Discrete Fourier Transform

$$F(\xi) = \int_{-\infty}^{\infty} f(x) exp(-i2\pi\xi x) dx$$

$$F(\xi) \approx F(\xi_m) = \sum_{n} f(x_n)\, exp(-i2\pi\xi_m x_n)\Delta x$$

Fast Fourier Transforms (FFTs) are a family of algorithms that optimize the DFT for computational speed.

DFT Examples

Editor - C:\Users\jschw\Dropbox\Class\OPTI 51

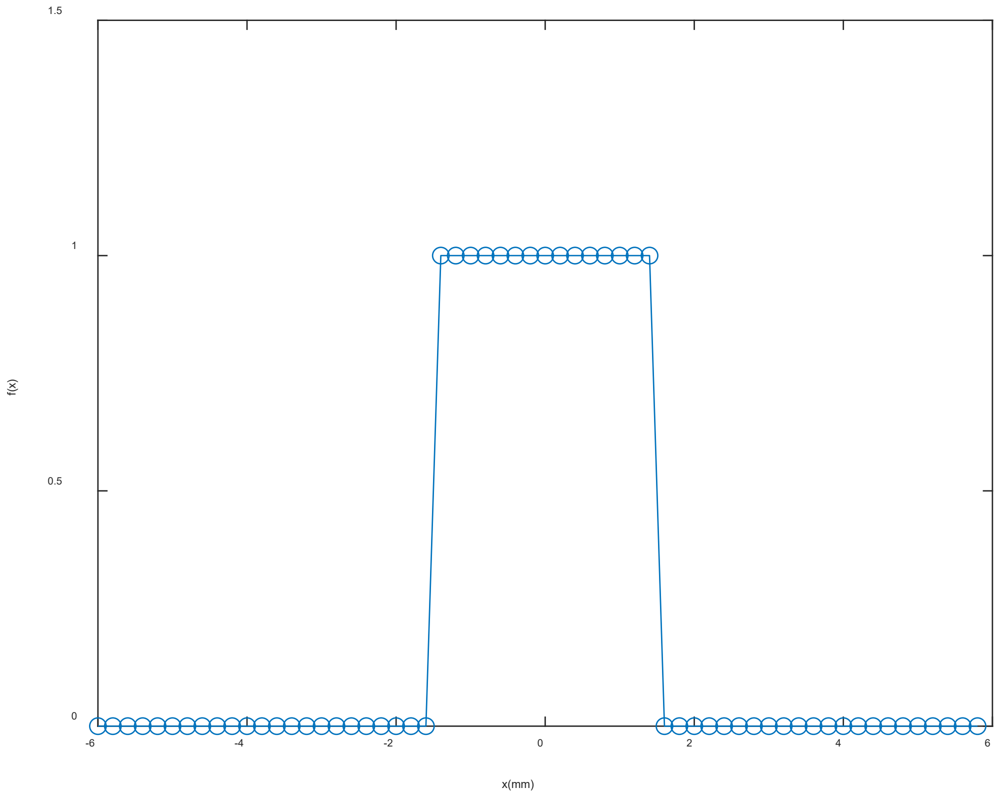
DFTDemos.m | rect.m

```
function[out]=rect(x)
% rectangle function
out=abs(x)<=1/2;
end
```

Editor - C:\Users\jschw\Dropbox\Class\OPTI 512 Linear Systems, Fourier Transforms

DFTDemos.m | rect.m

```
b=3.0; %rectangle width (mm)
L=12.0; %vector side length (mm)
N=60; %number of samples
Xs=L/N; %sample interval (m)

x=-L/2:Xs:L/2-Xs; %coordinate vector
f=rect(x/b); %rect function at values x
xi=-1/(2*Xs):1/(N*Xs):1/(2*Xs)-1/(N*Xs);
F=fft(f);

figure(1)
plot(x,f,'-o'); %plot f(x) = rect(x/b)
axis([-6 6 0 1.5]);
xlabel('x(mm)');
ylabel('f(x)');
```



