7 Investigation of Cross-Frequency Coupling in a Local Field Potential

Synopsis

Data 100 s of local field potential data sampled at 1000 Hz.

Goal Characterize the coupling between rhythms of different frequency.

Tools Hilbert transform, analytic signal, instantaneous phase, cross-frequency coupling.

7.1 Introduction

7.1.1 Background

In Chapter 5, we focused on the coherence between voltage activity recorded at two electrodes. The coherence is a measure of association between rhythms at the same frequency. In this chapter, we continue our study of field data but now focus on local field potential (LFP) recordings. The LFP is a measure of local population neural activity, produced from small aggregates of neurons [15]. In these data, we examine the association between rhythms of *different* frequencies.

In general, lower-frequency rhythms have been observed to engage larger brain areas and modulate spatially localized fast oscillations [16–20]. This cross-frequency coupling (CFC) between the power (or amplitude) of high-frequency rhythms and the phase of low-frequency rhythms has been observed in many brain regions, has been shown to change in time with task demands, and has been proposed to serve a functional role in working memory, neuronal computation, communication, and learning [21]. Although the cellular and dynamic mechanisms of specific rhythms associated with CFC are relatively well understood, the mechanisms governing interactions between different frequency rhythms and the appropriate techniques for measuring CFC remain active research areas. Although we consider only a single electrode recording here, note that these techniques can be extended to association measures between electrodes as well.

7.1.2 Case Study Data

We are approached by a collaborator recording the local field potential (LFP) from rat hippocampus. She has implanted a small bundle of electrodes, which remain (chronically) implanted as the rat explores a large circular arena. She is interested in assessing the association between different frequency rhythms of the LFP, and more specifically whether an association between different frequency rhythms exists as the rat explores the arena. To address this question, she has provided us with 100 s of LFP data recorded during the experiment (i.e., while the rat spontaneously explored the arena).

7.1.3 Goal

Our goal is to assess the associations between different frequency rhythms recorded in the LFP. To do so, we analyze the LFP data by computing the cross-frequency coupling of the time series. We construct two CFC measures that characterize how the phase of a low-frequency signal modulates the amplitude envelope of a high-frequency signal. This chapter assumes some knowledge of neural rhythms and their assessment. If the material seems overly dense, consult the earlier case studies in chapters 3–6.

7.1.4 Tools

In this chapter, we develop two CFC measures. We introduce the concepts of the Hilbert transform, analytic signal, instantaneous phase, and amplitude envelope.

7.2 Data Analysis

7.2.1 Visual Inspection

To access the data for this chapter, visit

`http://github.com/Mark-Kramer/Case-Studies-Kramer-Eden`

and download the file `Ch7-LFP-1.mat`. Let's begin with visual inspection of the LFP data. To do so, we first load the LFP data into MATLAB and plot it:

```
load('Ch7-LFP-1.mat')          %Load the LFP data,
plot(t,LFP)                    %... and plot it,
xlabel('Time [s]');            %... with axes labeled.
ylabel('Voltage [mV]')
```

Within an example 1 s interval, rhythmic activity in the LFP is apparent (figure 7.1). Visual inspection immediately suggests a dominant low-frequency rhythm interspersed with smaller-amplitude blasts of high-frequency activity.

> **Q:** Approximate the rhythmic activity by visual inspection of the LFP data plotted in figure 7.1. What is the frequency of the large-amplitude rhythm? Do you observe high-frequency activity? If so, where in time, and at what approximate frequency? What is the sampling frequency of these data? If you were to compute the spectrum of the entire dataset (100 s of LFP), what would be the Nyquist frequency and the frequency resolution? *Hint:* In figure 7.1, consider the times near 4.35 s and 4.5 s. Do you see the transient fast oscillations?

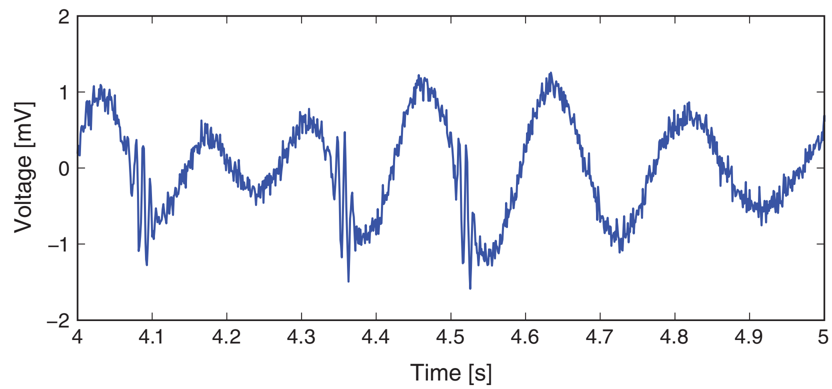

Figure 7.1
Example trace of LFP data.

Note: If you have no idea how to address these questions, or if the terminology seems completely unfamiliar, consider reviewing the case studies in chapters 3 and 4. In those chapters, the notions of rhythms are developed in much greater detail, as well as the spectrum, Nyquist frequency and frequency resolution.

