Topic 3: Digital Sampling

3.1 Digital Representation of Images

To represent an continuous image in a digital for it must be sampled, or measured, at regular intervals to form a two-dimensional array of numbers being the intensity at the sampled points as shown in figure 1.

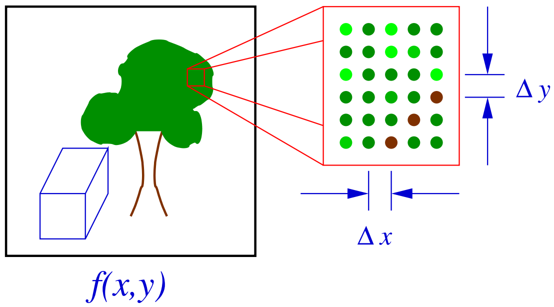


Figure 1: Sampled region of a continuous image.

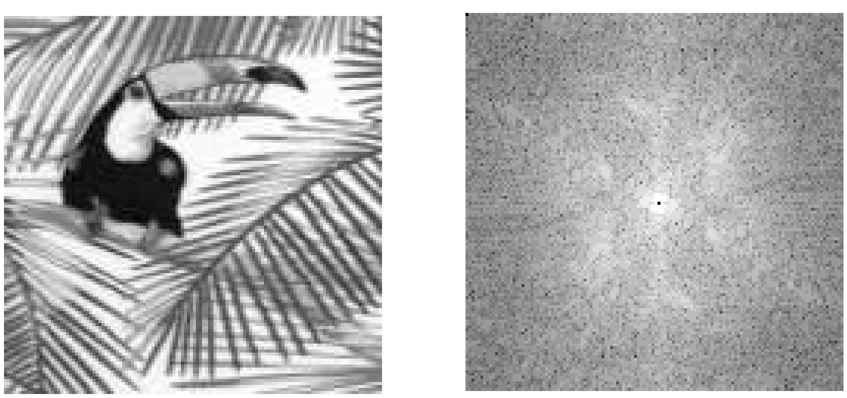
For an image $f(x,y)$, if the *top/left* is located at (x_0, y_0) then if we take $N \times N$ samples, these are located at,

$$(x_0 + i\Delta x,\, y_0 + j\Delta y) \quad \text{where } i \;\&\; j = 0, 1, \ldots, N-1$$

where Δx and Δy are the x and y sampling intervals. This gives $N \times N$ array of samples, or numbers,

$$f(i,j) \quad \text{where } i \;\&\; j = 0, 1, \ldots, N-1$$

which we will hold in a computer as a two-dimensional array with each sampled point being known as a *pixel*. A typical digital image is shown in figure 2 (a), being a 128×128[1] pixel image.



(a) (b)

Figure 2: (a) A typical 128×128 pixel monochrome image, and (b) the digital Fourier transform of the same image.

[1]The use of the rather odd number 128 will be explained shortly.

Clearly if the sampling distance $(\Delta x, \Delta y)$ is *sufficiency small* and so there are a *sufficiently large* number of pixels, we will get an accurate representation of the original image. We will consider way we mean by an accurate representation and develop a criteria for the sampling intervals later on in this section.

The two-dimensional Fourier transform of the image $f(x,y)$ is given by

$$F(u,v) = \iint f(x,y)\ \exp(-2\pi\imath(ux+uy))\,\mathrm{d}x\,\mathrm{d}y$$

Similarly, the Fourier transform $F(u,v)$ can be sampled at intervals of Δu and Δv to give a two-dimensional array of

$$F(k,l) \quad \text{where } k \,\&\, l = 0,1,\ldots,N-1$$

where, *it will be shown*[2], that for optimal sampling we have that

$$\Delta u = \frac{1}{N\Delta x} \quad \text{and} \quad \Delta v = \frac{1}{N\Delta y} \tag{1}$$

In Fourier space the samples will be complex, so we will not usually be able to view $F(k,l)$ directly, but it is more typical to display $|F(k,l)|^2$ as shown in figure 2 (b). We will also shortly be shown the relation between

$$f(i,j) \Leftrightarrow F(k,l)$$

so allowing us to numerically calculate $F(k,l)$ from $f(i,j)$.

3.2 Discrete Fourier Transform

In one-dimensions, if we have a continuous function $f(x)$ then its Fourier Transform is given by,

$$F(u) = \int f(x)\ \exp(-\imath 2\pi ux)\,\mathrm{d}x$$

If we have a sampled function , $f(i)$, with N samples, the we can define is *Discrete Fourier Transform* (DFT), as begin given by:

$$F(k) = \sum_{i=0}^{N-1} f(i)\ \exp\left(-\imath 2\pi \frac{ki}{N}\right)$$

and the *inverse Discrete Fourier Transform* being given by

$$f(i) = \frac{1}{N}\sum_{k=0}^{N-1} F(k)\ \exp\left(\imath 2\pi \frac{ki}{N}\right)$$

where, by convention, the normalisation by $1/N$ is applied to the inverse transform[3].

Similarly in two dimensions, for a continuous function $f(x,y)$ its Fourier transform is given by

$$F(u,v) = \iint f(x,y)\exp(-\imath 2\pi(ux+vy))\,\mathrm{d}x\,\mathrm{d}y$$

[2] later in this section.

[3] The DFT can also be defined with $1/\sqrt{N}$ normalisation on both forward and inverse transforms.

If we sample this function on a regular grid of $N \times N$ samples, we get a sampled function $f(i,j)$, then the two-dimensional *Discrete Fourier Transform* is given by,

$$F(k,l) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} f(i,j)\,\exp\left(-\imath 2\pi\left(\frac{ki}{N}+\frac{lj}{N}\right)\right) \tag{2}$$

and similarly the *inverse Discrete Fourier Transform* being given by,

$$f(i,j) = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} F(k,l)\,\exp\left(\imath 2\pi\left(\frac{ki}{N}+\frac{lj}{N}\right)\right) \tag{3}$$

where we have assumed that the sampled image is square[4] and we have applied the normalisation to the inverse tranform only.

The gives the numerical relation between $f(i,j)$ and $F(k,l)$, but not the relation between the sampling rate in the two spaces. This will be considered after some of the properties of *Discrete Fourier Transform*.

3.3 Properties of the DFT

The DFT, being the discrete version of the continuous Fourier transform exhibits all the properties detailed in the accompanying booklet[5], but we need to consider some of the particular properties of the discrete version. Consider fist a one-dimensional sampled function $f(i)$, where the samples are *Real Only*, then we can write

$$F(k) = F_R(k) - \imath F_I(k)$$

where we have that

$$\begin{aligned} F_R(k) &= \sum_{i=0}^{N-1} f(i)\,\cos\left(-\imath 2\pi\frac{ki}{N}\right) \\ F_I(k) &= \sum_{i=0}^{N-1} f(i)\sin\left(-\imath 2\pi\frac{ki}{N}\right) \end{aligned}$$

As shown in figure 3, cos() is a symmetric function while sin() is a anti-symmetric function, so since $F_R()$ and $F_I()$ are summations of cos() and sin() respectively, they while also display the same symmetries, so that

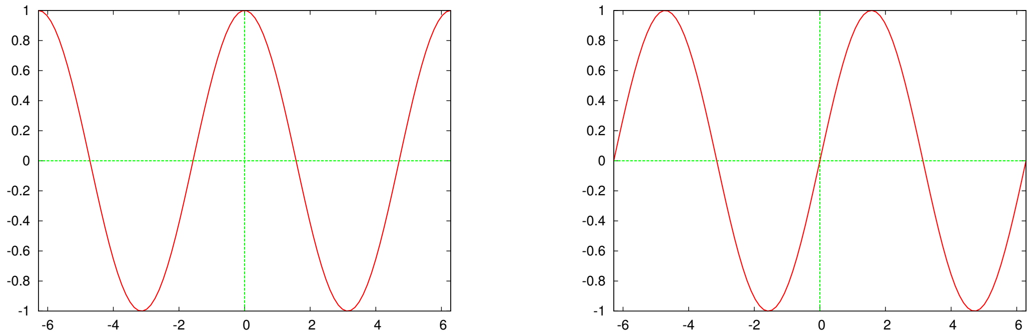
$$\begin{array}{rcll} F_R(k) & \Rightarrow & \text{Symmetric Function} & \Rightarrow \quad F_R(-k) = F_R(k) \\ F_I(k) & \Rightarrow & \text{Anti-symmetric Function} & \Rightarrow \quad F_I(-k) = -F_I(k) \end{array}$$

If we consider the exp() part of the expression for the one-dimensional DFT we also see that,

$$\exp\left(-\imath 2\pi\frac{(k\pm nN)i}{N}\right) = \exp\left(-\imath 2\pi\frac{ki}{N}\right)$$

[4] If the image is rectangular of size $M \times N$, all the formulas stall apply, but it makes the interpretation rather more complex.

[5] The Fourier Transform (what you need to know)



(a) (b)

Figure 3: Symmetry properties of(a) cos() and (b) sin() functions

which implies that $F(k)$ is *cyclic* of period N, so that

$$F(k \pm nN) = F(k)$$

So that k does *not* need to run from $0 \rightarrow N-1$, but *any* range of N consecutive sample fully specify specify $F(k)$, so noting that N is always *even*[6], then we can typically take $F(k)$ specified over the range,

$$F(k) \quad \text{for } k = -\frac{N}{2}, \ldots, 0 \ldots, \frac{N}{2} - 1$$

as shown in figure 4.