Continuous Functions

$$f(x) = rect\left(\frac{x}{3}\right)$$

$$F(\xi) = 3sinc(3\xi)$$

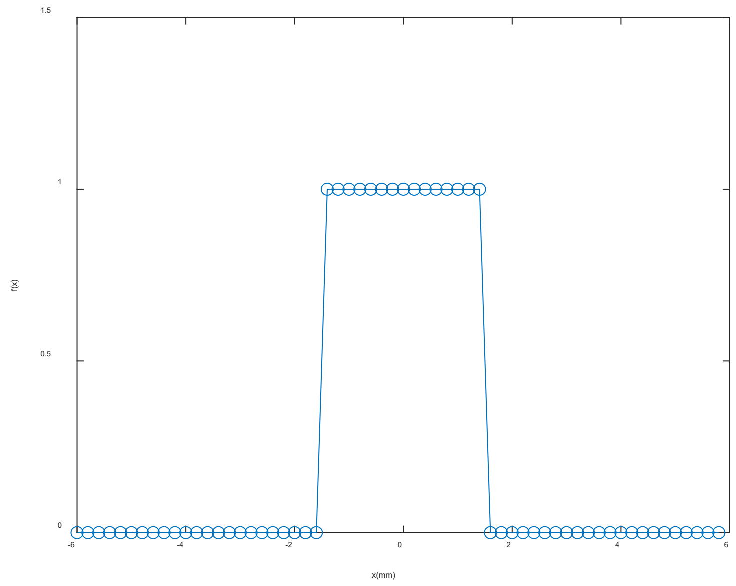
DFT Examples

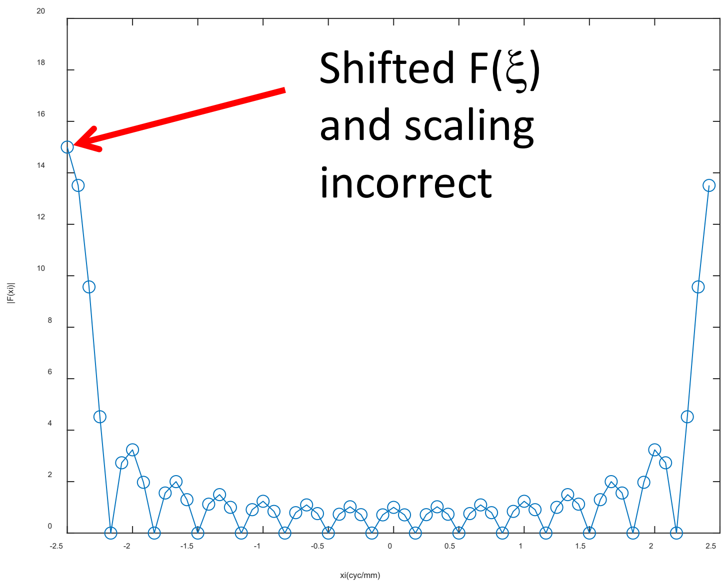
Editor - C:\Users\jschw\Dropbox\Class\OPTI 512 Linear Systems, Fourier Transforn

DFTDemos.m* | rect.m | +

```
b=3.0; %rectangle width (mm)
L=12.0; %vector side length (mm)
N=60; %number of samples
Xs=L/N; %sample interval (m)

x=-L/2:Xs:L/2-Xs; %coordinate vector
f=rect(x/b); %rect function at values x
xi=-1/(2*Xs):1/(N*Xs):1/(2*Xs)-1/(N*Xs);
F=fft(f);

figure(1)
plot(x,f,'-o'); %plot f(x) = rect(x/b)
axis([-6 6 0 1.5]);
xlabel('x(mm)');
ylabel('f(x)');

figure(2)
plot(xi,abs(F),'-o'); %plot F(xi) = b sinc(b*xi)
axis([-2.5 2.5 0 20]);
xlabel('xi(cyc/mm)');
ylabel('F(xi)');
```

Unshifted f(x)





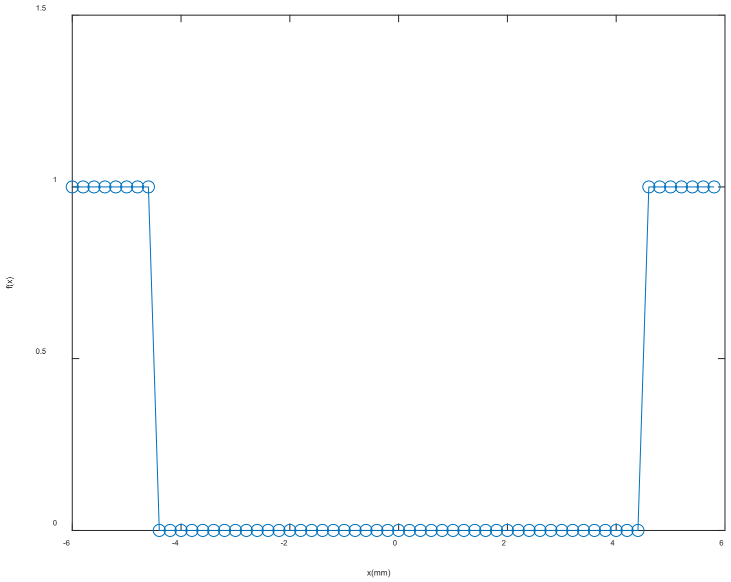
DFT Examples

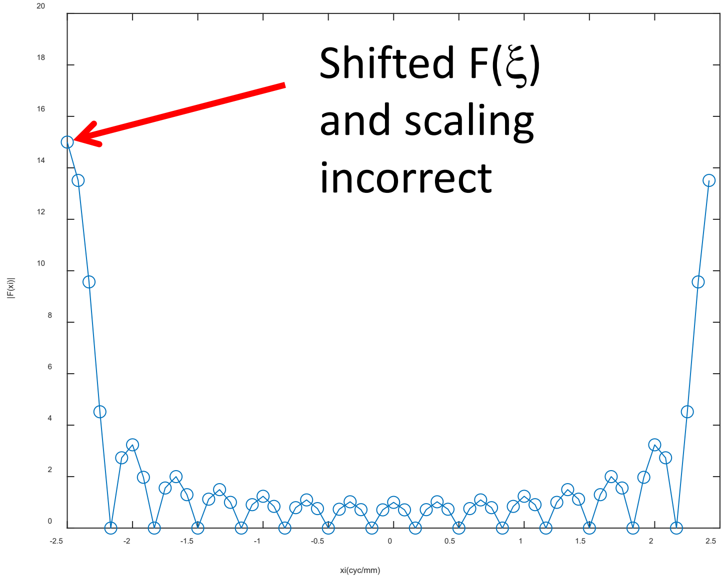
Editor - C:\Users\jschw\Dropbox\Class\OPTI 512 Linear Systems, Fourier Transforms\No