7.2.2 Spectral Analysis

Visual inspection of the LFP data suggests that multiple rhythms appear. To further characterize this observation, we compute the spectrum of the LFP data.[1] We analyze the entire 100 s of data and compute the spectrum with a Hanning taper (see chapter 4). In MATLAB,

```
load('Ch7-LFP-1.mat')              %Load the LFP data.
dt = t(2)-t(1);                    %Define the sampling interval.
T = t(end);                        %Define the duration of data.
N  = length(LFP);                  %Define no. of points in data.

x = hann(N).*transpose(LFP);       %Multiply data by Hanning taper.
xf = fft(x-mean(x));               %Compute Fourier transform of x.
Sxx = 2*dt^2/T *(xf.*conj(xf));    %Compute the spectrum.
Sxx = Sxx(1:N/2+1);                %Ignore negative frequencies.

df = 1/max(T);                     %Define frequency resolution.
fNQ = 1/dt/2;                      %Define Nyquist frequency.
```

1. We could instead write the *sample* spectrum because we use the observed data to estimate the theoretical spectrum that we would see if we kept repeating this experiment. However this distinction is not essential to the discussion here.

```
faxis = (0:df:fNQ);             %Construct frequency axis.
plot(faxis, 10*log10(Sxx))      %Plot spectrum vs frequency.
xlim([0 200]); ylim([-80 0])    %Set frequency & decibel range.
xlabel('Frequency [Hz]')        %Label axes.
ylabel('Power [ mV^2/Hz]')
```

Q: Does the use of the Fourier transform and Hanning taper make sense? Do the expressions for the frequency resolution (`df`), Nyquist frequency (`fNQ`), and spectrum (`Sxx`) make sense?

A: If you answered yes in all cases, you're right. If not, consider reviewing the case studies in chapters 3 and 4.

The resulting spectrum, shown in figure 7.2, reveals two intervals of increased power spectral density. The lowest-frequency peak at 6 Hz is also the largest and corresponds to the slow rhythm we observe dominating the signal through visual inspection of figure 7.1. At higher frequencies, we find an additional broadband peak at approximately 80–120 Hz. These spectral results support our initial visual inspection of the signal; there exist both low- and high-frequency activities in the LFP data. We now consider the primary question of interest: Do these different frequency rhythms exhibit associations?

7.2.3 Cross-Frequency Coupling

To assess whether different frequency rhythms interact in the LFP recording, we implement a measure to calculate CFC. The idea of CFC analysis is to determine whether a

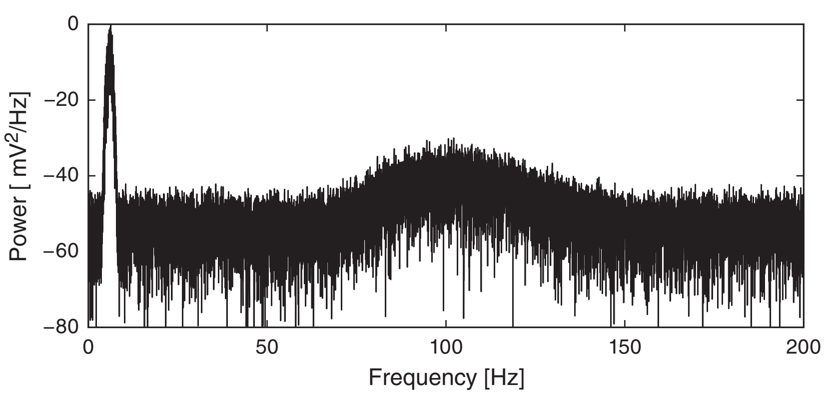

Figure 7.2
Spectrum of LFP data.

relation exists between the phase of a low-frequency signal and the envelope or amplitude of a high-frequency signal. In general, computing CFC involves three steps. Each step contains important questions and encompasses entire fields of study. Our goal in this section is to move quickly forward and produce a procedure we can employ, investigate, and criticize. Continued study of CFC—and the associated nuances of each step—is an active area of ongoing research.

CFC analysis steps

1. Filter the data into high- and low-frequency bands.
2. Extract the amplitude and phase from the filtered signals.
3. Determine if the phase and amplitude are related.

1. Filter the Data into High- and Low-Frequency Bands. The first step in the CFC analysis is to filter the data into two frequency bands of interest. The choice is not arbitrary: the separate frequency bands are motivated by initial spectral analysis of the LFP data. In this case, we choose the low-frequency band as 5–7 Hz, consistent with the largest peak in the spectrum, and the high-frequency band as 80–120 Hz, consistent with the second-largest broadband peak (figure 7.2). To consider alternative frequency bands, the same analysis steps would apply.

There are many options to perform the filtering. To do so requires us to design a filter that ideally extracts the frequency bands of interest without distorting the results. Here, we apply a finite impulse response (FIR) filter (see chapter 6). In MATLAB,

```
dt = t(2)-t(1);              %Define the sampling interval.
Fs = 1/dt;                   %Define the sampling frequency.
fNQ = Fs/2;                  %Define the Nyquist frequency.
                             %For low-frequency interval,
Wn = [5,7]/fNQ;              %...set the passband,
n  = 100;                    %...and filter order,
b  = fir1(n,Wn);             %...build bandpass filter.
Vlo = filtfilt(b,1,LFP);     %...and apply filter.
                             %For high-frequency interval,
Wn = [80,120]/fNQ;           %...set the passband,
n  = 100;                    %...and filter order,
b  = fir1(n,Wn);             %...build bandpass filter.
Vhi = filtfilt(b,1,LFP);     %...and apply filter.
```

For each frequency band, we specify a frequency interval of interest by defining the low- and high-cutoff frequencies in the vector `Wn`. This vector contains two elements, and we divide this vector by the Nyquist frequency (`fNQ`). In this way, we specify the passband of the filter on the interval between 0 and 1, where 1 represents the Nyquist frequency. We then set the filter order (`n`) and design the filter using the MATLAB function `fir1`. Finally, we apply the filter using the MATLAB function `filtfilt`, which performs zero-phase filtering by applying the filter in both the forward and reverse directions (see chapter 6). We note that the filtering procedure is nearly the same in both frequency bands; the only change is the specification of the frequency interval of interest.