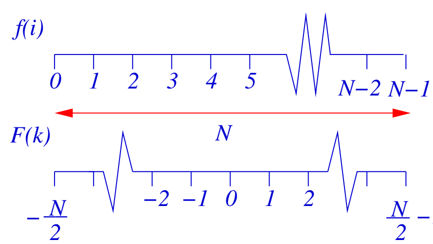


Figure 4: Range of a one-dimensional DFT.

Now consider the symmetry conditions, initially wit a simple example where $N = 4$, so we have four samples, being

$$f(i) \quad \text{for } i = 0, 1, 2, 3$$

so in Fourier space we have

$$F_R(k) - \imath F_I(k) \quad \text{for } k = -2, -1, 0, 1$$

which consists of 4 *real components* and 4 *imaginary components*, which initially looks as if we have doubled the amount of data. However look at this in detail and we see that for the *real component* we have,

$F_R(0) = \sum_{i=0}^{3} f(i)$	The DC term
$F_R(-1) = F_R(1)$	Symmetric property
$F_R(-2) = F_R(2)$	cyclic of period 4

[6]Requirement for the calculation of the DFT by the FFT algorithm, see below

so we have *three* of the *real components* that depend on $f(i)$ and the *one* that is given by the symmetry condition. For the *imaginary component* we similarly have that,

$$\begin{array}{ll} F_I(0) = 0 & \text{Since } \sin 0 = 0 \\ F_I(-1) = -F_I(1) & \text{Antisymmetric property} \\ F_I(-2) = 0 & \text{Since } \sin\pi = 0 \end{array}$$

so here we have only *one* of the *imaginary component* depending on $f(i)$, *one* given by the anti-symmetry, and the other *two* always being zero. This gives a total of *4* components in Fourier space that depend on the input data with the other *4* being given by the symmetry properties[7].

If we consider a larger example with 16 data points, where $f(i)$ is shown in figure 5. The modulus of $F(k)$ is shown in figure 6, with (a) with k over the range $0 \rightarrow 15$ and (b) over the range $-8 \rightarrow 7$. Both Fourier Transforms show the expected symmetry, but it is easier to see

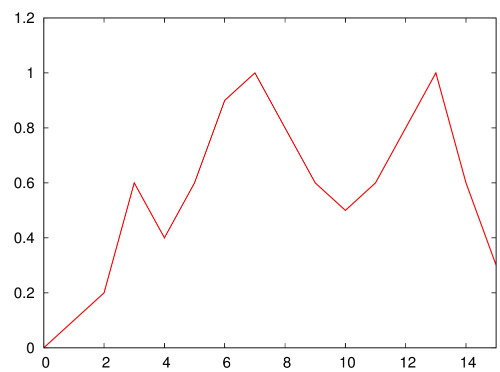


Figure 5: A 16 point input data samples.

and understand in the shifted version.

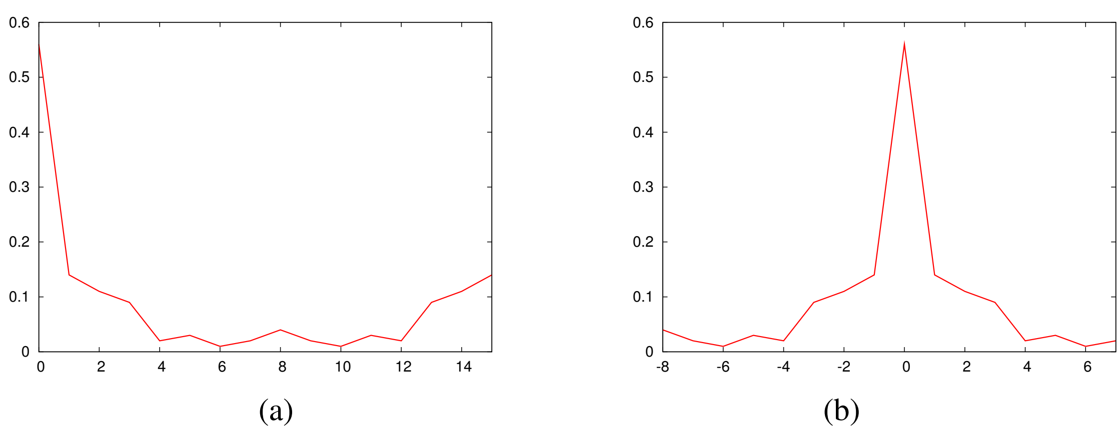


Figure 6: Modulus of the Fourier transform with (a) range $0 \rightarrow 15$ and (b) range $-8 \rightarrow 7$.

We can extend this to the general case of a N point real function $f(i)$, then its *Discrete Fourier Transform*, $F(k)$ will have:

$$\frac{N}{2}+1 \qquad \text{Real Value that depend on } f(i)$$

[7]The Fourier transform is unitary, so we expect the same number of values in each space.

$$
\begin{aligned}
\frac{N}{2} - 1 & \qquad \text{Imaginary Values depend of } f(i) \\
F_I(0) &= F_I(-N/2) = 0
\end{aligned}
$$

giving a *total* of N independent values in Fourier space with the other values given by symmetry properties. So there is the same number of data point in both real and Fourier space. This is useful when calculating DFT, allowing to use the same storage for real and Fourier space arrays.

Properties of the two-dimensional DFT

In two-dimensions things are little more complicated but follows the same basic pattern. Noting the $\exp()$ term in the two-dimensional expression, then for a transform $F(k,l)$ of size $N \times N$, then it will be cyclic of period N in both the k and l directions, so we have that.

$$F(k \pm nN, l \pm mN) = F(k,l)$$

so we can shift the $F(0,0)$ term in two dimensions to give,

$$F(k,l) \quad \text{for } k \,\&\, l = -\tfrac{N}{2}, \ldots, 0 \ldots, \tfrac{N}{2} - 1$$

as shown in figure 7, and has been using to display the modulus squared[8] of the Fourier transform in figure 2 (b). The $|F(u,v)|^2$ is identical to the Optical Diffraction pattern. Usually displayed with the *bright* centre in the middle.

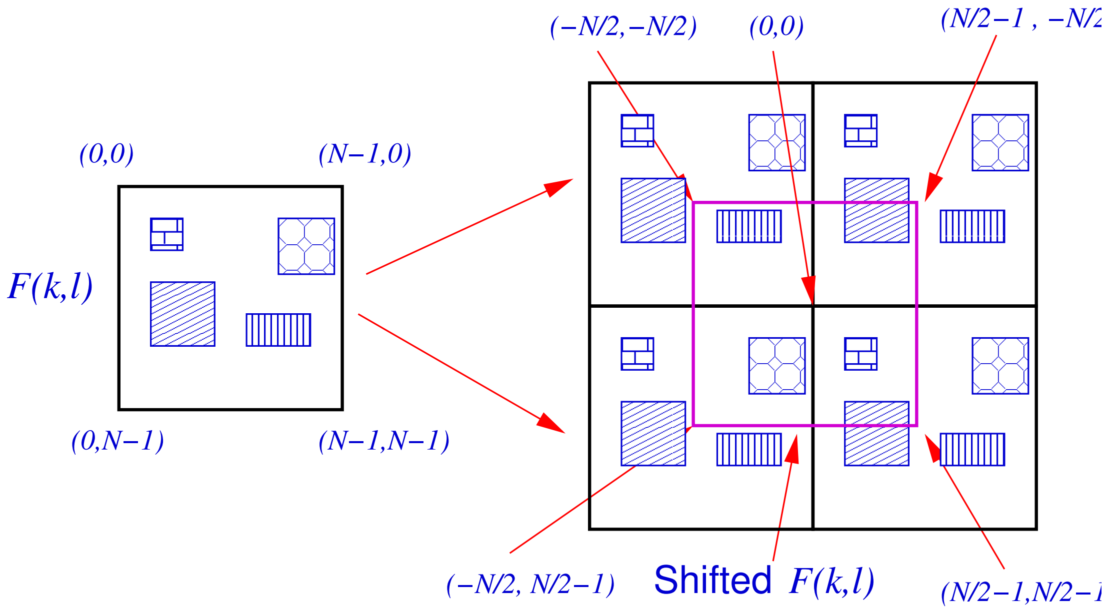


Figure 7: Range of the Fourier Transform of and $N \times N$ image shifted so that $F(0,0$ is at the centre.

For a real sampled input image $f(i,j)$ again we can write the Fourier Transform

$$F(k,l) = F_R(k,l) - \imath F_I(k,l)$$

[8] that is actually displayed is $\log(|F(k,l)|^2 + 1)$ to reduce the dynamic range.

where, after some effort, we get that, we can expand the expression for the Fourier Transform to get that,

$$F_R(k,l) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} f(i,j)\left(\cos\left(2\pi\frac{ik}{N}\right)\cos\left(2\pi\frac{jl}{N}\right)+\sin\left(2\pi\frac{ik}{N}\right)\sin\left(2\pi\frac{jl}{N}\right)\right)$$

and that

$$F_I(k,l) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} f(i,j)\left(\cos\left(2\pi\frac{ik}{N}\right)\sin\left(2\pi\frac{jl}{N}\right)+\sin\left(2\pi\frac{ik}{N}\right)\cos\left(2\pi\frac{jl}{N}\right)\right)$$

Now we have that $\cos()$ is symmetric, and $\sin()$ is anti-symmetric, while

$$\begin{array}{rll} F_R() & \text{has form} & \cos()\cos()+\sin()\sin() \\ F_I() & \text{has form} & \cos()\sin()+\sin()\cos() \end{array}$$

so about the centre at $(0,0)$ then

$$\begin{array}{rcl} F_R(k,l) & \Rightarrow & \text{Symmetric} \\ F_I(k,l) & \Rightarrow & \text{Anti-symmetric} \end{array}$$

which can be written out explicitly as

$$\begin{array}{rcl} F_R(k,l) & = & F_R(-k,-l) \\ F_R(-k,l) & = & F_R(k,-l) \\ F_I(k,l) & = & -F_I(-k,-l) \\ F_I(-k,l) & = & -F_I(k,-l) \end{array}$$

as is shown in figure 8. The modulus square, or power spectrum, of the Fourier transform is

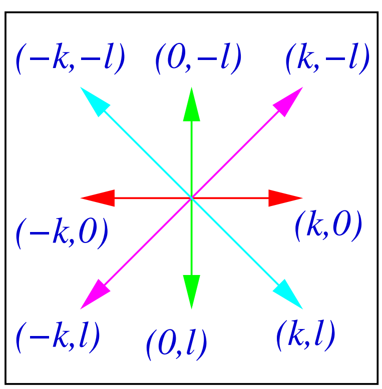


Figure 8: Symmetry of the two-dimensional Discrete Fourier transform.

$$|F(k,l)|^2 = |F_R(k,l)|^2 + |F_I(k,l)|^2$$

which must also be symmetric about the centre, which is clearly seen in figure 2 (b), being the Fourier transform of the real image in figure 2 (a).

