end
```

A-2. r matrix for space limited function

```
% N1 sample size in radial direction
% N2 sample size in angular direction
% R effective space limit
% zeromatrix precalculated Bessel zero
function r=rmatrix_SpaceLimited(N2,N1,R,zeromatrix)
M=(N2-1)/2;
for ii=1:N2;
    p=ii-1-M;
    for k=1:N1-1;
        zero2=zeromatrix(5001-abs(p),:);
        jpk=zero2(k);
        jpN1=zero2(N1);
        r(ii,k)=(jpk/jpN1)*R;
    end
end
```

A-3. Psi matrix for space limited function

```
% N1 sample size in radial direction
% N2 sample size in angular direction
function psi=psimatrix_SpaceLimited(N2,N1)
psi=zeros(N2,N1-1);
M=(N2-1)/2;
for ii=1:N2;
    q=ii-1-M;
    for l=1:N1-1;
        psi(ii,l)=(q/N2)*2*pi;
    end
end
```

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

A-4. Rho matrix for space limited function

```
% N1 sample size in radial direction
% N2 sample size in angular direction
% R effective space limit
% zeromatrix precalculated Bessel zero
function rho=rhomatrix_SpaceLimited(N2,N1,R,zeromatrix)
M=(N2-1)/2;
for ii=1:N2;
    q=ii-1-M;
    for l=1:N1-1;
        zero2=zeromatrix(5001-abs(q),:);
        jql=zero2(l);
        rho(ii,l)=jql/R;
    end
end
```

A-5. Y matrix Assembly Function

```
% Y is the N-1 x N-1 transformation matrix to be assembled
% n is the order of the bessel function
% N is the size of the transformation matrix
%zeros are the bessel zeros passed to the function

function Y = YmatrixAssembly(n,N,zero)
%tic


for l=1:N-1

    for k=1:N-1

        jnk=zero(k);
        jnl=zero(l);
        jnN=zero(N);
        jnplus1=besselj(n+1, jnk);

        Y(l,k)=(2*besselj(n,(jnk*jnl/jnN)))/(jnN*jnplus1^2);


    end
end

%toc

end
```

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

A-5. Forward transform of Gaussian function

```
N2=15; %number of sample points in angular direction
N1=383; %number of sample points in radial direction
M=(N2-1)/2; %highest order of bessel function
R=40;% space limit
Wp=30; % band limit
a=0.1;
load('zeromatrix.mat')
theta=thetamatrix_SpaceLimited(N2,N1); %Sample point in angular direction in space domain.
r=rmatrix_SpaceLimited(N2,N1,R,zeromatrix);%Sample point in radial direction in space domain.
psi=psimatrix_SpaceLimited(N2,N1);%Sample point in angular direction in frequency domain.
rho=rhomatrix_SpaceLimited(N2,N1,R,zeromatrix);%Sample point in radial direction in frequency
domain.
[x,y]=pol2cart(theta,r); %sample points in Cartesian coordinates in space domain
[x1,y1]=pol2cart(psi,rho); %sample points in Cartesian coordinates in frequency domain


%Discretizing the function
for ii=1:N2
    for jj=1:N1-1
        f(ii,jj)=exp(-r(ii,jj)^2);
    end
end
% DFT
fnk=circshift(fft(circshift(f,M+1,1),N2,1),-(M+1),1);
% DHT
for n=-M:M
    ii=n+M+1;
    zero2=zeromatrix(5001-abs(n),:);
    jnN1=zero2(N1);
    if n<0
    Y=((-1)^abs(n))*YmatrixAssembly(abs(n),N1,zero2);
    else
    Y=YmatrixAssembly(abs(n),N1,zero2);
    end
    fnl(ii,:)=(Y*fnk(ii,:)')';
    Fnl(ii,:)=fnl(ii,:)*(2*pi*(i^(-n)))*(R^2/jnN1);
end
% IDFT
TwoDFT=circshift(ifft(circshift(Fnl,M+1,1),N2,1),-(M+1),1);
%creating a discrete 2D Fourier transform
for ii=1:N2
    for jj=1:N1-1
        trueFunc(ii,jj)=pi*exp((-rho(ii,jj)^2)/4);
    end
end

