Collection: Module1 (08/30/11) - Overview

Print | Mark Read | Mark Unread | Tag | Go

Filter Author Show All Status Show All Read Status Show All Tags Show All GO

Select All Go **Sort by** Date **in** ↑ Ascending **Order** Go

Thread: The Fourier Transform
Post: The Fourier Transform
Author: Jered Wells

Posted Date: September 1, 2011 11:39 AM
Last Modified Date: September 1, 2011 2:08 PM
Status: Published

Q: FT theory requires a "band-limited" signal meaning that the signal must tend toward zero at infinity. How then do we take the FT of images?

A: Although an image may not have "zeros" at its border, we assume that all pixels which exist outside of the image bounds have zero-value. Try this in MATLAB:

>> Create an image with variable IM.
>> Use FFT2(IM) to take the 2D-DFT.
>> Display the FT using LOG10(FFTSHIFT(ABS(FFT2(IM)))) which is the log-transformed, zero-center Fourier amplitude.
>> Now zero-pad the original image using PADARRAY.
>> Take the FT of the new image and compare the Fourier amplitudes. Do you notice any differences?

The FT of the zero-padded image has caused the FT-matrix to grow in proportion to the amount of padding added. The zero-padding adds no new information, so you have effectively interpolated the original frequency space map to a larger pixel lattice.

You can also zero-pad in Fourier (frequency) space. Try this out. (Note that you will need to use REAL(IFFT2(FFT2(IM))) to apply the inverse transform. This forces the display of only the real parts of the inverse-transformed image.) What does zero-padding in Fourier space do for the image in the spatial domain? The answer... sinc interpolation!

There will be more information in class regarding the math of the DFT at a later date.

Tags: None Add Reply Quote Mark as Unread

Thread: Nyquist Sampling and aliasing
Post: Nyquist Sampling and aliasing
Author: Jered Wells
Attachment: Dobbins_on_MTF_NPS_NEQ.pdf (1.056 Mb)

Posted Date: September 1, 2011 11:40 AM
Last Modified Date: September 6, 2011 8:17 AM
Status: Published

Q: How exactly can we throw away something as high as 99% of the information in a signal and still get back the original image?

A: Sampling can be a tricky concept to understand. In order to help clarify concepts regarding sampling, the FT, etc., I will often resort to explanations involving a single, 1-D sinusoid (such as SIN(X)) for clarity. The expansion to 2-D is left to you, but the task is fairly trivial.

If I have a sinusoidal *function*, that is, a mathematical statement of the form:

$f(x) = \sin(x)$

this function is defined for all real numbers x which is infinite in extent. Therefore, if one were to sample $f(x)$ with a finite number of samples, you end up ignoring a lot of information. However, the Nyquist criterion says that we can sample this function at a frequency greater than twice the maximum frequency contained within the signal and still retain all of the information of the original signal. If you refer back to Dr. Dobbins' aliasing slide from the first lecture, you see that the high-frequency sinusoid was not sampled according to the Nyquist criterion in that fewer than $2*f_{max}$ samples were taken along the time course of the signal resulting in a string of samples which looks like a sinusoid of different frequency (i.e. it masquerades as or takes the alias of a different frequency). Try doing the following:

>> Draw a simple sinusoid which contains one full modulation every 5 cm on your paper.
>> Now draw vertical bars spaced as every 5 cm so that they intersect your sinusoid.

The intersections of the vertical bars a your signal are samples. If you observe only the samples, you should see that a line of zero-slope connects them all. Your sub-Nyquist sampling has resulted in the "detection" of a zero-frequency sinusoid. How unfortunate!

>> Using the same drawing from above, draw an additional sample in-between every pair of original samples so that the sampling frequency is now twice that of your original sampling frequency.
>> Using the lowest-frequency sinusoid possible, try to connect the dots/samples.

An infinite number of sinusoids can be used to fit these data, but all of those sinusoids have the same (minimum) frequency. If you were to again sample at *slightly* higher sampling rate (1 sample per 2.49999999 cm), you would instantly be able to reproduce the original sinusoid with exact frequency, phase and amplitude since you are sampling *above* the Nyquist sampling rate. (This is because the general form of a sinusoid is $f(x) = A \sin(ux + p)$).

In essence, as Dr. Dobbins said before, sampling of a band-limited signal limits the set of possible interpolation points between samples until we reach the Nyquist threshold which tells us that we know all of the values of the original function exactly (given that we know the maximum frequency contained in the original signal). This is achieved through sinc-interpolation which is discussed briefly in the other thread via an example.

Q: I didn't quite get why sampling creates replicas of F(u) in Fourier space.