> If the filtering procedure seems puzzling consider completing the case study in chapter 6. That chapter describes filtering in detail.

To understand the impact of this filtering operation on the LFP, let's plot the results (figure 7.3). As expected, the low-frequency band captures the large-amplitude rhythm dominating the LFP signal, while the higher-frequency band isolates the brief bursts of faster activity.

2. Extract the Amplitude and Phase from Filtered Signals. The next step in the CFC procedure is to extract the phase of the low-frequency signal and the amplitude envelope (or simply, amplitude) of the high-frequency signal. To gain some intuition for this operation, let's consider the amplitude and phase for the example signals in figure 7.3. We plot the low-frequency signal and its phase in figure 7.4a. As time progresses, the phase increases nearly linearly from $-\pi$ to π. At π, the phase jumps suddenly to $-\pi$. This apparent discontinuity is imposed by the space on which the phase evolves: a circle.

As another example of such a space, consider the time on a 24-hour clock changing suddenly from 23:59 to 00:00. In this case, the measurement has suddenly shifted discontinuously, but time has not; this discontinuity appears because we choose to measure time in 24-hour intervals (for good reason). The same is true of the phase that we choose to

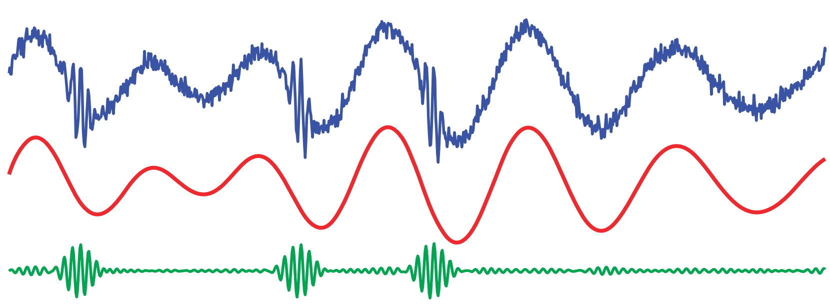

Figure 7.3
Examples of filtering the LFP data. Original LFP signal (*blue*) is filtered into a low-frequency band (*red*) and a high-frequency band (*green*). Scale bar indicates 0.1 s.

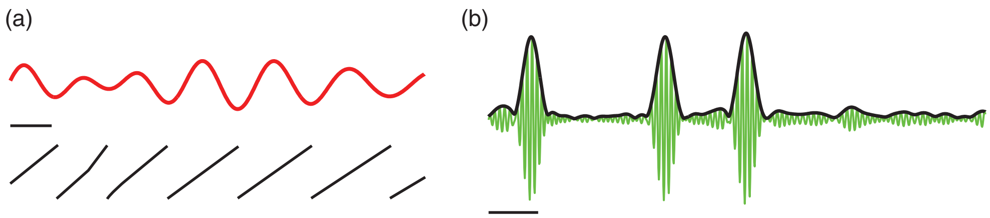

Figure 7.4

Examples of phase and amplitude of filtered LFP data. (*a*) Phase (*black*) of low-frequency signal (*red*) increases from $-\pi$ to π over time. (*b*) Amplitude envelope (*black*) outlines deviations of high-frequency signal (*green*). Scale bar indicates 0.1 s.

measure, from $-\pi$ to π. The two endpoints ($-\pi$ and π) actually touch on the space of a circle but appear to discontinuously jump when plotted on the plane.

The amplitude envelope (figure 7.4b) outlines the extent of deviations of the high-frequency signal. Notice that the envelope fluctuates much less rapidly than the underlying high-frequency signal.

To compute CFC, we compare the two signals we've extracted from the data, the phase of the low-frequency activity and the amplitude envelope of the high-frequency activity (figure 7.4). How do we actually extract the phase and amplitude signals from the data? There are a variety of options to do so, and we choose here to employ the *analytic signal* approach, which allows us to estimate the instantaneous phase and amplitude envelope of the LFP.

The first step in computing the analytic signal is to compute the *Hilbert transform*. We begin with some notation. Define x as a narrowband signal (i.e., a signal with most of its energy concentrated in a narrow frequency range,[2] e.g., the low- or high-frequency band filtered signals in figure 7.3). Then the Hilbert transform of x, let's call it y, is

$$y = H(x).$$

It's perhaps more intuitive to consider the effect of the Hilbert Transform on the frequencies f of x,

$$H(x) = \begin{cases} -\pi/2 \text{ phase shift} & \text{if } f > 0, \\ 0 \text{ phase shift} & \text{if } f = 0, \\ \pi/2 \text{ phase shift} & \text{if } f < 0. \end{cases}$$

> The Hilbert transform $H(x)$ of the signal x produces a phase shift of ± 90 degrees for $\mp$ frequencies of x.

2. The impact of this narrowband assumption on CFC estimates remains an open research topic. One might consider, for example, the meaning and implications of estimating phase from a broadband signal, and the impact on subsequent CFC results.

The Hilbert Transform can also be described in the time domain, although its representation is hardly intuitive (see the appendix at the end of this chapter). Then the analytic signal z is

$$z = x + i\,y = x + i\,H(x). \tag{7.1}$$

The effect of the Hilbert transform is to remove negative frequencies from z. As it stands, this is not obvious. To get a sense for why this is so, let's consider a simple example.

