Math Highlights

Convolution: $(f * g)(\tau) = \sum_{i=0}^{N} f(t_i) g(t_i - \tau)$

Correlation and convolution: $R_{fg}(\tau) = (f * g)(\tau),\ f(t), g(t) \in \mathbb{R}$

Fourier transform: $\hat{f}(\omega) = \int_0^T e^{i\omega t} f(t) dt$

Convolution theorem: $(f * g)(\tau) = \mathcal{F}^{-1}\left[\hat{f}(\omega)\hat{g}(\omega)\right],$

Weiner-Khinchin theorem: $P_{ff}(\omega) = \mathcal{F}\left[R_{ff}(\tau)\right] = \frac{|\mathcal{F}[f(t)]|^2}{T} = \frac{\left|\hat{f}(\omega)\right|^2}{T}$

Fourier Amplitude: $\left|\hat{f}(\omega)\right| = \sqrt{Re\left[\hat{f}(\omega)\right]^2 + Im\left[\hat{f}(\omega)\right]^2}$

Fourier Phase: $\angle\hat{f}(\omega) = tan^{-1}\left(Im\left[\hat{f}(\omega)\right] / Re\left[\hat{f}(\omega)\right]\right)$

Convolution

```
a = 1;
mu = 0;
sigma = 1;
u = a*exp(-((x-mu).^2)/2*sigma^2);

b = 1;
k = 1;
v = b*exp(-k*x);

c_uv = conv(u,v);
plot(x,v)

c_uv = conv(u,v,'shape');
'shape' = full, same, valid
full = full convolution
same = central region that's the same length as u and v
valid = returns only the region that can be computed without zero padding
```

Correlation

```
Most often will need to correlate deviation from the mean:
delta_u = u - mean(u);
delta_v = v - mean(v);

xcorr(u,v,'option',maxlag);  % cross-correlation
```

```
xcorr(u,'option',maxlag);    % autocorrelation
```

set maxlag to the maximum number of elements that corresponds to the maximum desired lag time.

xcorr will compute the two-sided correlation with positive and negative lags. This will be symmetrical for real-valued functions, so typically you just see the one-sided correlation with the positive lags. Means that, for a maximum lag time (converted to an index) of N, xcorr returns 2N-1 elements. To remove the negative lags as well as the zero lag, simply take the elements (N+1):end.

'option' refers to the type of normalization. 'biased' normalizes the entire sequence by 1/N. 'unbiased' normalizes by 1./(N-lag). 'coeff' normalizes the zero lag to 1. 'none' is the default.

fft, sampling frequency, Nyquist frequency

```
Ns = 512;  % Number of samples in frequency domain
fs = 100;  % sampling frequency = 1/(time resolution)
```

Nyquist frequency is half the sampling frequency. This means that the fastest resolvable process is 2 samples, i.e., 2 time bins.

fast Fourier transform: fft(x,Ns);

Inverse fft: ifft(x,Ns);

Power and Phase Spectra

pretty straightforward:

```
x_hat = fft(x,Ns);
psd = real(x_hat).^2 + imag(x_hat).^2
phase = atan(imag(x_hat)./real(x_hat));
```

fft for convolutions

We must zero-pad our vectors in using fft and ifft to calculate convolutions, otherwise we wind up with artifacts from the fft.

```
upad = [u zeros(1,numel(u)-1)];
vpad = [v zeros(1,numel(v)-1)];
c_uv = ifft(fft(upad).*fft(vpad));
c_uv = c_uv(1:numel(u));
plot(x,c_uv);
```