Lecture 13: Frequency/Wavenumber Spectra

Recap

We've looked at multiple strategies for computing spectra in the frequency domain or by extension in the wavenumber domain. We've considered uncertainties, resolution and multiple methods. Now, we'll consider what happens when we want to consider sinusoidal patterns of variability in both time and space.

Frequency-Wavenumber examples.

First consider some example frequency-wavenumber spectra. What frequencies and wavenumbers are resolved? What is plotted? What is the Nyquist frequency and Nyquist wavenumber? Is the full frequency-wavenumber space represented?

As a reminder, frequency represents cycles per unit time, and wavenumber represents cycles per unit distance.

We use frequency-wavenumber spectra as a means to track propagating sinusoidal patterns. If it has a characteristic wavelength and frequency, we might suppress that variability if we didn't think about the full structure of the propagating wave. Westward propagating Rossby waves and eastward propagating Kelvin waves have characteristic frequencies and characteristic wavenumbers.

Basics.

Consider a data set $y(x,t)$ where $-x_f < x < x_f$ and $-T < t < T$ (where here we're using $\pm x_f$ for the end points in space, and following our previous examples $\pm T$ for the end points in time.) We know from our definition of the Fourier transform that we can represent y as

$$y(x,t) = \sum_{n=-\infty}^{\infty} a_n e^{i2\pi f_n t}, \tag{1}$$

or

$$y(x,t) = \sum_{m=-\infty}^{\infty} a_m e^{i2\pi k_m x}, \tag{2}$$

and by extension we can do this process in two dimensions:

$$y(x,t) = \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} a_{nm} e^{i2\pi(f_n t + k_m x)}, \tag{3}$$

where

$$2\pi k_m = \frac{2\pi m}{2x_f} \tag{4}$$

$$2\pi f_n = \frac{2\pi n}{2T} \tag{5}$$

The corresponding spectral density estimate can be calculated from the squared coefficients:

$$\hat{E}(k_m, f_n) = \frac{|a_{nm}|^2}{\Delta k \Delta f} \tag{6}$$

Practicalities.

Suppose we take a time-space data set and Fourier transform it in both directions. Here are some issues that might concern us:

1. *Does it matter whether we Fourier transform time or distance first?* All other things being equal, no. Time and space are orthogonal, and one will not influence the other. To see this consider the following:

$$\int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty} y(x,t)e^{i2\pi kx}\,dx\right]e^{i2\pi ft}\,dt = \int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty} y(x,t)e^{i2\pi ft}\,dt\right]e^{i2\pi kx}\,dx = \quad (7)$$

   (How you demean or detrend might matter, and you can ponder these issues.)

2. *Can we compute a frequency-wavenumber spectrum by Fourier tranforming a frequency spectrum in space?* No. If you look at the above equation, you'll see that you need the full complex structure of the Fourier transformed variables in order to determine the spectrum. You have to retain the phase information. If you computed spectra first, you would suppress some of the signal that you wanted.

3. *When we compute frequency spectra, we usually only plot positive frequencies? Does the same strategy apply for frequency-wavenumber spectra?* No, at a given frequency we can have forward and backward propagating signals, so we'll often want to plot a half plane for frequency with positive and negative wavenumbers, or vice versa.

4. *In practical terms, how do we implement this?* If you Fourier transform in time and space, you'll end up with a domain that should be structured as in Figure **??**. But Matlab will of course line up the frequencies and wavenumbers starting with positive and then negative. You can use the Matlab command "fftshift" to swap the presentation around.

An example Let's suppose that we have a propagating signal. We can generate an artificial signal of the form:

```
H=1000;

dt=0.05; %time interval about 20 times a day
dz=10; %10-m
t=dt:dt:10; %time in days
z=(dz:dz:H)'; %depth in m

%indices for the calculation
iz=1:length(z);
it=1:length(t);

%Generate a propagating signal with no noise.
[tt,zz]=meshgrid(t,z);
sig1=sin(3*pi*zz/H - 2*pi*1/3*tt);
%Change the sign to make propagation go the other way.

data=sig1;

% plot
figure(2)
imagesc(t,z,data);colorbar; axis xy
```

This will show a propagating signal.

To compute a frequency-wavenumber spectrum for this, we just need to Fourier transform in two dimensions. Since this is a noise free case, we can be fairly cavalier about how we approach it.

```
[m,n]=size(data);

fn=1/2/dt;
kn=1/2/dz;

%fundamental frequency and wavenumber
df=1./n./dt;
dk=1./m./dz;


% make frequencies and wavenumbers that run from -Nyquist to + Nyquist
f=[-fliplr(1:(n/2)) 0 (1:(n/2-1))].*df;
k=[-fliplr(1:(m/2)) 0 (1:(m/2-1))].'.*dk;

% Fourier transform in two dimensions
%   here we use fft2 for the 2-d Fourier transform
%   and fftshift to reorder the Fourier transform
st=fftshift(fft2(data))/m/n;

% alternatively you could do this as
st= fftshift(fft(fft(data')'))/m/n;

% turn this into a spectrum
spec=st.*conj(st)./df./dk; %UNITS: (m/s)^2/cpd/cpm

% and plot
figure(3)
imagesc(f,k,log10(spec)); axis xy
colormap(jet)
shg
xlabel('\omega / cpd')
ylabel('m / cpm')

% this plots the full 2-d plane

% But you might want a half plane.  In that case, you should
% scale by a factor of 2:
spec=spec(:,101:200); spec(:,102:200)=2*spec(:,102:200);
imagesc(f(101:200),k,log10(spec)); axis xy
colormap(jet)
shg
xlabel('\omega / cpd')
ylabel('m / cpm')
```