What Does the Hilbert Transform Do? Let x_0 be a sinusoid at frequency f_0,

$$x_0 = 2\cos(2\pi f_o\, t) = 2\cos(\omega_0\, t), \tag{7.2}$$

where to simplify notation we have defined $\omega_0 = 2\pi f_o$. We know from Euler's formula that

$$x_0 = e^{i\omega_0 t} + e^{-i\omega_0 t}. \tag{7.3}$$

The real variable x_0 possesses both a positive and a negative frequency component (i.e., ω_0 and $-\omega_0$). So, the spectrum has two peaks (figure 7.5). For real signals, which include nearly all recordings of brain activity, the negative frequency component is redundant, and we usually ignore it. However, the negative frequency component still remains.

By definition, the effect of the Hilbert transform is to induce a phase shift. For positive frequencies, the phase shift is $-\pi/2$. We can produce this phase shift by multiplying the positive frequency part of the signal by $-i$.

Q: Why does a phase shift of $-\pi/2$ correspond to multiplication by $-i$?

A: Consider the complex exponential $e^{i\omega_0 t}$, which consists of only a positive frequency component (ω_0). This signal shifted in phase by $-\pi/2$ corresponds to the new signal $e^{i(\omega_0 t-\pi/2)}$, which simplifies to

$$e^{i(\omega_0 t-\pi/2)} = e^{i\omega_0 t}e^{-i\pi/2} = e^{i\omega_0 t}\Big(\cos(\pi/2) - i\sin(\pi/2)\Big) = e^{i\omega_0 t}(-i).$$

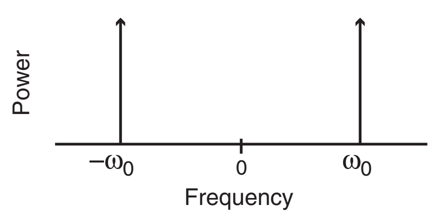

Figure 7.5
Spectrum of a sinusoid has two peaks, at positive and negative frequencies.

Notice the result simplifies to the original complex exponential $e^{i\omega_0 t}$ multiplied by $-i$. This shows that the $-\pi/2$ phase shift corresponds to multiplication of the positive frequency component ($e^{i\omega_0 t}$) by $-i$.

Q: Can you show that a $\pi/2$ phase shift corresponds to multiplication by i?

This analysis shows that we can represent the Hilbert Transform of x at frequency f as

$$H(x) = \begin{cases} -ix & \text{if } f > 0, \\ 0 & \text{if } f = 0, \\ ix & \text{if } f < 0. \end{cases}$$

Therefore, the Hilbert transform of $x_0 = e^{i\omega_0 t} + e^{-i\omega_0 t}$ becomes

$$y_0 = H(x_0) = -ie^{i\omega_0 t} + ie^{-i\omega_0 t}.$$

In this equation, we multiply the positive frequency part of x_0 (i.e., $e^{i\omega_0 t}$) by $-i$, and the negative frequency part of x_0 (i.e., $e^{-i\omega_0 t}$) by i. Simplifying this expression using Euler's formula, we find

$$y_0 = 2\sin(\omega_0 t). \tag{7.4}$$

Q: Can you perform this simplification? In other words, can you show that $-ie^{i\omega_0 t} + ie^{-i\omega_0 t} = 2\sin(\omega_0 t)$?

The Hilbert Transform of x_0 (a cosine function) is a sine function. We could perhaps have guessed this: sine is a 90-degree ($\pi/2$) phase shift of cosine.

We are now ready to define the analytic signal (z_0) for this example. Using the expressions for x_0 and y_0 and Euler's formula, we find

$$z_0 = x_0 + i\,y_0 = 2\cos(\omega_0 t) + i\,2\sin(\omega_0 t) = 2e^{i\omega_0 t}.$$

Notice that this analytic signal z_0 contains no negative frequencies; as mentioned, the effect of the Hilbert Transform is to eliminate the negative frequencies from x. The spectrum of this signal consists of a single peak at ω_0, compared to the two peaks at $\pm\omega_0$ in the original signal x (figure 7.5). In this sense, the analytic signal (z_0) is simpler than the original signal x_0. To express the original signal x_0 required two complex exponential functions—see (7.3)—compared to only one complex exponential required to express the corresponding

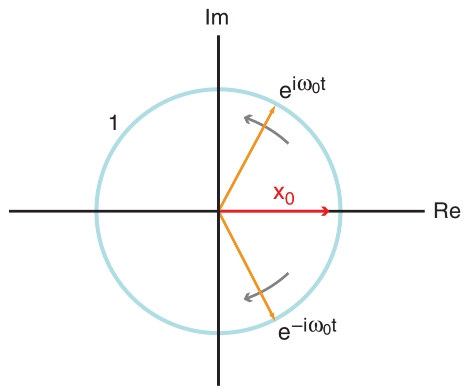

Figure 7.6

Cartoon of real time series x_0 (*red*) plotted in the complex plane. The two complex exponentials (*orange*) cancel in their imaginary parts to produce x_0. Over time, the two complex exponentials rotate (*gray*) so that their imaginary parts cancel at each moment in time and x_0 remains on the real axis. Blue circle indicates radius 1.

analytic signal z_0. There's an interesting geometrical interpretation of this. Consider plotting x_0 in the complex plane (figure 7.6). Because x_0 is real, this quantity evolves in time along the real axis. To keep x_0 on the real axis, the two complex exponentials that define x_0 (i.e., $e^{i\omega_0 t}$ and $e^{-i\omega_0 t}$) rotate in opposite directions along the unit circle. By doing so, the imaginary components of these two vectors cancel, and we're left with a purely real quantity x_0.

Q: The phase of a complex quantity is the angle with respect to the real axis in the complex plane. What is the angle of x_0 in figure 7.6?