DFTDemos.m | rect.m

```
b=3.0; %rectangle width (mm)
L=12.0; %vector side length (mm)
N=60; %number of samples
Xs=L/N; %sample interval (m)

x=-L/2:Xs:L/2-Xs; %coordinate vector
f=fftshift(rect(x/b)); %rect function at values x
xi=-1/(2*Xs):1/(N*Xs):1/(2*Xs)-1/(N*Xs);
F=fft(f);

figure(1)
plot(x,f,'-o'); %plot f(x) = rect(x/b)
axis([-6 6 0 1.5]);
xlabel('x(mm)');
ylabel('f(x)');

figure(2)
plot(xi,abs(F),'-o'); %plot F(xi) = b sinc(b*xi)
axis([-2.5 2.5 0 20]);
xlabel('xi(cyc/mm)');
ylabel('|F(xi)|');
```

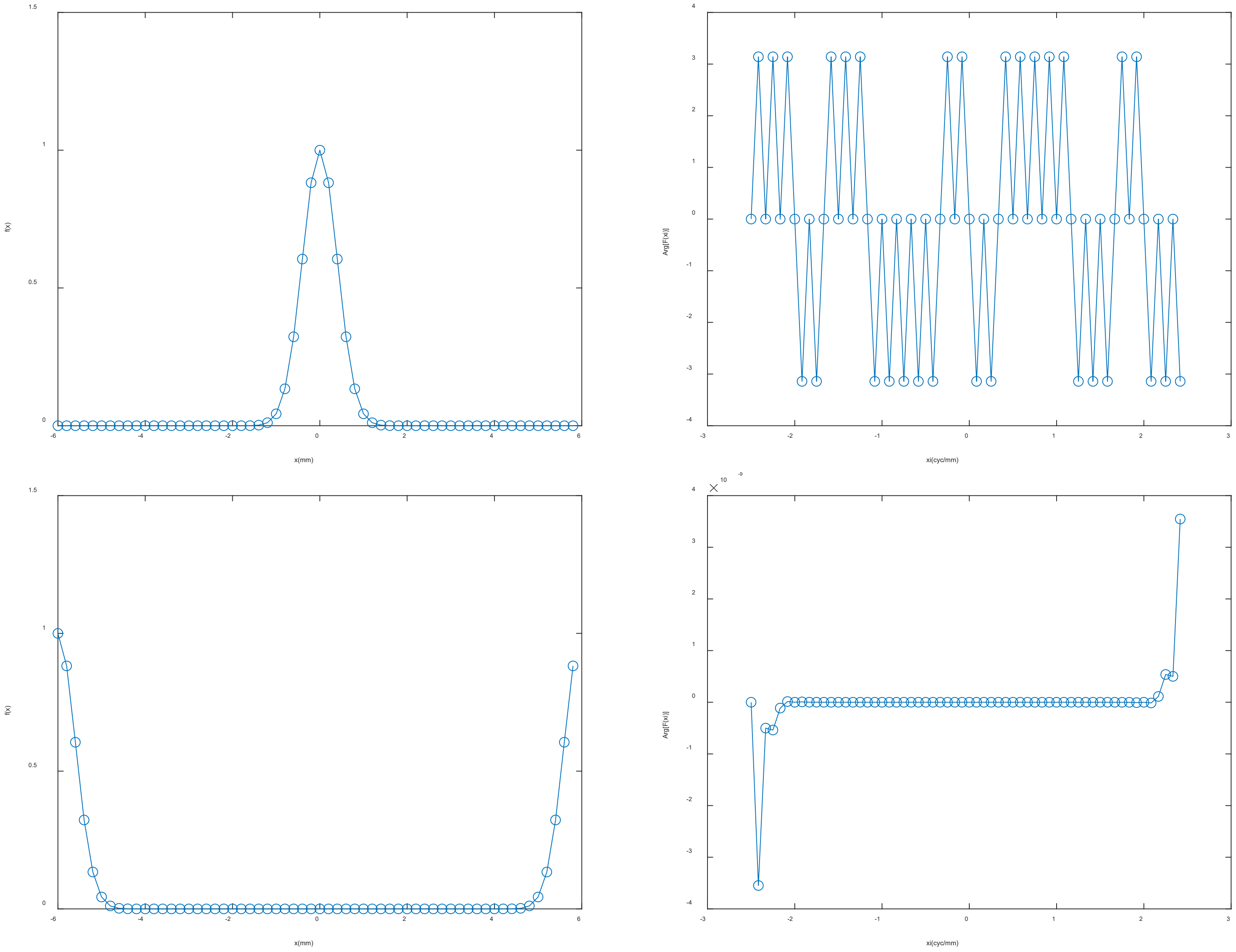
Shifted f(x)





Shifted F(ξ) and scaling incorrect

Shifted Object and Phase of Transform



At first glance, the effect of the shift on the object might not be obvious. Here, when the object is centered.

The modulus looks correct for both centered and decentered objects, but the phase only looks correct when the object is shifted as well.