This symmetry initially looks simple, but has to be considered carefully when N is even, to see this consider a simple case when $N = 4$, then in Fourier, after shifting the $(0,0)$ to the centre we have

$$F_R(k,l) \quad \text{and} \quad F_I(k,l) \quad \text{for } k \;\&\; l = -2,\ldots 1$$

as shown in figure 9.

In the *Real Part* there are:

- Four elements with no pair, these being $(0,0)$, $(-2,0)$, $(0,-2)$ and $(-2,-2)$.
- Four elements with symmetric pairs being $(-1,-1) \Leftrightarrow (1,1)$ $(-1,1) \Leftrightarrow (1,-1)$, $(0,-1) \Leftrightarrow (0,1)$ and $(-1,0) \Leftrightarrow (1,0)$.
- Two elements where their symmetric pair has been cyclically wrapped round by period 4, there being $(-2,-1) \Leftrightarrow (-2,1)$ and $(-1,-2) \Leftrightarrow (1,-2)$.

which gives a total of 10 elements that depend on the input data with the other 6 given by the symmetry properties.

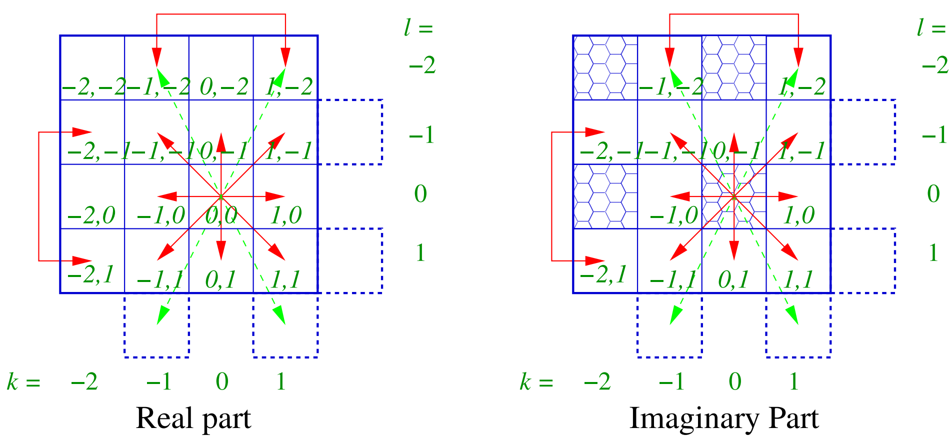


Figure 9: Symmetry of the Real and Imaginary parts of 4×4 Fourier Transform.

In the *Imaginary Part* there are:

- Four zero elements , these being $(0,0)$, $(-2,0)$, $(0,-2)$ and $(-2,-2)$, since $\sin(0) = \sin(\pi) = 0$.
- Four elements with anti-symmetric pairs being $(-1,-1) \Leftrightarrow (1,1)$ $(-1,1) \Leftrightarrow (1,-1)$, $(0,-1) \Leftrightarrow (0,1)$ and $(-1,0) \Leftrightarrow (1,0)$.
- Two elements where their anti-symmetric pair has being cyclically wrapped round by period 4, there being $(-2,-1) \Leftrightarrow (-2,1)$ and $(-1,-2) \Leftrightarrow (1,-2)$.

which gives a total of 6 elements that depend on the data, 6 more being given by the anti-symmetry and 4 always zero. Thus combining the real and imaginary results there are 16 elements in Fourier space that depend on the input data, so again there are the same number of elements in both spaces.

In the general case for the Discrete Fourier transform of an $N \times N$ pixel real image then in Fourier space we have

$$\begin{aligned} \frac{N^2}{2} + 2 &\rightarrow \text{Real Values} \\ \frac{N^2}{2} - 2 &\rightarrow \text{Imaginary Values} \end{aligned}$$

giving a *total* of N^2 values in the complex Fourier plane that depend on the input data, the others being given by the symmetry conditions. This allows allows the Fourier transform to be stored in the same amount of space as the original image which is of significant importance when N is large.

3.4 Calculation of the two-dimensional Discrete Fourier Transform

The expression for the two-dimensional Fourier transforms given in equation 2 and its inverse in equation 3, appear to be four-dimensional summations where it is required to sum over all $N \times N$ pixels of the image for each point in Fourier space. However noting that the exponential terms is separable, the the two-dimensional Fourier transform can be implemented in two passes giving to give

$$F(k,l) = \sum_{j=0}^{N-1} P(k,j) \exp\left(-\imath 2\pi \frac{lj}{N}\right)$$

where

$$P(k,j) = \sum_{i=0}^{N-1} f(i,j) \exp\left(-\imath 2\pi \frac{ki}{N}\right)$$

which can be considered as

1. N one-dimensional DFTs *along-the-rows*
2. N one-dimensional DFTs *down-the-columns*

as shown in figure 10. So we can implement the *two-dimensional* DFT, by a series of $2N$ *one-dimensional* DFTs.

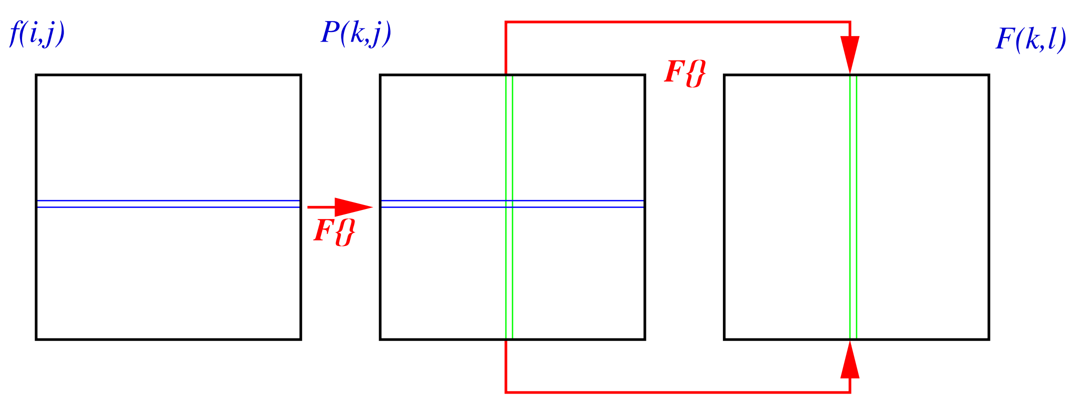


Figure 10: The separability of the two-dimensional Fourier transform into one-dimensional Fourier transforms.

3.5 Calculation of the one-dimensional Discrete Fourier Transform

The one-dimensional Discrete Fourier Transform of sampled signal $f(i)$ is given by

$$F(k) = \sum_{i=0}^{N-1} f(i) \exp\left(-\imath 2\pi \frac{ki}{N}\right)$$

which initially looks like a computational problems that scales at N^2, since for each of the N values of k there is a summation over the N samples of $f(i)$. This makes the direct calculation with a pair of *nested* `for` loops very computationally expensive and, for large N computationally impractical.

This calculation is typically performed by the `Fast Fourier Transform` algorithm that implements that above formula, with certain restrictions, and a computational cost that scales at $N\log_2(N)$, which is a very substantial saving for large N.The restrictions are:

- *Cooley & Tukey* original algorithm from the mid-1960's only works for

  $$N = 2^n$$

  known as the *Radix-2* transform. This algorithm is very widely available in libraries or packages. It can also be coded in a few dozen lines of code.

- *Singelton* developed a *Mixed-Radix* FFT that works with $N\log_2(N)$ cost for highly factorisable numbers, so typically

  $$N = 2^{n+1}3^m5^p$$

  This is much more complex scheme involving several hundred lines of code and is available in many computer languages. *Singeltons* code will actually take a DFT for any value N but unless N is highly factorisable, it will use the slow DFT algorithm that scales at N^2.

- *Numerical Package* many numerical and data analysis packages such as MATLAB, LABVIEW, IDL, R-PROJECT etc., all have internal FFT and the restrictions on N depend on the internal algorithm used, for example R-PROJECT uses *Singeltons* algorithm.

- FFTW[9] by Frigo & Johnson at MIT have developed a C-library that gives $N\log_2(N)$ scaling performance for *any* N removing the usual factorisable restriction. This packages also gives highly optimsied schemes for two and three dimensional Fourier transforms and *real-only* transforms.

Of these the FFTW packages is most efficiency, and for such a complex set of algorithms, relatively easy to use from C/C++, or via the local `jfftw3`[10], interface methods, from JAVA.