2. Extracted the Amplitude and Phase from Filtered Signal (Continued). Having developed some understanding of the Hilbert Transform, let's now return to the LFP data of interest here. It's relatively easy to compute the analytic signal and extract the phase and amplitude in MATLAB:

```
phi=angle(hilbert(Vlo));%Compute phase of low-freq signal.
amp=abs(hilbert(Vhi));  %Compute amplitude of high-freq signal.
```

These operations require just two lines of code. But beware of the following.

Alert! The command `hilbert(x)` returns the analytic signal of x, not the Hilbert transform of x.

To extract the phase, we apply the MATLAB function `angle` to the analytic signal of the data filtered in the low-frequency band (variable `Vlo`). We then extract the amplitude of the analytic signal of the data filtered in the high-frequency band (variable `Vhi`) by computing the absolute value (MATLAB function `abs`).

To summarize, in this step we apply the Hilbert transform to create the analytic signal and get the phase or amplitude of the bandpass-filtered data.

3. Determine if the Phase and Amplitude are Related. As with the previous steps, we have at our disposal a variety of procedures to assess the relation between the phase (of the low-frequency signal) and amplitude (of the high-frequency signal). We do so here in two ways.

Method 1: Phase-amplitude plot. To start, define the two-column phase-amplitude vector,

$$\begin{pmatrix} \phi(1) & A(1) \\ \phi(2) & A(2) \\ \phi(3) & A(3) \\ \vdots & \vdots \end{pmatrix},$$

where $\phi(i)$ is the phase of the low-frequency band activity at time index i, and $A(i)$ is the amplitude of the high-frequency band activity at time index i. In other words, each row defines the instantaneous phase and amplitude of the low- and high-frequency filtered data, respectively.

We now use this two-column vector to assess whether the phase and amplitude envelope are related. Let's begin by plotting the average amplitude versus phase. We divide the phase interval into bins of size 0.1 beginning at $-\pi$ and ending at π. The choice of bin size is somewhat arbitrary; this choice will work fine, but you might consider the impact of other choices.

For a chosen phase bin, determine the time indices where the phase ϕ falls within this phase bin; we can think of these times as indexing specific rows of the two-column phase-amplitude vector. Then compute the average amplitude at these same time indices. The result is the average amplitude for phases that lie within the chosen phase bin. Finally, repeat this procedure for all phase bins. We implement these steps in MATLAB and plot the results as follows:

```
p_bins = (-pi:0.1:pi);               %Define the phase bins.
a_mean = zeros(length(p_bins)-1,1); %Vector for average amps.
p_mean = zeros(length(p_bins)-1,1); %Vector for phase bins.
```

```
for k=1:length(p_bins)-1            %For each phase bin,
    pL = p_bins(k);                 %... lower phase limit,
    pR = p_bins(k+1);               %... upper phase limit.
    indices=find(phi>=pL & phi<pR); %Find phases falling in bin,
    a_mean(k) = mean(amp(indices)); %... compute mean amplitude,
    p_mean(k) = mean([pL, pR]);     %... save center phase.
end
```

Q: Consider the plot of average amplitude versus phase (figure 7.7a). Does this result suggest CFC occurs in these data?

A: We plot in figure 7.7a the phase bins (variable `p_mean`) versus the mean amplitude in each bin (variable `a_mean`). Visual inspection of this phase-amplitude plot suggests that the amplitude of the high-frequency signal depends on the phase of the low-frequency signal. In particular, we note that when the phase is near a value of 2 radians, the amplitude tends to be large, while at other phases the amplitude is smaller. This conclusion suggests that CFC does occur in the data; the high-frequency amplitude depends on the low-frequency phase.

Q: If no CFC occurred in the data, what would you expect to find in the plot of average amplitude versus phase?

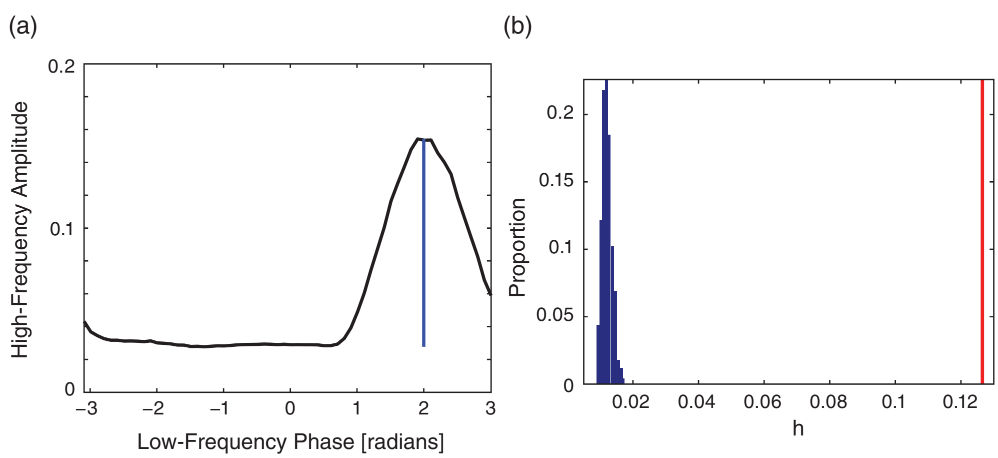

Figure 7.7

Phase-amplitude plot between low-frequency phase and high-frequency amplitude envelope. (*a*) Average amplitude of the high-frequency band versus phase of low-frequency band. Value of statistic *h* is length of the blue vertical line. (*b*) Distribution of *h* values for surrogate data (*bars*) versus observed value of *h* (*red vertical line*).