DFT Examples

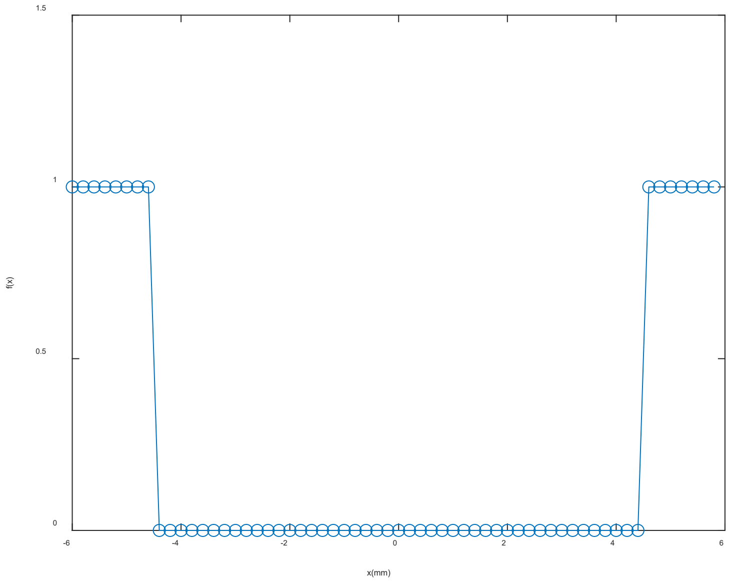
Editor - C:\Users\jschw\Dropbox\Class\OPTI 512 Linear Systems, Fourier Transforms\Notes

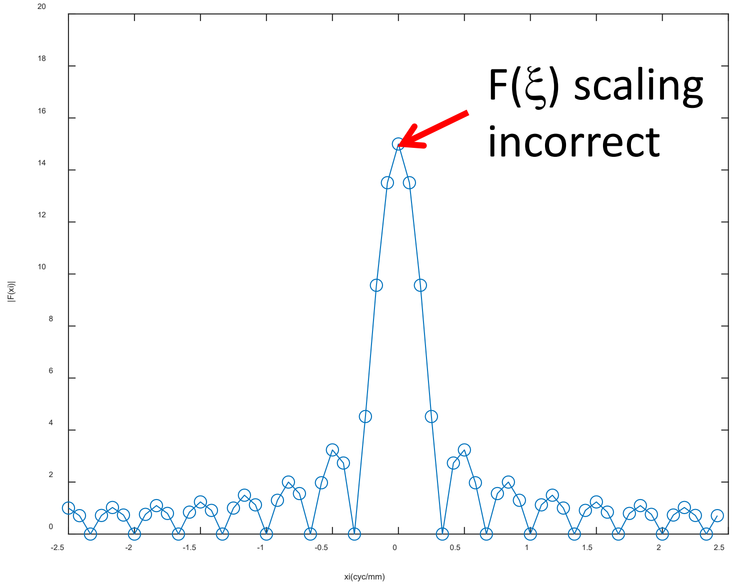
DFTDemos.m | rect.m

```
b=3.0; %rectangle width (mm)
L=12.0; %vector side length (mm)
N=60; %number of samples
Xs=L/N; %sample interval (m)

x=-L/2:Xs:L/2-Xs; %coordinate vector
f=fftshift(rect(x/b)); %rect function at values x
xi=-1/(2*Xs):1/(N*Xs):1/(2*Xs)-1/(N*Xs);
F=fftshift(fft(f));

figure(1)
plot(x,f,'-o'); %plot f(x) = rect(x/b)
axis([-6 6 0 1.5]);
xlabel('x(mm)');
ylabel('f(x)');

figure(2)
plot(xi,abs(F),'-o'); %plot F(xi) = b sinc(b*xi)
axis([-2.5 2.5 0 20]);
xlabel('xi(cyc/mm)');
ylabel('|F(xi)|');
```

Shifted f(x)

Shifted F(ξ)





DFT Examples

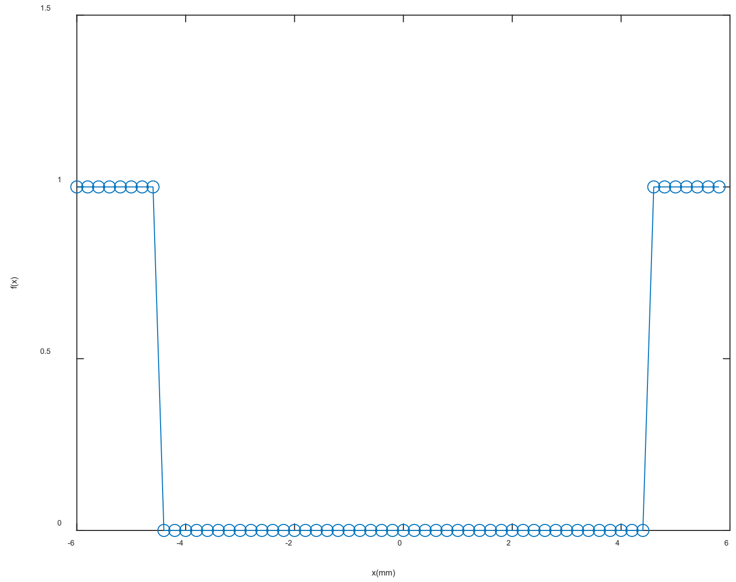
Editor - C:\Users\jschw\Dropbox\Class\OPTI 512 Linear Systems, Fourier Transforms\No