3.6 Practical Considerations

Whether using one-dimensional DFTs for signal processing or, in our case, two-dimensional DFTs in image processing and analysis, there are a few practical considerations.

[9] see www.fftw.org
[10] see details of project work

Most common image are recorded at 8-bits per pixel, so each pixel is an integer in the range $0 \to 255$. This is usually limited by the quality of the recording sensor, typically a CCD array and is sufficient for all but the most demanding scientific applications, for example in astronomy at x-ray medical imaging, 12-bits giving range of $0 \to 4096$. Therefore most images can simply be displayed on a computer monitor with

$$0 = \text{Black} \quad \Rightarrow \quad 255 = \text{White}$$

we will consider this in more detail later, but for most images, there is no significant problem here. The DFT, on the other hand is *complex*, and must be calculated in as floating point numbers. This has the effect both increasing the computational cost, since floating point calculation take longer than integer, and making the DFT more difficult to display and visualise. The most obvious function to display is

$$|F(k,l)|^2 \quad \textit{power spectrum}$$

which gives the power in each spatial frequency, however, for most images this has a huge dynamic range, $0 \to 10^{12}$ being typical. To if the maximum, usually at $(0,0)$ is displayed as *white* on a computer screen, then all the other pixels tend to be *black*, so we are unable to see the details. The dynamic range can be reduced using any monotonic function, the most common being to display

$$p(k,l) = \log\left(|F(k,l)|^2 + 1\right)$$

where the extra 1 is used to prevent problems when $|F(k,l)|^2 \approx 0.0$. This is what is displayed in figure 2 (b), which is the two-dimensional Fourier transform of the TOUCAN.

Most of the numerical algorithms to calculate the two-dimensional FFT result in a real and imaginary two-dimensional floating point arrays with the location of $F(0,0)$ at the *top/left*, where as we have seen above, we can shift the $(0,0)$ to any location without affecting the information displayed. So we typically want to shift the $F(k,l)$ so that the $(0,0)$ point is located at the centre of the screen. To perform this shift consider *convolution* of $F(l,l)$ with

$$\delta\left(i - \frac{N}{2}, j - \frac{N}{2}\right)$$

then from the *convolution theorem*, it can be shown, see workshops question 3.3 that this is equivalent to a multiplication in real space with a *checker pattern* of

$$\begin{array}{ccccc} 1 & -1 & 1 & \dots & -1 \\ -1 & 1 & -1 & \dots & 1 \\ 1 & -1 & 1 & \dots & -1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -1 & 1 & -1 & \dots & 1 \end{array}$$

so if we pre-multiply $f(i,j)$ with this pattern then the resultant Fourier transform will be centred in the middle of the output arrays, as shown in figure 2 (b).

3.7 Sampling Theory

Before we can use the above theory we need as ask what, given an continous image $f(x,y)$, then what

$$\Delta x \quad \text{and} \quad \Delta y$$

should we use to retain the all useful information in $f(x,y)$ when it is sampled. We well find that the answer to this depends on the maximum spatial frequency in the image, which is turns depends on the imaging system used. This is the problem considered by *sampling theory*.

3.8 Sampling a one-dimensional function

If we have a continious function $f(x)$, and the *width* of its Fourier tranforms is a, so that,

$$F(u) = 0 \quad \text{for } |u| > a/2$$

then *Shannon's Sampling Theoem*[11], states that $f(x)$ is *completely specified* by taking samples as intervals

$$\Delta x = \frac{1}{a}$$

but does not say anything about how many samples must be taken.

Before looking more closely at this, we need a mathemtical model for *taking a sample*. We know from the *shifting property* of the δ-function, that for a continious function $f(x)$ then

$$\int_{-\infty}^{\infty} f(x)\,\delta(x-a)\,\mathrm{d}x = f(a)$$

which is shown in figure 11, which has the effect of measureing, or *sampling* $f(x)$ at the position $x = a$.

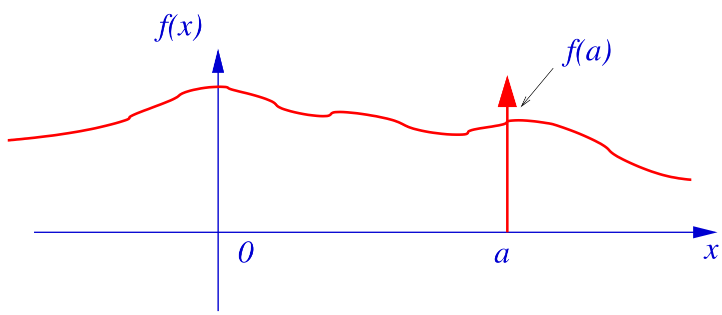


Figure 11: The shifting property of the δ-function.

If we want to *sample* or measure a function at regular intervals of Δx, then consider a *sampling function* consisting of a series of δ-functions separated by Δx, so being,

$$s(x) = \sum_{i=-\infty}^{\infty} \delta(x - i\Delta x)$$

which is the *Comb* function, shown in figure 12.

We have seen above that taking a single sample is equivalent to multiplication by a single δ-function, so taking a series of samples is equivalent to multiplication by the *sampling function*, $s(x)$. So if we sample $f(x)$ at interval Δx, then in real space we have

$$f(x)\,s(x)$$

[11] also known a Nyquist sampling frequency

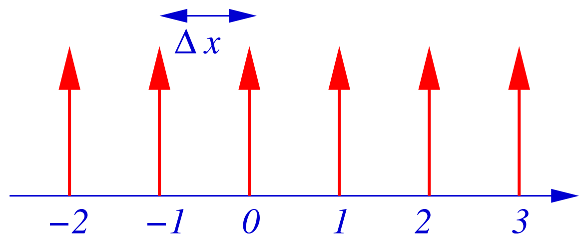


Figure 12: The Comb function, consisting of a series of δ-functions seperated by Δx.

Then from the *convolution theorem* we have that this is equivalent to a convolution in Fourier space of,

$$F(u) \odot S(u)$$

where $S(u)$ is the Fourier Transform of the $s(x)$ the Comb function. *It-can-be-shown* (see Tutorial 7 of Fourier Transform Booklet) that this is also a *comb* is reciprocal spacing, given by:

$$S(u) = \sum_{i=-\infty}^{\infty} \delta(u - \frac{i}{\Delta x})$$

So in real space we have the condition shown in figure 13 where the continious function $f(x)$ is samples at intervals of Δx to give our discrete function $f(i)$. In Fourier space we have the equivalent condition, as shown in figure 14 where the $F(u)$ is convolved with $S(u)$, which forms a series of replications of $F(u)$ at an interval of $1/\Delta x$.

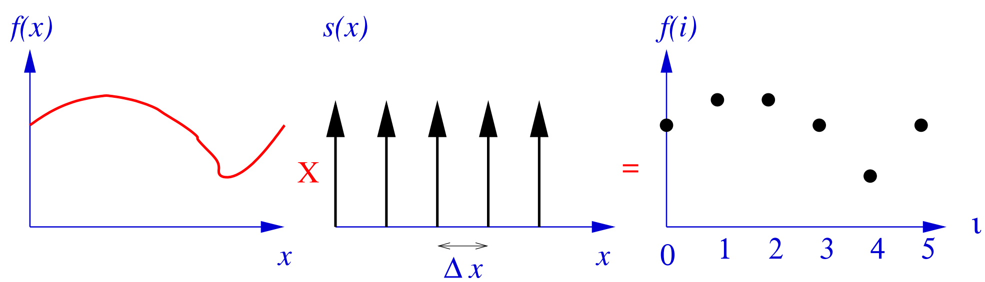


Figure 13: Sampling of a continious function in real space.

If the *width* of $F(u)$ is a, then provided that

$$a \leq \frac{1}{\Delta x}$$

then the replications in Fourier space will be separated with each replication being a *perfect* copy of $F(u)$. The sampling process that retain $F(u)$, and since the Fourier transform is a unitary transform then $f(x)$, the original function, can be completely recovered *any* of the replicated versions of $F(u)$, so the sampled version $f(i)$ thus retains *all* information about $f(x)$, which is exactly what Shannon's Theorem states.

The implications of the sampling conditions are that there are three possible conditions, these being:

Shannon Sampling: where the condition that

$$\Delta x = \frac{1}{a}$$

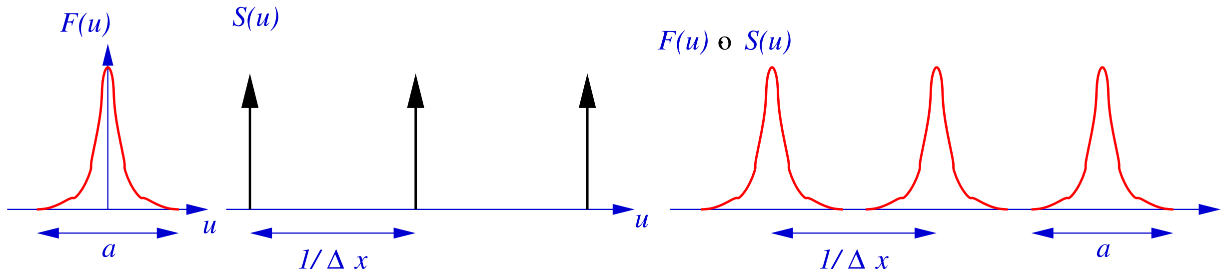


Figure 14: Effect of sampling continious function in Fourier space

is exactly obeyed. In Fourier space the replicated order are adjacent but non-overlapping as shown in figure 15. This is the condition for taking the *least* number of samples, and the maximun frequency recorded is given by

$$u_0 = \frac{1}{2\Delta x}$$

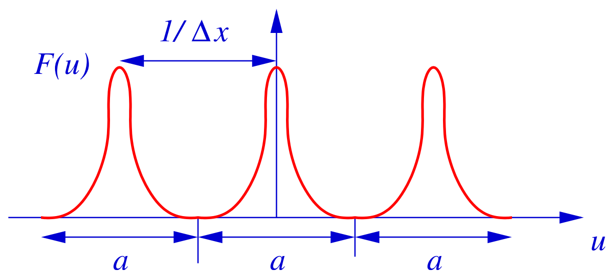


Figure 15: Effect of Shannon sampling in Fourier space

Undersampling: where the input function $f(x)$ is sampled less often than required to retain all information, so that

$$\Delta x > \frac{1}{a}$$

This results in overlap of the replicated orders in Fourier shape as shown in figure 16. This corrupts the Fourier transform in the area of overlap, which results in loss of information about $F(u)$ and hence loss of information about $f(x)$. This the original continious function can no longer be recovered from the sampled data $f(i)$. The effect of this depemds on the details of the Fourier transform $F(u)$, but typically results in spurious high frequency noise and *ringing* in any reconstrcution of $f(x)$, which is known as *ailising*.