As a basic statistic to capture the extent of this relation, we compute the difference between the maximum and minimum of the average amplitude envelope over phases (blue vertical line in figure 7.7a). Let's assign this difference the label h. In MATLAB,

```
%Difference between max and min modulation.
h = max(a_mean)-min(a_mean);
```

We find a value of $h = 0.1265$. This value, on its own, is difficult to interpret. Is it bigger or smaller than we expect? To assess the significance of h, let's generate a surrogate phase-amplitude vector by resampling without replacement the amplitude time series (i.e., the second column of the phase-amplitude vector). Resampling is a powerful technique that we have applied in our analysis of other case study data.[3] By performing this resampling, we reassign each phase an amplitude chosen randomly from the entire 100 s LFP recording. We expect that if CFC does exist in these data, then the timing of the phase and amplitude vectors will be important; for CFC to occur, the amplitude and phase must coordinate in time. By disrupting this timing in the resampling procedure, we expect to eliminate the coordination between amplitude and phase necessary to produce CFC.

For each surrogate phase-amplitude vector, we compute the statistic h. To generate a distribution of h values, let's repeat 1,000 times this process of creating surrogate data through resampling and computing h. In MATLAB,

```
n_surrogates = 1000;                    %Define no. of surrogates.
hS = zeros(n_surrogates,1);             %Vector to hold h results.
for ns=1:n_surrogates;                  %For each surrogate,
    ampS = amp(randperm(length(amp)));  %Resample amplitude,
    p_bins = (-pi:0.1:pi);              %Define the phase bins.
    a_mean = zeros(length(p_bins)-1,1); %Vector for average amps.
    p_mean = zeros(length(p_bins)-1,1); %Vector for phase bins.
    for k=1:length(p_bins)-1            %For each phase bin,
      pL = p_bins(k);                   %...lower phase limit,
      pR = p_bins(k+1);                 %...upper phase limit.
      indices=find(phi>=pL & phi<pR);   %Find phases in bin,
      a_mean(k) = mean(ampS(indices));  %...compute mean amp,
      p_mean(k) = mean([pL, pR]);       %...save center phase.
    end
    hS(ns) = max(a_mean)-min(a_mean);   %Store surrogate h.
end
```

In this code, we first define the number of surrogates (variable `n_surrogates`) and a vector to store the statistic h computed for each surrogate phase-amplitude vector (variable `hS`). Then, for each surrogate, we use the MATLAB function `randperm` to randomly

3. For a review of resampling, applied in a different case study, see chapter 2.

permute the amplitude vector without replacement. We then use this permuted amplitude vector (variable `ampS`) and the original phase vector (variable `phi`) to compute h; this last step utilizes the MATLAB code developed earlier to compute h for the original (unpermuted) data.

> **Q:** Do you notice any inefficiencies in this code? If so, how would you modify the code to avoid these inefficiencies (e.g., repeated calculations of unchanging quantities)?

Figure 7.7b shows the results of this resampling procedure as a histogram of the variable `hS`. The value of h computed from the original data lies far outside the distribution of surrogate h values. To compute a p-value, we determine the proportion of surrogate h values greater than the observed h value:

```
p = length(find(hS > h))/length(hS);    %Compute p-value.
```

We find a p-value that is very small; there are no surrogate values of h that exceed the observed value of h. We therefore conclude that in this case the observed value of h is significant. As a statistician would say, we reject the null hypothesis of *no* CFC between the phase of the 5–7 Hz low-frequency band and the amplitude of the 80–120 Hz high frequency band.

Method 2 (advanced): A generalized linear model approach. The method described to assess the relation between the phase (of the low-frequency activity) and amplitude (of the high-frequency activity) is common in the neuroscience literature (as reviewed in [22]). However, alternatives exist. Here, we outline a method based on the concept of a *generalized linear model* (GLM). This method involves more advanced statistical concepts than the phase-amplitude plot described in method 1. We discuss generalized linear models in more detail in chapter 9. Here we only outline the highlights of a procedure to create a GLM-CFC statistic. A more detailed description of this procedure, which includes many simulated examples, may be found in [23].

The idea of the GLM-CFC statistic is to build a statistical model that describes the amplitude (A) of the high-frequency signal as a function of the phase (ϕ) of the low-frequency signal. To create this model, we first assume a distribution for the amplitudes. We choose a gamma distribution because the amplitudes are real and positive, and typically confined to a limited interval with infrequent large events. We must then choose a function that links the amplitudes to the phases. Here we choose the log link,

$$\log(\mu) = \beta X,$$

where μ is the expected value of the amplitudes, X is a function of the phases, and β are the unknown model coefficients to determine. The number of coefficients depends on the model choice. The log link function is common for GLMs using a gamma distribution and leads to models where predictors have multiplicative effects on the response.

We construct two GLMs to fit the amplitude A as a function of the phase ϕ. In the first, we assume that the amplitude does not depend on the phase; we label this the *null model* because this model represents the null hypothesis of no CFC. In this case, X is a constant and does not depend on the phase. We set $X = 1$, and there is a single unknown coefficient to estimate, which we label β_0. Conceptually, the null model estimates the average amplitude across all phases.

In the second model, which we label the *spline model*, we use cardinal splines to fit a smooth function for the expected amplitude as a function of the phase. Cardinal splines are smooth, piecewise-connected third-order polynomial functions that are defined by a set of control points. The advantage of the cardinal spline is that it is capable of approximating any continuous functional relation between phase and amplitude with a small number of parameters [24]. These parameters, the control points, are directly interpretable as the expected amplitude at a specific set of phase values. The spline fit then smoothly interpolates between the estimated control points.