Now what happens when you have noise and need to segment. Let's make a larger data set and chop it up into pieces:

```
t=dt:dt:30; %time in days
z=(dz:dz:H)'; %depth in m

%indices for the calculation
iz=1:length(z);
it=1:length(t);

%Generate a propagating signal with no noise.
[tt,zz]=meshgrid(t,z);
sig1=sin(3*pi*zz/H - 2*pi*1/3*tt) + .5*randn(length(z),length(t));
data=sig1;

% plot
figure(2)
imagesc(t,z,data);colorbar; axis xy
```

To chop this into segments, we have choices to chop in time or space or both.

```
icount=0; clear st;
for i=1:50:501
 icount=icount+1;
 data_use=data(:,i:i+99);
 n_use=size(data_use,2);
 st(:,:,icount)=fftshift(fft2(data_use))/m/n_use;
end

spec=sum(abs(st).^2/df/dk,3);
f=[-fliplr(1:(n_use/2)) 0 (1:(n_use/2-1))].*df;
figure(3)
imagesc(f,k,log10(spec)); axis xy
colormap(jet)
shg
xlabel('\omega / cpd')
ylabel('m / cpm')
```

Interestingly, we can achieve the averaging that we need, either by segmenting in time or in space or in both directions. We do have to be careful about detrending, and we need to pay attention to our windowing strategy.

Variance preserving spectra

Last time we said, that one of the virtues of the properly normalized spectrum is that the area under the curve should represent the signal variance within a specific frequency band:

$$\text{variance in a band} = \int_{f-\Delta f/2}^{f+\Delta f/2} |X(f)|^2 \, df \tag{8}$$

This sounds good as a concept, but in log-log space, it's challenging to figure out what the area under the curve really represents.

That might not bother you, but sometimes that visual representation might seem helpful. Consider a spectrum $S_{xx}(f)$ derived from our Fourier amplitudes $X(f)$. In log space, the area under our curves is a pseudo-variance s_*^2:

$$s_*^2 = \int_{f-\Delta f/2}^{f+\Delta f/2} \log(S_{xx}(f))\, df \tag{9}$$

If we want to plot something that is more directly representative of variance, we can try this:

$$s^2 = \int_{f-\Delta f/2}^{f+\Delta f/2} S_{xx}(f)\, df = \int_{f-\Delta f/2}^{f+\Delta f/2} f S_{xx}(f) d[\log(f)] \tag{10}$$

This means that instead of plotting log$S$ vs log(f), we could plot $fS_{xx}(f)$ in linear space vs f in log space. This is especially useful for features that have strong peaks not exactly at the lowest frequency.

Here's an example, using the red noise that we started with earlier:

```
lambda=10; % 10 m wavelength
V=0.3; % 0.3 m/s propagation
n2s=0.2; % noise-to-signal ratio
time=(1:10000)';
x=n2s*cumsum(randn(10000,1))+cos(2*pi/lambda*V*time);


bb=reshape(b,500,10000/500);
fbb=fft(detrend(bb));
amp=abs(fbb(1:251,:)).^2 / 500;
sbb=mean(amp,2);
sbb(2:250)=sbb(2:250)*2;

subplot(2,1,1)
loglog(0:250,sbb,'LineWidth',3)
xlabel('Frequency','FontSize',14)
ylabel('(units^2)/frequency','FontSize',14)

subplot(2,1,2)
semilogx(0:250,(0:250)'.*sbb,'LineWidth',3)
xlabel('Frequency','FontSize',14)
ylabel('(units^2)','FontSize',14)
```

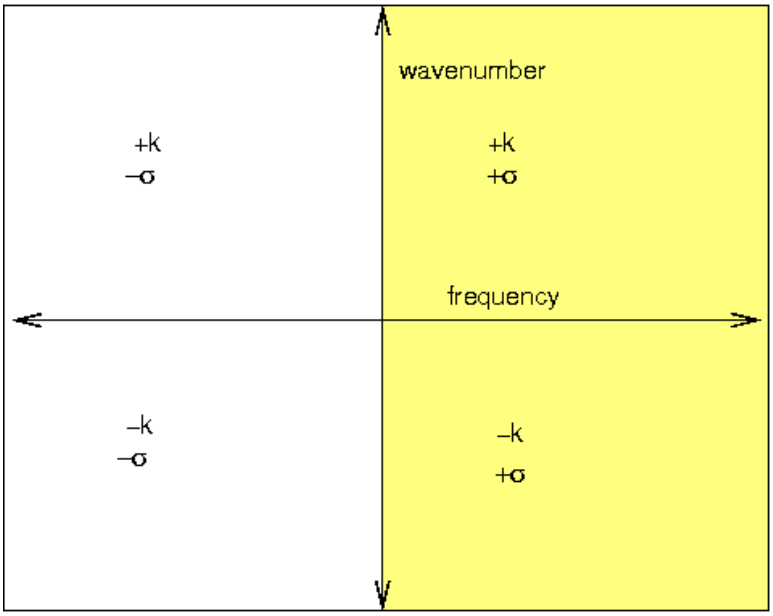

Figure 1: Fourier transform of time/space domain to form frequency/wavenumber domain. Note that $+k, +f$ is the complex conjugate of $-k, -f$ and similarly $+k, -f$ is the complex conjugate of $-k, +f$ so we normally plot only half the domain.