%calculating the dynamic error from transform and true function
error= 20*log10(abs(trueFunc- TwoDFT)/max(max(abs(TwoDFT))));

figure(1)
subplot(2,1,1)
surf(x1,y1,abs(trueFunc))
title('\fontsize{24}Sampled Continuous Forward Transform')
subplot(2,1,2)
surf(x1,y1,abs(TwoDFT))
title('\fontsize{24}Discrete Forward Transform')
```

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

```
figure(2)

surf(x1,y1,error)
xlabel('x');
ylabel('y');
zlabel('db')
str=sprintf('Error distribution with N2 = %d, N1 = %d,R= %d, a= %d ', N2,N1,R,a);
title(['\fontsize{24}Error distribution with N2=',num2str(N2),', N1=',num2str(N1),',
R=',num2str(R), ', Wp=',num2str(Wp)]);

mean1=mean(mean(error)); % Average dynamic error
max1=max(max(error)); % Maximum dynamic error
```

A-6. Inverse transform of Gaussian function

```
N2=15 ; %number of sample points in angular direction
N1=383; %number of sample points in radial direction
M=(N2-1)/2; %highest order of bessel function
R=40;% space limit
Wp=30; % band limit
a=0.1;
load('zeromatrix.mat')
theta=thetamatrix_SpaceLimited(N2,N1);%Sample point in angular direction in space domain.
r=rmatrix_SpaceLimited(N2,N1,R,zeromatrix);%Sample point in radial direction in space domain.
psi=psimatrix_SpaceLimited(N2,N1);%Sample point in angular direction in frequency domain.
rho=rhomatrix_SpaceLimited(N2,N1,R,zeromatrix);%Sample point in radial direction in frequency
domain.
[x,y]=pol2cart(theta,r); %sample points in Cartesian coordinates in space domain
[x1,y1]=pol2cart(psi,rho); %sample points in Cartesian coordinates in frequency domain

%creating a discrete true function
for ii=1:N2
    for jj=1:N1-1
       trueFunc(ii,jj)=pi*exp((-rho(ii,jj)^2)/4);
    end
end
% DFT
FNL=circshift(fft(circshift(trueFunc,M+1,1),N2,1),-(M+1),1);
% DHT
for n=-M:M
    ii=n+M+1;
    zero2=zeromatrix(5001-abs(n),:);
    jnN1=zero2(N1);
    if n<0
    Y=((-1)^abs(n))*YmatrixAssembly(abs(n),N1,zero2);
    else
    Y=YmatrixAssembly(abs(n),N1,zero2);
    end
    Y0=Y';
    Fnk(ii,:)=FNL(ii,:)*Y0;
    fnk(ii,:)=Fnk(ii,:)*((jnN1)*(j^n))/(2*pi*R^2);

end
% IDFT
TwoDIFT=circshift(ifft(circshift(fnk,M+1,1),N2,1),-(M+1),1);
```

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2019 doi:10.20944/preprints201907.0189.v1

Peer-reviewed version available at PeerJ Computer Science 2020; doi:10.7717/peerj-cs.257

```
%%discretizing the function in space domain
for ii=1:N2
    for jj=1:N1-1
        f(ii,jj)=exp(-r(ii,jj)^2);
    end
end
%calculating the dynamic error from transform and origal function
error= 20*log10(abs(f- TwoDIFT)/max(max(abs(TwoDIFT))));

figure(1)
subplot(2,1,1)
surf(x,y,abs(f))
title('\fontsize{24}Continuous Inverse Transform')
subplot(2,1,2)
surf(x,y,abs(TwoDIFT))
title('\fontsize{24}Discrete inverse Transform')

figure(2)
surf(x,y,error)
xlabel('x');
ylabel('y');
zlabel('db')
str=sprintf('Error distribution with N2 = %d, N1 = %d,R= %d, a= %d ', N2,N1,R,a);
title(['\fontsize{24}Error distribution with N2=',num2str(N2),', N1=',num2str(N1),',
R=',num2str(R), ', Wp=',num2str(Wp)]);


mean=mean(mean(error)); % Average dynamic error
max=max(max(error));% Maximum dynamic error
```