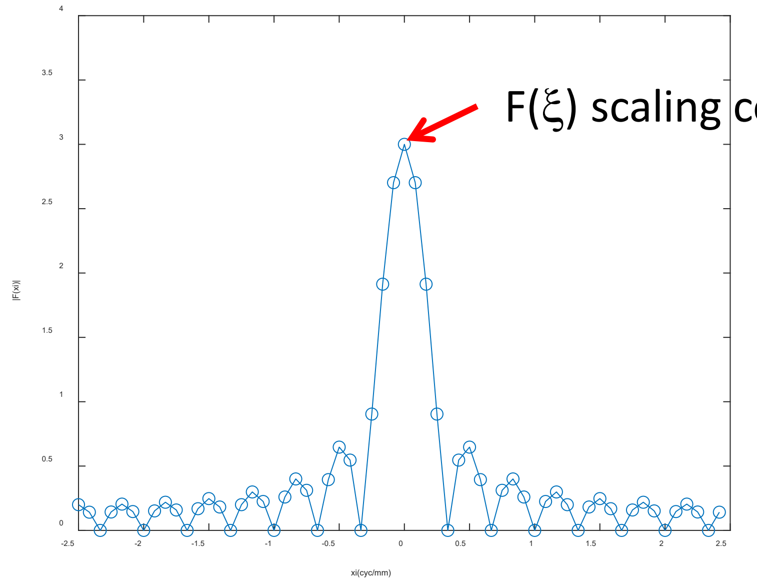
DFTDemos.m | rect.m

```
b=3.0; %rectangle width (mm)
L=12.0; %vector side length (mm)
N=60; %number of samples
Xs=L/N; %sample interval (m)

x=-L/2:Xs:L/2-Xs; %coordinate vector
f=fftshift(rect(x/b)); %rect function at values x
xi=-1/(2*Xs):1/(N*Xs):1/(2*Xs)-1/(N*Xs);
F=Xs*fftshift(fft(f));

figure(1)
plot(x,f,'-o'); %plot f(x) = rect(x/b)
axis([-6 6 0 1.5]);
xlabel('x(mm)');
ylabel('f(x)');

figure(2)
plot(xi,abs(F),'-o'); %plot F(xi) = b sinc(b*xi)
axis([-2.5 2.5 0 4]);
xlabel('xi(cyc/mm)');
ylabel('|F(xi)|');
```

Shifted f(x)

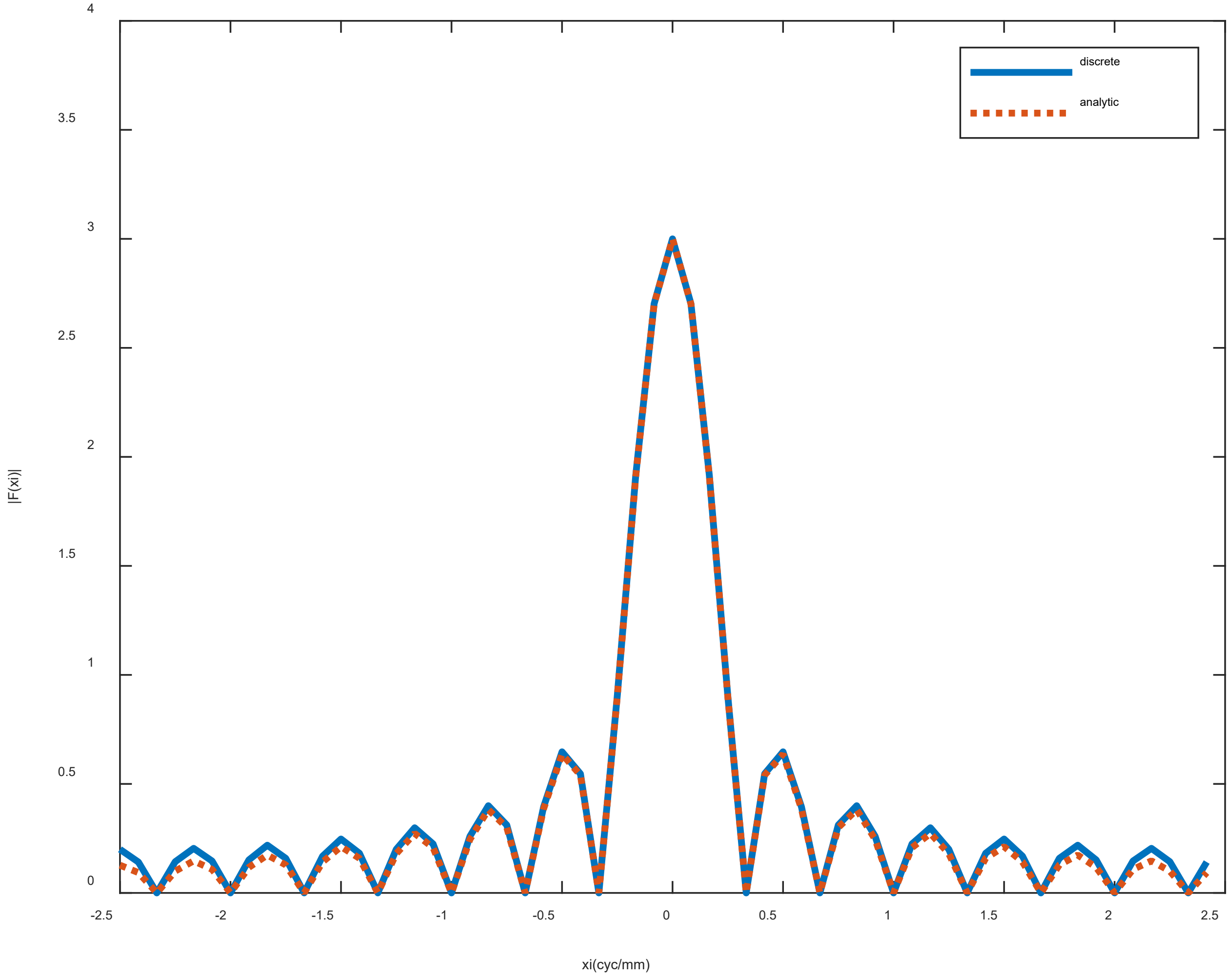
Scaled and Shifted F(ξ)