Oversampling: where the input function $f(x)$ is sampled more often than required, so that

$$\Delta x < \frac{1}{a}$$

Here the replicated order are separated by blank regions in Fourier space as shown in figure 17. This adds no additional information over Shannon sampling, but you have more data so subsequent digital processing is slower.

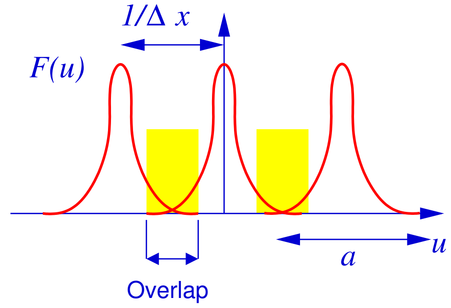


Figure 16: Effect of under sampling in Fourier space resulting in overlap of the replicated orders.

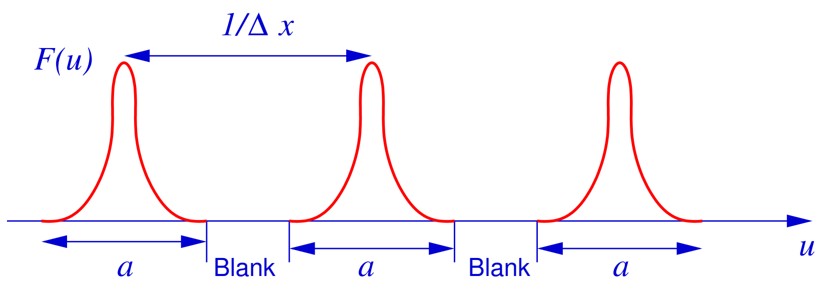


Figure 17: Effect of over sampling in Fourier space.

3.9 Sampling a two-dimensional function

When we sample a two-dimensional function $f(x,y)$, typically an image in our case, all of the above results carry through exactly as above. We now define a two-dimensional sampling function being

$$s(x,y) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} \delta(x - i\Delta x, y - j\Delta y)$$

which is a grid of δ-functions with separations Δx, Δy in the x and y directions. Again we can consider sampling as multiplication by the sampling function in real space, where we have

$$f(x,y)\,s(x,y)$$

as shown in figure 18.

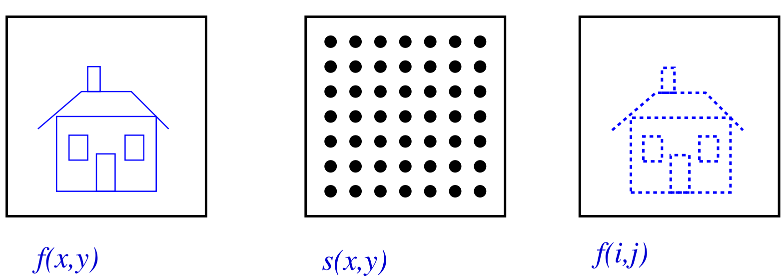


Figure 18: Two-dimensional sampling in real space.

Then in Fourier plane we get:

$$F(u,v) \odot S(u,v)$$

where we have that

$$S(u,v)=\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}\delta\left(u-\frac{i}{\Delta x},v-\frac{j}{\Delta y}\right)$$

being also a grid of δ-functions but with reciprocal spacing of $1/\Delta x$ and $1/\Delta y$ in the u and u directions. The result in Fourier space is a two-dimensional replication of $F(u,v)$ as shown in figure 19.

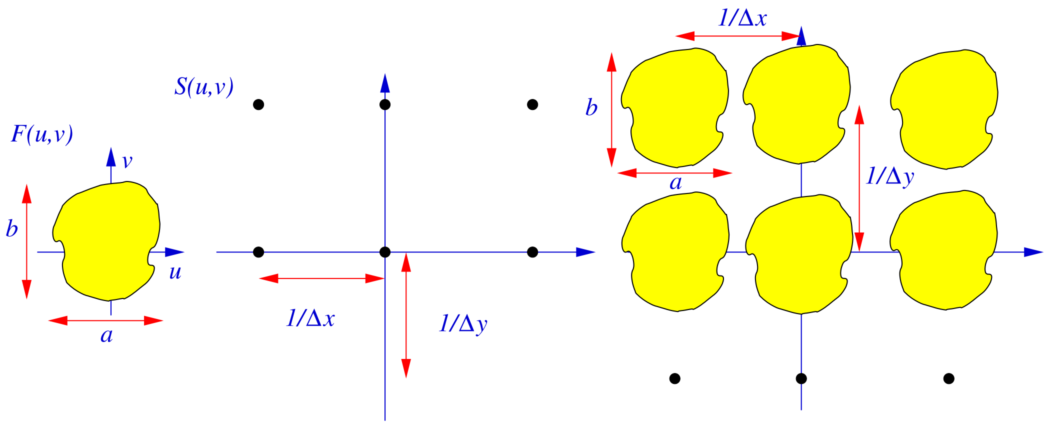


Figure 19: Two-dimensional sampling in Fourier space.

Now if $F(u,v)$ is contained with a *rectangle* of size $a \times b$, then the replications will be fully separated provided that that Δx and Δy are small enough, and in particular is

$$\Delta x < \frac{1}{a} \quad \Delta y < \frac{1}{b}$$

so setting the *Shannon Sampling Rate* in two-dimensions to be

$$\Delta x = \frac{1}{a} \quad \Delta y = \frac{1}{b}$$

In most practical cases whe will take $\Delta x = \Delta y$, simplifies the analysis considerably.

3.10 Functions of Finite Extent

Up to now we have considered either one or two-dimensional functions of infinte extend so, in principal, we have been taking a *infinite* number of samples, clearly not very practical. What we more typically really have is a function $\tilde{f}(x)$ being

$$\tilde{f}(x)=f(x)\,w(x)$$

where $w(x)$ is a *window* function given by,

$$w(x)=\Pi\left(\frac{x}{2d}\right)$$

so that $\tilde{f}(x)$ is of extend d so that $f(x)$ is only known in the range $x=-d/2\rightarrow d/2$, as shown in figure 20.

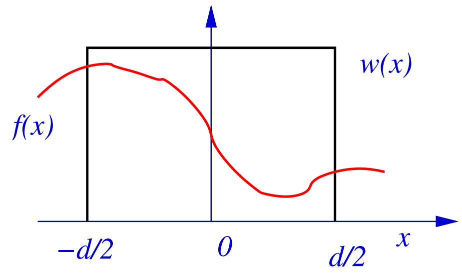


Figure 20: A windowed function of width d.

In real space we have a multiplication, from the *convolution theorm* in Fourier space we have that,

$$\tilde{F}(u) = F(u) \odot W(u)$$

where $W(u)$ is the Fourier tansform of a *tophat* so is given by,

$$W(u) = \mathrm{sinc}(\pi du)$$

as shown in figure 21, which for large du tends to zero, but is still of *infinite* extend.

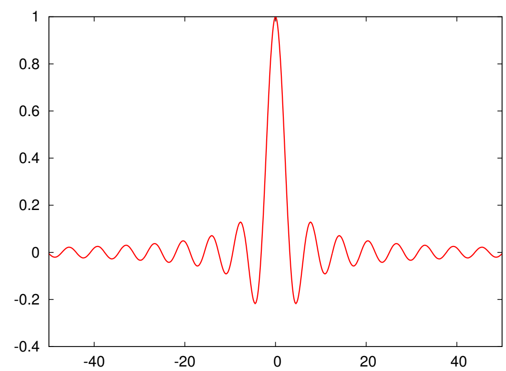


Figure 21: Plot of $\mathrm{sinc}(x)$, for large x.