To fit a spline model, we generate X by applying a set of cardinal spline basis functions to the observed phase values, ϕ, at each time step. We select a number of control points, n, and space these evenly between 0 and 2π. The value of the spline estimate at any phase is determined by the two nearest control points to the left and the two nearest control points to the right, where the control point values below zero or above 2π are taken modulo 2π. In this way, the spline function is defined over the circular topology of phase values. We must also select the tension parameter for the spline, which controls the curviness of the function at the control points. Here we choose a tension parameter of 0.5, a standard choice. In this case, X is a matrix consisting of n independent variables (the control points), and therefore n unknown model coefficients to estimate, which we label β_S. Exponentiating these estimates represents the multiplicative effect of the phase on the expected value of the amplitude envelope at each control point.

Principled methods exist to determine the number of control points (or knots) n in the spline model (e.g., the Akaike information criterion (AIC); see [23]). An alternative method for selecting the number of knots is to employ prior physical or observational knowledge about the system. For example, if we believe that the amplitude increases at one or two broad phase intervals, then we may choose to utilize 4 or 10 knots, respectively, in the spline model. However, if instead we believe that the amplitude increases in a sharp phase interval, then we may choose to utilize more knots (e.g., more than 10). By selecting too few knots, we may fail to detect amplitude increases restricted to narrow phase intervals, and by selecting too many knots, we may lose statistical power. In general, by selecting the number of knots, we impose a class of models in the GLM-CFC procedure, and we may do so using either quantitative techniques (e.g., AIC) or prior physical knowledge about the system.

After estimating the unknown coefficients β_0 and β_S of the two models, we then compute the predicted values for the amplitude using the spline model (A_S) and the null model (A_0) at 100 phase values evenly spaced between $-\pi$ and π. As a scalar statistic to characterize the CFC we compute,

$$r = \max\big[\text{abs}\big[1 - A_S/A_0\big]\big], \tag{7.5}$$

which is the maximum absolute fractional change between the spline and null models. This statistic is therefore simply interpreted as the largest proportional (or percentage) change between the null and spline models. A large value of r is indicative of CFC; when r is large, the amplitude at some phase in the spline model differs from the constant amplitude of the null model.

An advantage of this modeling approach is that we can use the GLM to generate confidence intervalss for the statistic r. To do so, we use the estimated coefficients β_S to generate 10,000 normally distributed samples of the coefficients (β_S^j, where $j = \{0, 1, 2, \ldots, 10,000\}$). For each j, we then compute the predicted values for the amplitude using the spline model (A_S^j) and reestimate the amplitude for the null model (A_0^j) as the mean of A_S^j. Finally, we compute the measure r_S^j for each j. From the resulting distribution of r_S^j, we determine the 0.025 and 0.975 quantiles. In this way, we use the surrogate distribution to define the 95% confidence intervals for the statistic r.

The statistic r provides a single scalar value representative of CFC between the phase and amplitude time series, and the associated confidence intervals provides a range of certainty in the statistic. To visualize CFC, we plot the predicted amplitude as a function of phase for the null and spline GLMs, and the pointwise 95% confidence bounds of the predicted amplitude values for both models. This provides a graphical representation of the differences between the two models; strong CFC at some phase results in large differences between the two models.

To summarize, we measure CFC using the single quantity r in (7.5) with a corresponding 95% confidence intervals determined from the GLM. This measure represents the largest deviation between the null model (which allows no variation in amplitude with phase) and the spline model (which permits variation of amplitude with phase). To visualize CFC as a function of phase, we plot both models with corresponding pointwise 95% confidence intervalss.

To compute the GLM-CFC statistic in MATLAB, download the file `GLM_CFC.m` from

`http://github.com/Mark-Kramer/Case-Studies-Kramer-Eden`

This file defines a MATLAB function to compute the GLM-CFC statistic. We provide as input to this function the low-frequency signal, high-frequency signal, and the number of control points to use in the spline model. The function returns the statistic r and its 95% confidence intervals. In MATLAB,

```
nCtlPts = 8;                          %Define no. of control points,
[r,r_CI]=GLM_CFC(Vlo,Vhi,nCtlPts);%...and compute r.
```

Here we have chosen to use eight control points, and analyzed the low- (variable `Vlo`) and high- (variable `Vhi`) frequency signals, as defined, previously. The function returns for the GLM-CFC statistic $R = 1.73$, with 95% confidence intervals $\left[1.71, 1.76\right]$.

In addition to returning the statistic r and its 95% confidence intervals, the function also produces the plot in figure 7.8. Graphically, the approximately 173% maximal deviation between the two models (corresponding to $r = 1.73$) is represented by the height of the blue vertical line at a phase near 2 radians. The statistic r is quite large, with a 95% confidence intervals that exceeds zero, in support of the conclusion that CFC occurs between the chosen low- and high-frequency bands. We note that these results are consistent with the conclusions from method 1 (the phase-amplitude plot) as well, which inspires further confidence.

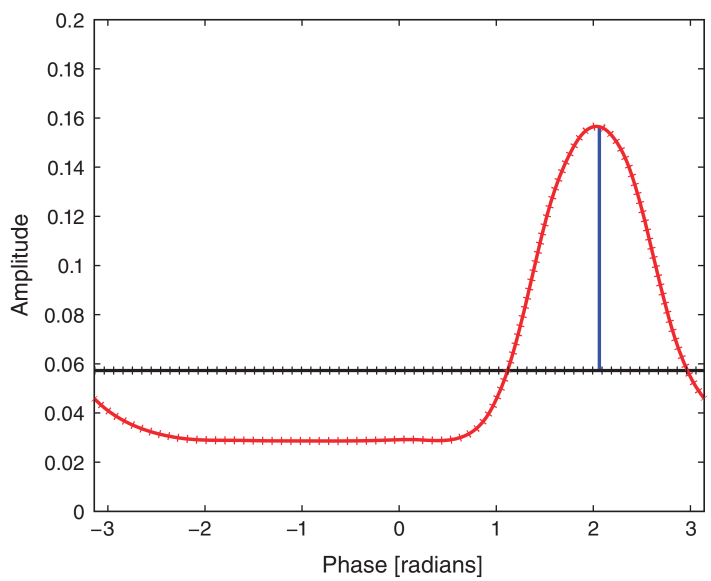

Figure 7.8

GLM fits to high-frequency band amplitude as a function of low-frequency band phase. Fits of the null model (*black*) and spline model (*red*). Means are solid curves; 95% confidence intervals are dotted curves, which remain close to the means. Blue vertical line indicates largest difference between the two models.