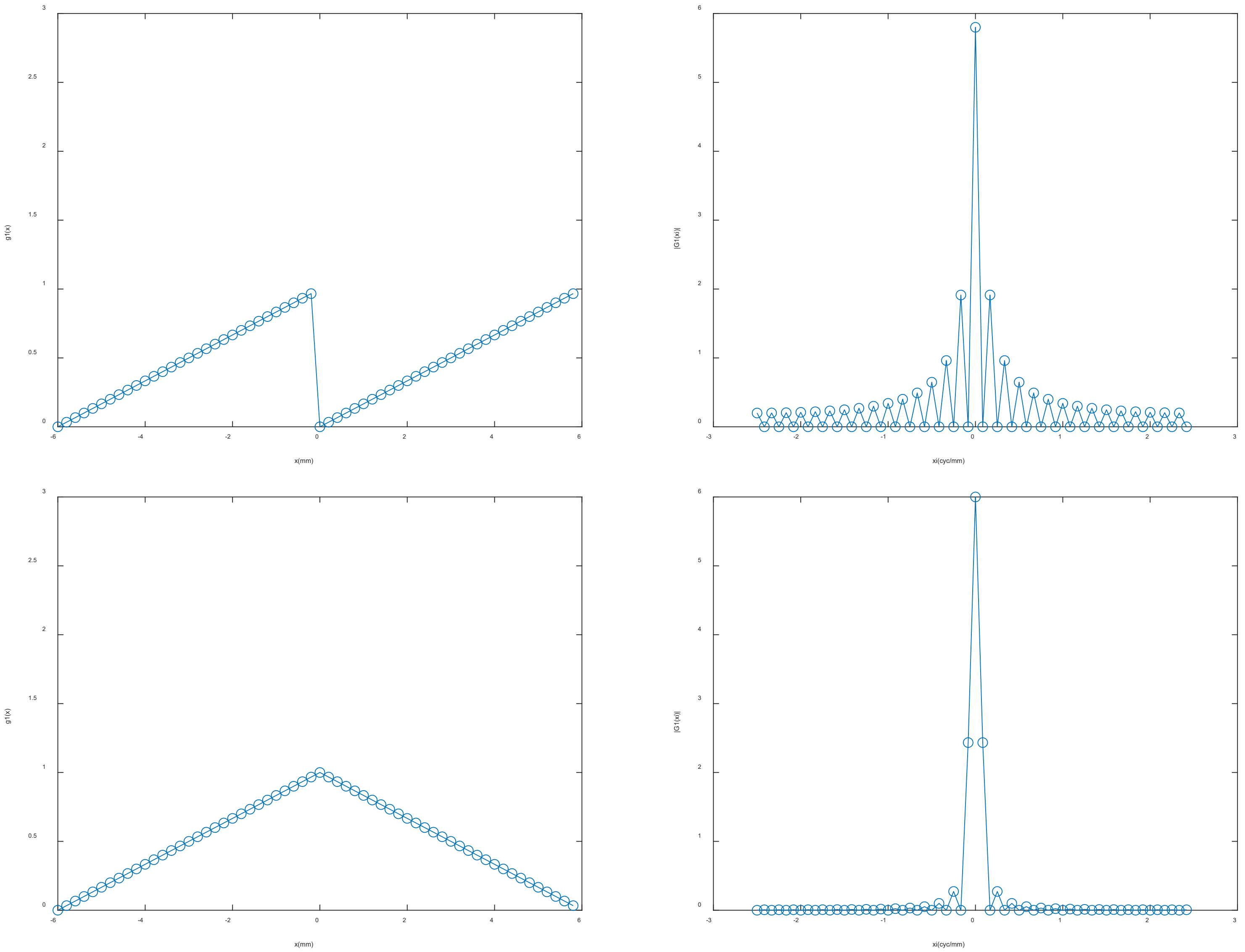
F(ξ) scaling correct

DFT Examples



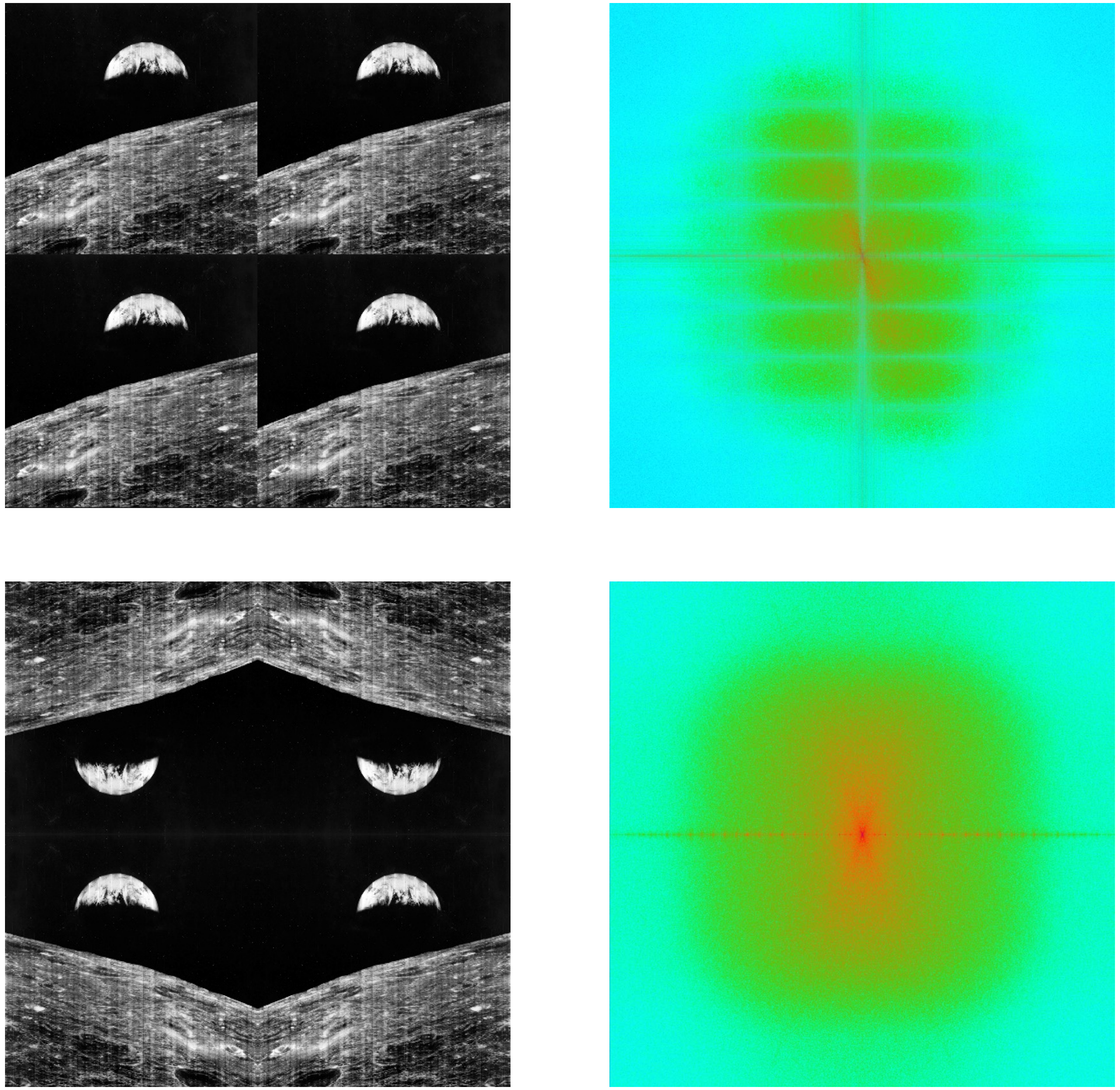
Comparing the discrete and continuous answers shows some small discrepancies near the edges.

Discontinuities at Edges



Small values for high frequencies

Discontinuities at Edges



Since the DFT inherently replicates the object, any discontinuities across the boundaries of this replication causes high frequency values in the transform which leads to aliasing.