This looks like a major problem for sampling theory, since although $F(u)$ mak be of finite extend, being of width a, as used above, but by limiting $f(x)$ is real space, we cane just convolved $F(u)$ with a function of *infinte* extend, so that,

$$W(u) \quad \text{infinite in extend} \quad \Longrightarrow \quad \tilde{F}(u) \quad \text{infinite in extend}$$

so *width* in Fourier space of $\tilde{F}(u)$, which is what we are really sampling, $a \to \infty$ so that

$$\Delta x = \frac{1}{a} \to 0$$

so sample point become *infititely* close together as shown in

However when d is large, so we have a large range of $f(x)$, then as shown in figure 22 then

$$W(u) = \mathrm{sinc}(\pi du)$$

becomes sharply peaked with the first zeros at $\pm 1/d$, with limit being,

$$d \rightarrow \infty \quad \text{then} \quad W(u) \rightarrow \delta(u)$$

so the *width* of $\tilde{F}(u)$ is the same as the *width* of $F(u)$, so we can apply *Shannons Sampling* with

$$\Delta x = \frac{1}{a}$$

where a is the width, or more technically, *bandwidth* of the $F(u)$, the Fourier transform of $f(x)$ the infinite extent function. Note that d is the length of the signal sampled, so if the sampling rate is Δx then

$$d = N \Delta x$$

so if N large, (take a lot of samples), then we can assume *Shannon Sampling*

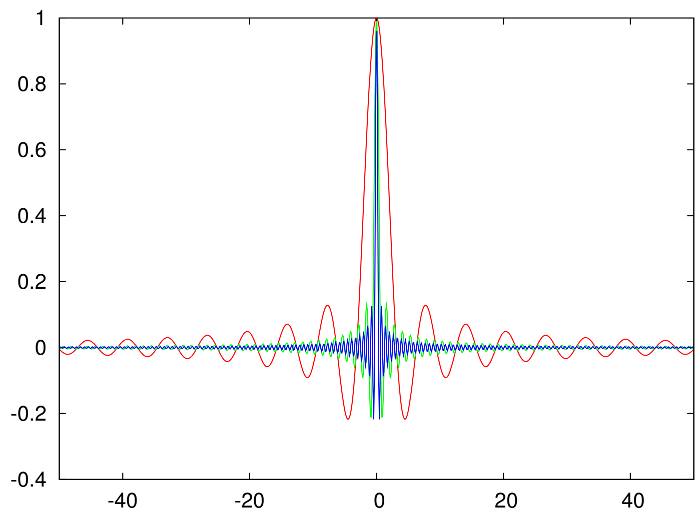


Figure 22: Plot of $\text{sinc}(du)$, as d become larger

Now if we consider a function $f(x)$ with bandwidth a, so that

$$F(u) = 0 \quad \text{for } |u| > a/2$$

then we can define the *Shannon Sampling* rate as

$$\Delta x = \frac{1}{a}$$

If in real space we take N samples, then the window length is $N\Delta x$, so that

$$d = N \Delta x$$

as we have just seen.

In Fourier space we cow consider sampling the funtion $\tilde{F}(u)$, where, as we have just seen that if N is large, then $\tilde{F}(u) \approx F(u)$. The *inverse* Fourier transform of $\tilde{F}(u)$ is then

$$\tilde{f}(x) = f(x)\, w(x) = 0 \quad \text{for } |x| > d/2$$

which is if *width* d, so we have the *Shannon Sampling* rate for $\tilde{F}(u)$ given by

$$\Delta u = \frac{1}{d} = \frac{1}{N\Delta x}$$

so for a function sampled at a rate Δx is real space, the equivalent sampling is Fourier space is $\Delta u = \frac{1}{N\Delta x}$.

In two-dimensions we have exactly the same analysis, so if we have a function $f(x,y)$ which is samples in real space at internals $(\Delta x, \Delta y)$, and we take $N \times N$ samples, then the equivalent sampling in Fourier space is,

$$\Delta u = \frac{1}{N\Delta x} \quad \& \quad \Delta v = \frac{1}{N\Delta y}$$

which is what we previously stated in equation 1.

3.11 Example Ideal Imaging System

Consider applying this theory to real imaging system, we have seen from the previous section that, provided that the system is *linear* and *space invariant* the imaging process can be characterised by the convolution of *object*, $o(x,y)$ and system *point spread function* $h(x,y)$, to give the continious image to be detected by the system of,

$$f(x,y) = o(x,y) \odot h(x,y)$$

so in Fourier space we have that

$$F(u,v) = O(u,v)H(u,v)$$

where $H(u,v)$ is the *Optical Transfer Function* being a fixed property on the imaging sytsem. We have also covered that for an *ideal*[12] imaging system, then $H(u,v)$ has an analytic solution, and in particular is given by

$$H(u,v) = \frac{2}{\pi}\left[\cos^{-1}\left(\frac{w}{v_0}\right) - \frac{w}{v_0}\left(1 - \left(\frac{w}{v_0}\right)^2\right)^{\frac{1}{2}}\right]$$

plotted in figure 23, where $w = \sqrt{u^2 + v^2}$.

For an optical system with focal length f and input aperture diameter d,

$$v_0 = \frac{d}{\lambda f} = \frac{1}{\lambda \mathrm{F_{No}}}$$

so that

$$H(u,v) = 0 \quad \text{for } w > v_0$$

therefore we have that

$$F(u,v) = 0 \quad \text{for } w > v_0$$

[12] The best possible, real system have aberration that make this worse.

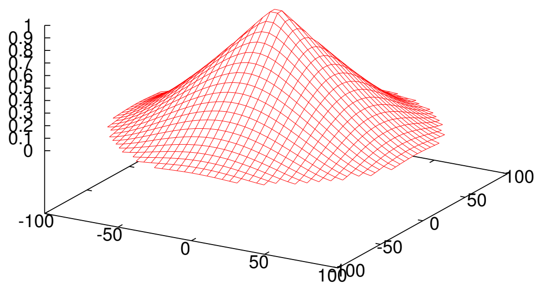


Figure 23: Shape of the OTF of an ideal imaging system.

so as shown in figure 19, $F(u,v)$ is contained withing a square of size $2v_0 \times 2v_0$, so that the image $f(x,y)$ is *bandlimited*, which is exactly what is required for *Shannon Sampling*. The *Shannon Sampling* rate for this is given by

$$\Delta x = \Delta y = \frac{\lambda \mathrm{F_{No}}}{2}$$

so if we consider the system of

- $\lambda = 0.5\mu$m (Green Light)
- $\mathrm{F_{No}} = 8$ (Medium aperture)
- Then $\Delta x = 2\mu$m.

so to fully *Shannon Sample* a 35 mm slide, of size 36×24 mm you would need to take $18,000 \times 12,000$ samples, so for full colour with 24-bits per pixels means a single image is 618 Mbytes!

In practice few systems are *ideal* and for film based systems, the limit is actually set by the size of the silver grains in the photographic film, for examples the SuperCOSMOS system at ROE digitises photographic plates at a step size of 10μm.

3.12 Reconstruction from Sampled Data

Once we have sampled the data into discrete samples we have to consider the inverse problem, of how do we reform the original data from the samples. In particular we have a function $f(x)$ which we have samples at interval Δx, to obtain $f(i)$, then if we have taken N samples we have the value if $f(x)$ at points,

$$x = x_0, x_0 + \Delta x, , x_0 + 2\Delta 2, \ldots x_0 + (N-2)\Delta x, x_0 + (N-1)\Delta x$$

but to fully reconstruct[13], we need to find $f(x)$ when x is *not* a sample point. We have from above that

$$f(i) = f(x)\, s(x) = \mathcal{F}^{-1}\left\{F(u) \odot S(u)\right\}$$

[13]If we have obeyed Shannon Sampling, we should have retained *all* information about $f(x)$.

which is shown in figure 14, so in Fourier space we have a series of replications of $F(u)$ separated by $1/\Delta x$. We can now isolate a single period by a *top-hat* filter of length $1/\Delta x$, being

$$H(u) = \Pi\left(\frac{u}{\Delta x}\right)$$

as shown in figure 24, so that, provided that the replications are *sufficiently* far apart, the

$$(F(u) \odot S(u))\, H(u) = F(u)$$

which in real space we have that the original function,

$$f(x) = h(x) \odot (f(x)\, s(x)) = h(x) \odot f(i)$$

where $h(x)$ is the inverse Fourier transform of $H(u)$, which is just,

$$h(x) = \frac{1}{\Delta x}\,\mathrm{sinc}\left(\frac{\pi x}{\Delta x}\right)$$

so to *reconstruct* $f(x)$ from the sampled data $f(i)$ we have to convolve with $h(x)$ which is know as the *interpolation* function.

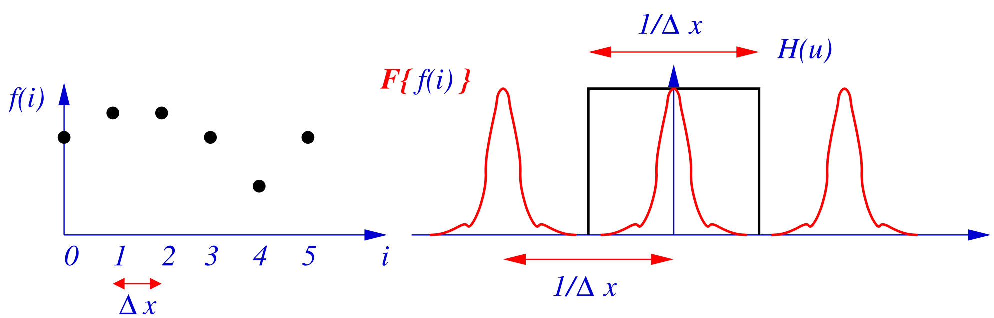


Figure 24: Fourier filter $H(u)$ used to isolate a single order in Fourier space.

.

Typically we normalise to get,

$$h(x) = \mathrm{sinc}\left(\frac{\pi x}{\Delta x}\right)$$

so that $f(x) = f(i)$ at a sample point, known as *ideal* interpolation function. This looks like the fully solution since as shown in figure 25, a sin() is placed at each sample point, where an then

- *At sample point* then $(f) = f(i)$ since the sinc() contributions from the other point are all zero.
- *Not at sample point* then $f(x)$ is a sum of the sinc() weighted components.

The problem occurs when we are not at a sample point and N is large; the computational cost is impractical, and approximations have to be made.

