Re: FFT and DFT

Source: http://sci.tech−archive.net/Archive/sci.physics/2006−03/msg01226.html

- *From*: Matthew Lybanon <lybanon@xxxxxxxxxxxx>
- *Date*: Tue, 14 Mar 2006 15:20:49 GMT

in article 4415D32C.3B47ECE5@xxxxxxxxxxx, Mike Fontenot at mlfasf@xxxxxxxxxxx wrote on 3/13/06 2:16 PM:

> laguiche2004@xxxxxxxx wrote:
>
> > I use the function cvDFT (OpenCV library) in Visual C++. But i don't find the same Matlab 's result with the function fft2. Why?
> >
> > for the vector [0 0 1], Matlab return [1, −0.5+0.866i, −0.5−0.866i], and with the function cvDFT in OpenCV i have [1, 1, −0.5−0.86i] for the same vector.
>
> I haven't used either package, but the fft requires that the number of samples be a power of 2, whereas the dft doesn't restrict the number of samples. Your input has three samples, so you have to use the dft, not the fft. It may be that the fft program accepts your input, and pads a zero on the end to get four samples...but the spectrum of the resulting signal is different than for just the original three samples (because the dft and fft basically consider the samples to be one period of a periodic stream).
>
> Mike Fontenot

The "Fast" Fourier transform is fast (faster than simply implementing the definition of the discrete FT) when the number of samples can be factored (i.e., the number is not a prime). The details of the FFT algorithm exploit the factored form of the number, and the greatest speed gain occurs when the number is a power of 2. More or less, a power of two can be factored "the most." Some implementations of the FFT may be limited to powers of 2, but complete implementations of the algorithm do not.

As another poster points out, some implementations may cheat by "padding" the end of the sample vector with enough 0s to make the augmented number of samples a power of 2.

.