This issue can be reduced by creating an object that is mirrored about its horizontal and vertical edges. This leads to a smoother transition at the boundaries, lower frequency values and less aliasing.

Convolution

Convolution tends to smooth out high frequency and lead to a resulting function that has a "width" equal to the sum of the widths of the two functions being convolved.

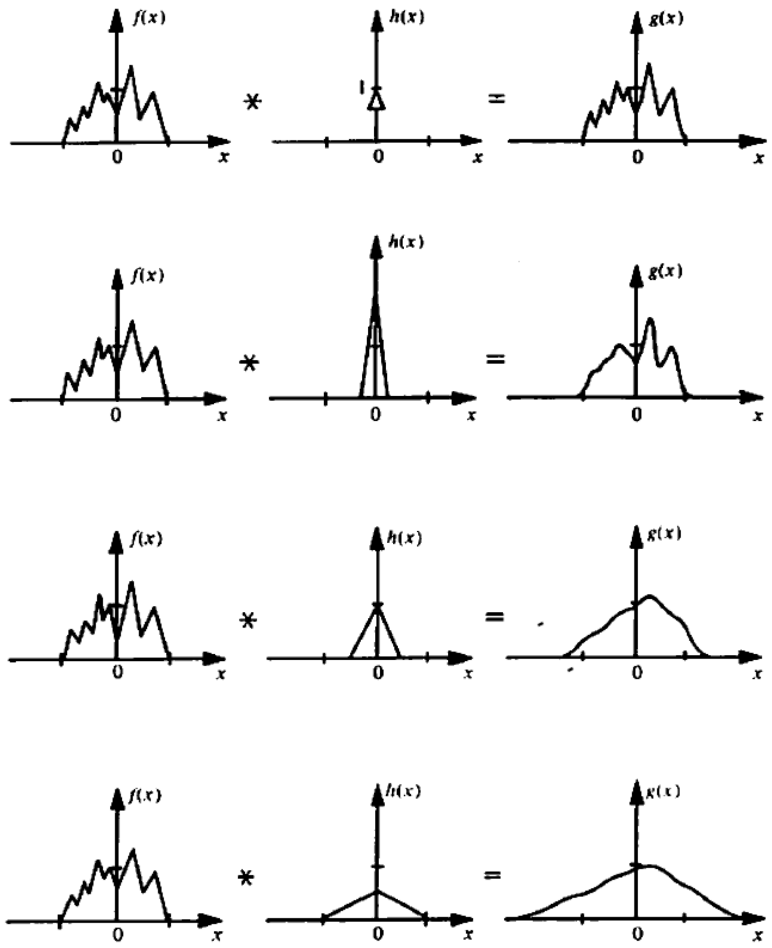


Figure 6-5 Smoothing effects of convolution.