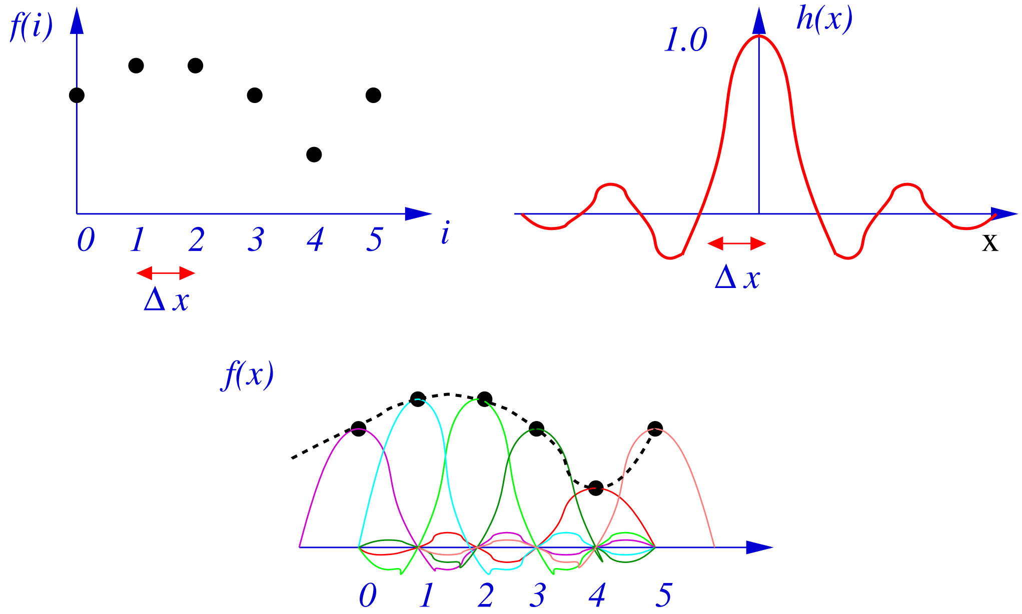


Figure 25: Ideal sinc interpolation in real space.

.

3.12.1 Zero Order Interpolation

The simplest scheme is *nearest neighbour rule*, also known as *zero-order* interpolation where we set,

$$f(x) = f(i) \quad \text{where } |x - i\Delta x| \text{ is minimised}$$

which is mathematically equivalent to taking the interpolation function $h(x)$ to be a *top hat*, so being

$$h_0(x) = \Pi\left(\frac{x}{\Delta x}\right)$$

This gives the characteristic *staircase* effect in the reconstruction shown in figure 26, which sharp discontinuities when the approximation swaps from one sample to the next.

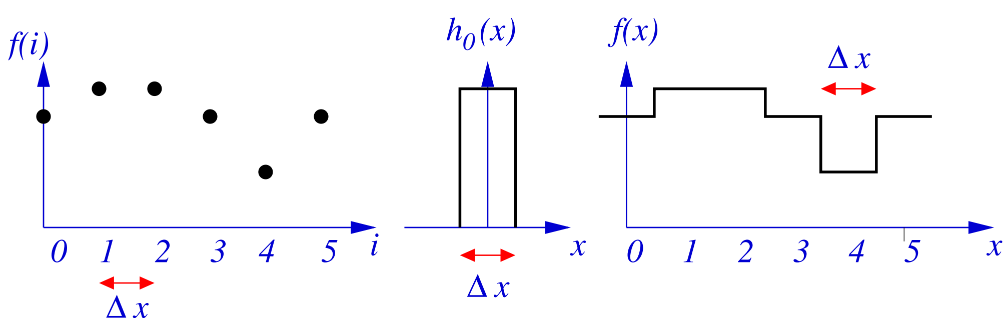


Figure 26: Zero order interpolation in real space.

The real problem of this scheme is evident in Fourier space. In real space we have convolved $f(i)$ with $h_0(x)$, so the Fourier space we have multiplied the periodic Fourier domain by it Fourier transform which is

$$H_0(u) = \mathrm{sinc}(\pi\Delta x u)$$

rather than the ideal window function, as shown in figure 27. This only partially separates the required single replication, and in particular

- *low pass* of the partially isolated $F(u)$ since $H_0(u)$ is not constant over the range $-1/2\Delta x \rightarrow 1/2\Delta x$.
- *aliasing* where information from the replicated orders are included in the reconstruction since $H_0(u) \neq 0$ for $|u| > 1/2\Delta x$.

Of there two issues, *aliasing* is by far the largest problem as it introduces spurious frequencies, and hence information, that were not in the original data. This is where the extra sharp transitions or edges come from in figure 26.

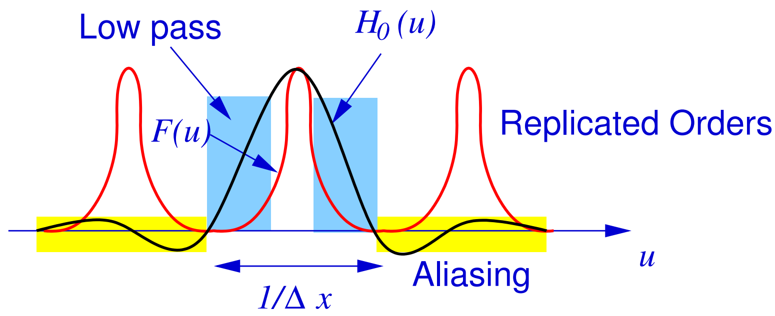


Figure 27: Zero order interpolation in Fourier space.

In two-dimensions where the sampled image is $f(i,j)$, we the *nearest neighbour* rule becomes

$$f(x,y) = f(i,j) \quad \text{for } |x - i\Delta x| \text{ and } |y - j\Delta y| \text{ minimised}$$

so the interpolation function become a two-dimensional *top-hat* being given by

$$h(x,y) = \Pi\left(\frac{x}{\Delta x}\right)\Pi\left(\frac{y}{\Delta y}\right)$$

The effect of using this to expand an image is shown in figure 28 where a 128×128 pixel original image is expanded to a 256×256 pixel image using zero-order interpolation. In the expanded image the *block pattern* is just visible, while in Fourier space the effect is rather more dramatic. The central region, outlined in red, is a *low-passed* version of the Fourier transform of the original image, but all the outer regions are spurious, aliased, information from the replicated Fourier orders. The example shown that the somewhat abstract interpolation theory does really apply in practice and simply doubling the size of the image by turning each pixel into a 2×2 block has had a huge effect on the Fourier transform of the image[14].

3.12.2 First Order Interpolation

The next order of interpolation is *linear* or *first order* interpolation where we take a linear combination of the two closest sampled elements, so for a for the value of $f(x)$ we take,

$$i = \text{int}\left(\frac{x}{\Delta x}\right) \quad \text{and} \quad \alpha = \frac{x - i\Delta x}{\Delta x}$$

[14]This is the first example of an apparently simple bit of processing having a very significant effect on the image data, we will see more later in the course.

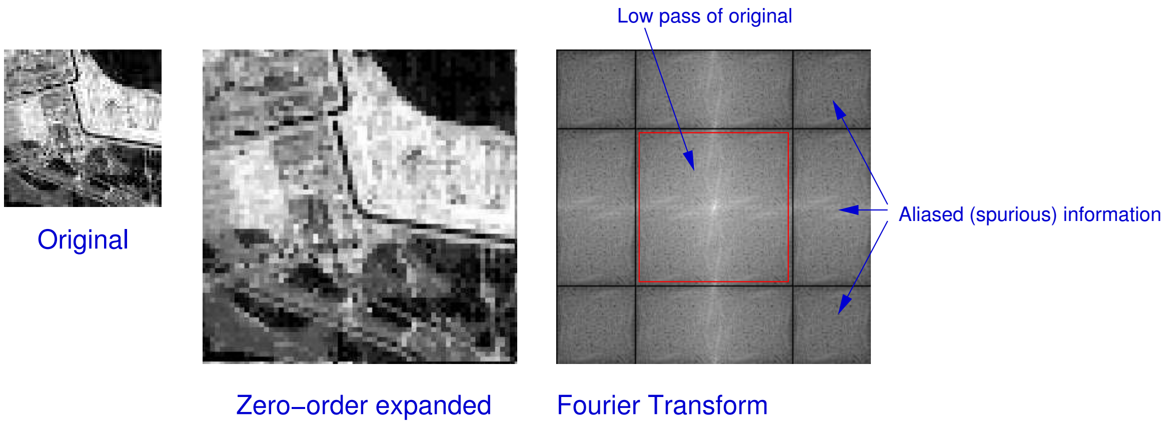


Figure 28: Effect of zero order interpolation in two-dimensions

so that i and $i+1$ are the two closest sample point, and then we have that

$$f(x) = (1-\alpha)f(i) + \alpha f(i+1)$$

which we can represent mathematically as convolution with a pyramid interpolation function $h_1(x)$ given by,

$$\begin{aligned} h_1(x) &= 1 - \tfrac{|x|}{\Delta x} && \text{for } |x| \leq \Delta x \\ &= 0 && \text{else} \end{aligned}$$

which is shown in figure 29, which gives a *smoother* reconstruction of $f(x)$ without the sharp steps.