A: The concept of aliasing is derived from the same reason that we see replicas of a sampled function in Fourier space. First, we know that the FT of a comb (sampling) function in the spatial domain is also a comb function in the frequency domain. Second, we all know that multiplication (sampling) in one domain is equivalent to convolution in the other domain. Therefore, the multiplication of a comb function with a spatial function (image) is the same as convolving the FT of the comb function with the FT of the image which result in the replicates observed in frequency space. With regard to aliasing, if the original samples are too far apart in the spatial domain, the replicates in the frequency domain will end up being too close together. Due to superposition, the high-frequency tails of the replicates overlap those of adjacent replicates resulting in the phenomenon known as aliasing. Because of the overlap, the signals sum and we are unable to retrospectively separate the aliased signals. There will be more on this topic in future lectures. For more information, see p. 176 of Dr. Dobbins theoretical paper on MTF, NPS and DQE from Medical Physics (attached).

Tags: None Add Reply Quote Mark as Unread

Thread: The Fourier Transform
Post: RE: The Fourier Transform
Author: Nooshin Kiarashi
Attachment: results.tif (476.436 Kb)

Posted Date: September 1, 2011 3:42 PM
Last Modified Date: September 1, 2011 3:46 PM
Status: Published

Thanks for the explanations. I tried the zeropadding in frequency domain and would like to share the outcome. Note the size of the final spectrum and image in the attached file. Below is the code used to generate it.

```
a = imread('cameraman.tif');
figure; subplot(2,2,1);
imagesc(a); colormap gray; title('original image')

A = fftshift(fft2(a));
subplot(2,2,2);
imagesc(log(abs(A))); colormap gray; title('original spectrum')
```

```
Ap = padarray(A,[100 100],'both');
subplot(2,2,3);
imagesc(log(abs(Ap))); colormap gray; title('zeropadded spectrum')

ap = ifft2(Ap);
subplot(2,2,4);
imagesc(abs(ap)); colormap gray; title('final image')

print -dtiff results
```

Tags: None Add Reply Quote Mark as Unread

Thread: The Fourier Transform
Post: RE: The Fourier Transform
Author: Jered Wells
Posted Date: September 2, 2011 12:12 AM
Last Modified Date: September 2, 2011 12:12 AM
Status: Published

Thank you very much for your contribution, Nooshin. It took me a while to figure out that zero-padding in MATLAB is an exact execution of sinc-interpolation. For the longest time I tried writing my own sinc-interpolation algorithm while the quick and easy solution was actually hiding right there under my nose! I'm glad this example could be of some help.

Tags: None Add Reply Quote Mark as Unread

Thread: The Fourier Transform
Post: RE: The Fourier Transform
Author: Lynda Ikejimba
Posted Date: September 7, 2011 10:47 AM
Status: Published

Very interesting! I hadn't noticed this effect previously. In MATLAB using padarray to make the matrix dimensions a power of 2 improves processing time, but based on this exercise it seems we are in a way modifying our FT with sinc interpolation. I'm guessing the scientific community is aware of this, but it is not clear to me why we are allowed to use this now interpolated data to make conclusions about whatever it is one is analyzing, as opposed to using the "true" FT.

Another question is: whenever we take the DFT of an image, aren't we effectively multiplying by a 2D rect function? Won't the FT of this image always be convolved with a sinc, regardless of additional padding or not?

Any ideas?

Tags: None Add Reply Quote Mark as Unread

Thread: The Fourier Transform
Post: RE: The Fourier Transform
Author: Jered Wells
Posted Date: September 8, 2011 10:25 AM
Status: Published

Lynda:

MATLAB does automatically pad your images up to the next power of 2 when you call FFT, but you will notice that the actual transformed image is still the same size as your original, meaning that you have not actually done any interpolation. Rather, MATLAB is simply using the next power of 2 in order to accelerate processing. However, you can change the size of the padded array by calling FFT2(X,MROWS,NCOLS) in order to force MATLAB to produce a result which has a size MROWSxMCOLS thus forcing sinc interpolation of the result. The same is true for the inverse transform.

With regard to your second question, by binning your image onto an MxN pixel lattice, you are in effect convolving by your sampling function at that step. The corruption of an analogue signal brought on by sampling is done at the discretization step, not by the DFT. The DFT understands only what it is given in terms of the amount of information contained in your image (image dimensions and bit-depth/data type). We use the DFT because we do not typically have analogue information available for input when dealing with computer systems. We can approach the analogue case by increasing the sampling rate (decreasing the pixel size) so that the execution of the DFT approaches that of the FT. This happens because, in the limit of small pixel size, your pixel basis function approaches a delta function whose FT is unity. If the FT of a function is unity, its MTF is also unity everywhere meaning that it does not corrupt the input signal at all. Returning to your original question, the MTF of a rectangular pixel basis function is the magnitude of a sinc function which indeed suppresses the visualization/replication of higher frequencies. Therefore, we conclude that insufficient sampling has corrupted our image, not the DFT.

Tags: None Add Reply Quote Mark as Unread

OK