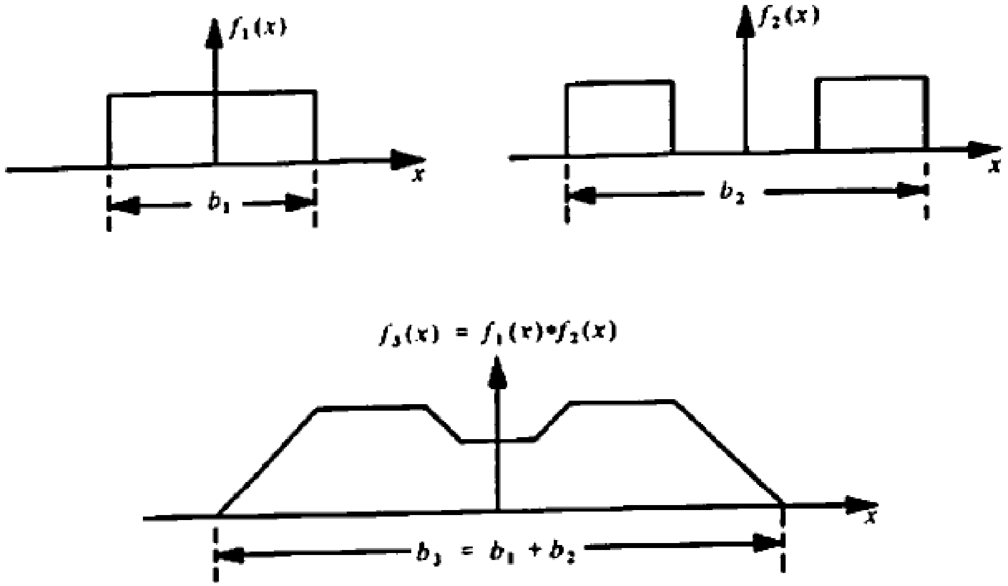
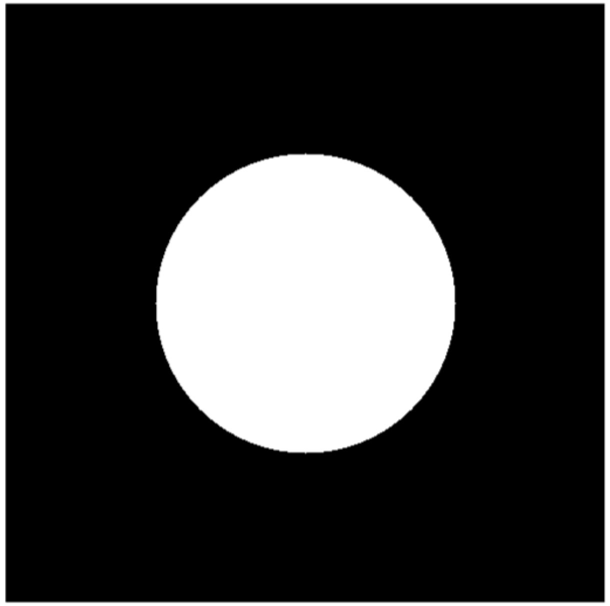
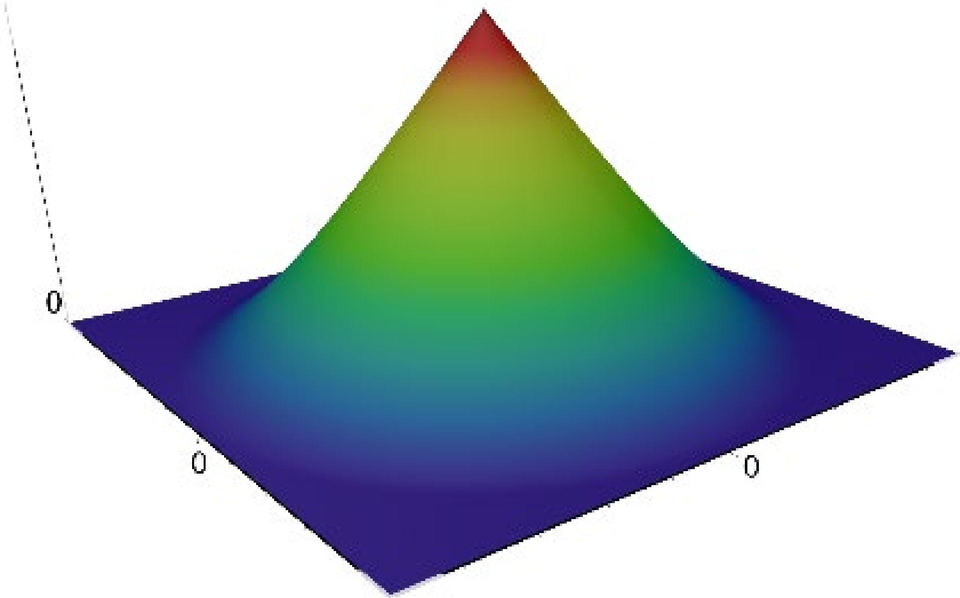


Figure 6-6 Convolution of two functions with compact support.

Convolution & Correlation





In general, when calculating convolution & correlation, the size of the array should be at least the size of the sum of the two functions being convolved (correlated). Here a cyl() function is shown along with its autocorrelation. The array needs to be twice the diameter of the cylinder to avoid aliasing. This is sometimes referred to as "padding" the array.