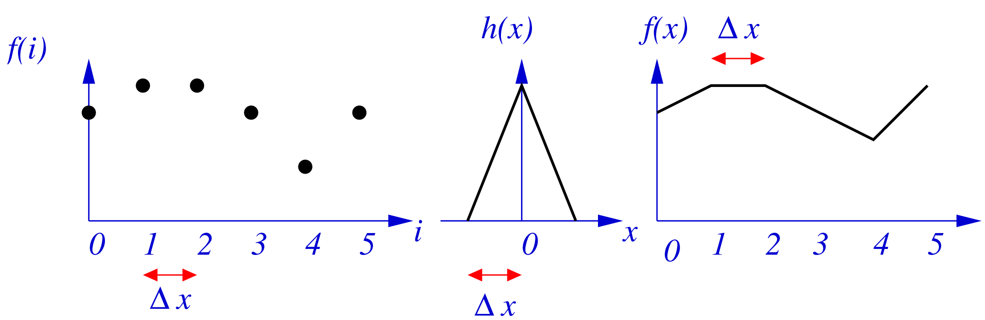


Figure 29: First order interpolation in real space.

In Fourier space we are applying the function $H_1(u)$ to the replicated Fourier domain, where *can be shown*[15], that

$$H_1(u) = \text{sinc}^2(\pi \Delta x u)$$

being just the square of the previous zero-order interpolation function $H_0(u)$. The plot of sinc() and sinc^2() shown the difference, with

- sinc$^($) having a fast fall of in the range $-1/2\Delta x \rightarrow 1/2\Delta x$, so has a greater *low pass* filtering effect on the reconstruction, so giving a smoother reconstruction,

[15] See *The Fourier Transform* (what you need to know)

- sinc$^{(}$) is *must smaller* in the region $|u| > 1/2\Delta x$, so introduces much less of the spurious aliased information.

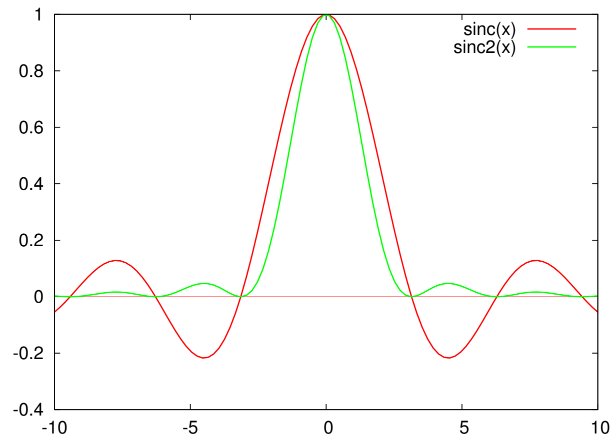


Figure 30: Plot of sinc() and sinc^2() on the same scale.

In two-dimensions when we have a sampled image $f(i,j)$, the interpolation function in real space becomes square pyramid given mathematically by

$$h_1(x,y) = \left(1 - \frac{|x|}{\Delta x}\right)\left(1 - \frac{|y|}{\Delta y}\right)$$

which can be implemented in real space as a weighted average of **four** adjacent points as shown in figure 31. We define

$$i = \mathrm{int}\left(\frac{x}{\Delta x}\right) \quad \& \quad j = \mathrm{int}\left(\frac{y}{\Delta y}\right)$$

being the closes sample point located in *top/left*, and then

$$\alpha = \frac{x - i\Delta x}{\Delta x} \quad \& \quad \beta = \frac{y - j\Delta y}{\Delta y}$$

as being the distance to the required point (x,y). The weighted average then become,

$$\begin{aligned} f(x,y) &= (1-\alpha)(1-\beta)f(i,j) + \alpha(1-\beta)f(i+1,j) + \\ &\quad (1-\alpha)\beta f(i,j+1) + \alpha\beta f(i+1,j+1) \end{aligned}$$

so we have to access **four** points for each x,y value.

The effect of this method of enlarging the 128×128 image to 256×256 is shown in figure 32. Compared to *zero-order* interpolation in figure 28, in real space the *block pattern* is almost totally gone due to the large reduction in the aliasing, but reconstruction looks rather *blurred* due to the low pass effect reducing the sharpness of the edges. This will be discussed in detail in the filtering section of this course. The same effect is visible in Fourier space with the central region, outlined in red, being more heavily *low passed* filtered and the amplitude of the aliased sections being much reduced. We will consider the implications of interpolation again later in the course.

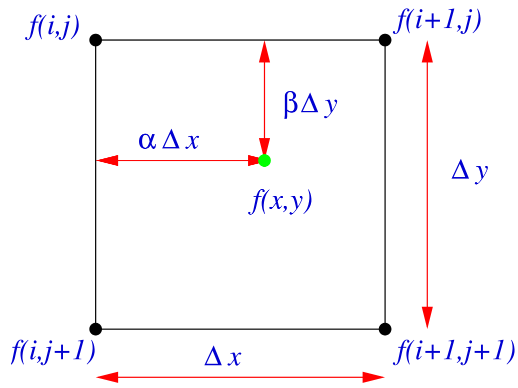


Figure 31: Two-dimensional first order interpolation in real space.

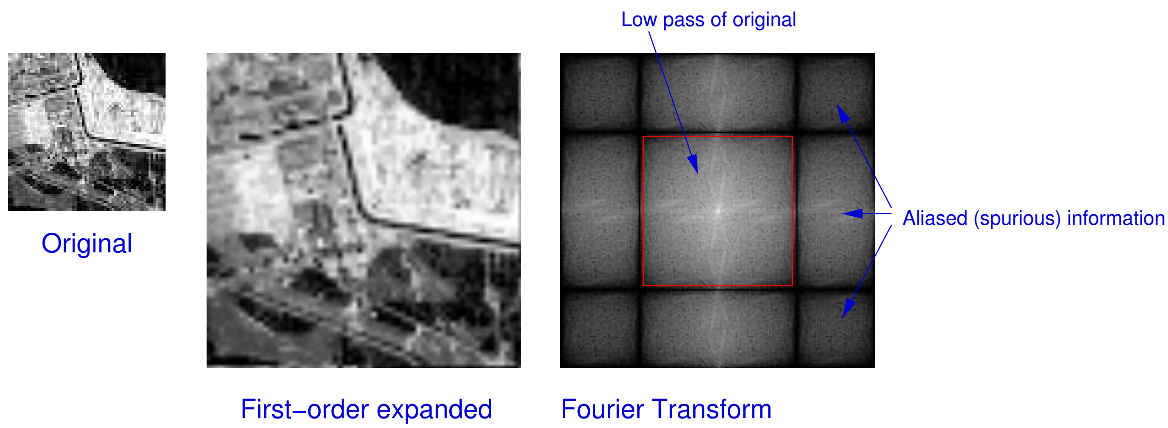


Figure 32: Effect of first order interpolation in two-dimensions

3.13 Higher Order Interpolation Schemes

There are range of higher order schemes to define $h(x)$ based on polynomial, splines, Gaussians and limited range $\cos()$s, etc, In general the larger the window over which the interpolation is formed be better the reconstruction. There schemes have been mainly developed for one-dimensional signal processing, where due to the relatively small amount of data a range of complex schemes have been developed, especially in digital music playback. In digital imaging the amount of data normally precludes use of these complex schemes, and interpolation is almost always limited to zero, first order as discussed above, and occasionally bicubic where the reconstructed value is formed from a weighted average over a 4×4 neighbouring samples. This is much more computationally expensive, but does result in improved reconstruction, and in particular a reconstruction that is continuous and has continuous partial derivatives. This scheme is especially favoured in digital photography where the smoothest and most *natural* result is required.

3.14 Summary

In this long theory section we have considered

1. Digital representation of images in real and Fourier space.

2. The discrete Fourier transform and its properties in one and two dimensions.
3. Calculation of the discrete Fourier transform by the FFT, and practical consideration in its calculation.
4. Sampling theory in one and two dimensions from a Fourier viewpoint.
5. Limitations of the sampling theorem and its practical application to an ideal optical imaging system.
6. Reconstruction from sampled and the ideal interpolation function.
7. Zero and First order interpolation and their effects in real and Fourier space.
8. Outline of higher order interpolation schemes.

Workshop Questions

3.1 Two-dimensional Symmetry

Show that the DFT of a two-dimensional real function has the symmetry properties of

$$\begin{aligned} F_R(k,l) &= F_R(-k,-l) \\ F_R(-k,l) &= F_R(k,-l) \\ F_I(k,l) &= -F_I(-k,-l) \\ F_I(-k,l) &= -F_I(k,-l) \end{aligned}$$

3.2 Symmetry Pairing

Verify for a 6×6 image that the DFT of a two-dimensional real function has:

$$\begin{array}{ll} \dfrac{N^2}{2}+2 & \text{Independent real values} \\ \dfrac{N^2}{2}-2 & \text{Independent imaginary values} \end{array}$$

Fourier filters involve multiplying the DFT by a filtering function $H(i,j)$. Many of these filters are real only. Suggest a scheme for packing the real and imaginary parts of a DFT into a square array that makes multiplication with such a filter simple.

3.3 Shifting The Centre

Show that if your two dimensional DFT code locates the $(0,0)$ term in the top/left of the array, then this can be shifted to the centre of the array by pre-multiplying the by a ± 1 checker-board.

3.4 Speed of the FFT

On a particular computer system the FFT of a 128×128 image takes 0.11 seconds, estimate how long this system would take to calculate the FFT of a 1024×1024 image.

3.5 CCD Sensors

A CCD sensor is a two-dimensional array of detectors that can be used to sample an image. A typical TV quality CCD camera will have 586×768 sensors on a 15 by 20 mm area with a $3:4$. Calculate the size of the sensors and the maximum spatial frequency in the detected image.

You wish to use this CCD camera to image pages of text for a character recognition system that is able to easily resolved 8pt (1pt is 1/72nd of an inch) letters. What magnification is required and how large a page of text can be images at once.

Hint: To easily resolve a letter you must be able to resolve line approximately 5 times closer together than the minimum separation of lines in the letter.