Summary

In this chapter, we considered techniques to characterize cross-frequency coupling (CFC), associations between rhythmic activity observed in two different frequency bands. To do so, we introduced the Hilbert transform, which can be used to compute the instantaneous phase and amplitude of a signal. We focused on characterizing the relation between the phase of low-frequency band activity (5–7 Hz) and the amplitude of high-frequency band activity (100–140 Hz) using two approaches. In one approach, we computed the average amplitude at each phase and determined the extent of the variability (or wiggliness). In the other approach, we utilized the GLM framework to develop a statistical model of the data.

For the LFP data of interest here, we found evidence for CFC between the two frequency bands using both methods. Importantly, these results also appear consistent with visual inspection of the unfiltered data. Careful inspection of the example in figure 7.1 suggests that CFC does in fact occur in these data. In general, such strong CFC, visible to the naked eye in the unprocessed LFP data, is unusual. Instead, data analysis techniques are required to detect features not obvious through visual inspection alone. Developing techniques to assess CFC and understanding the biophysical mechanisms of CFC and implications for function, remain active research areas.

In developing these approaches, we utilized expertise and procedures developed in other chapters. In particular, we relied on the notions of frequency and power, amplitude and phase, filtering, resampling, and generalized linear models. Such a multifaceted approach is typical in the analysis of neural data, where we leverage the skills developed in analyzing other datasets.

Problems

7.1. Load the file `Ch7-LFP-2.mat` available at

`http://github.com/Mark-Kramer/Case-Studies-Kramer-Eden`

into MATLAB. You will find two variables. The variable `LFP` corresponds to an LFP recording. The variable `t` corresponds to the time axis, in units of seconds. Use these data to answer the following questions.

a. Visualize the time series data. What rhythms do you observe? Do you detect evidence for CFC in your visualizations?

b. Plot the spectrum versus frequency for these data. Are the dominant rhythms in the spectrum consistent with your visual inspection of the data?

c. Apply the two CFC methods developed in this chapter to these data. In doing so, you must choose the low-frequency and high-frequency bands. What choices will you make, and why? What do you find using the two CFC methods?

d. Describe (in a few sentences) your spectrum and CFC results, as you would to a colleague or collaborator.

7.2. Load the dataset `Ch7-LFP-3.mat` available at

`http://github.com/Mark-Kramer/Case-Studies-Kramer-Eden`

into MATLAB. You will find two variables. The variable `LFP` corresponds to an LFP recording. The variable `t` corresponds to the time axis, in units of seconds. Use these data to answer the following questions.

a. Visualize the time series data. What rhythms do you observe? Do you detect evidence for CFC in your visualizations? Note: Look carefully at these traces.

b. Plot the spectrum versus frequency for these data. Are the dominant rhythms in the spectrum consistent with your visual inspection of the data?

c. Apply the two CFC methods developed in this chapter to these data. In doing so, you must choose the low-frequency and high-frequency bands. What choices will you make, and why? What do you find using the two CFC methods?

d. Describe your spectrum and CFC results, as you would to a colleague or collaborator.

7.3. Generate synthetic data consisting of Gaussian noise. More specifically, generate 100 s of artificial noise data sampled at 1000 Hz. Then compute CFC of these data. To do so, use the low-frequency band of 5–7 Hz and the high-frequency band of 80–120 Hz. What do you expect to find (i.e., will this noisy signal exhibit CFC)? What do you find?

7.4. In this chapter, we examined the relation between the phase of the low-frequency band (5–7 Hz) and the amplitude of a selected high-frequency band (80–120 Hz) and found strong evidence in support of CFC. Perhaps CFC occurs between other frequency bands in these data. Repeat both methods of CFC analysis described, but now consider the relation between the low-frequency band (5–7 Hz) and a second high-frequency band of 40–60 Hz. What evidence do you find for CFC between these two frequency bands?

7.5. In our computation of the GLM-CFC statistic for the LFP data analyzed in this chapter, we fixed at 8 the number of control points in the spline model. Repeat the analysis of the LFP data using different choices for the number of control points. How are the results affected?

7.6. In our analysis of CFC, we focused on distinct choices of high- and low-frequency bands. However, sometimes we would like to explore a broader range of potential cross-frequency interactions. To do so, we need a comodulagram. Use the code developed in this chapter to define a new function that computes a comodulogram. Your comodulogram should have two axes:

a. x-axis: the phase frequency (e.g., 3 Hz to 12 Hz in 1 Hz steps)
b. y-axis: the amplitude envelope frequency (e.g., 50 Hz to 200 Hz in 10 Hz steps)

For each pair of (x-axis, y-axis) values, determine the statistic h (as defined in method 1) and plot the three-dimensional results. For reference and motivation, consider the comodulograms in [25].

Appendix: Hilbert Transform in the Time Domain

We have presented the Hilbert Transform in the frequency domain: it produces a quarter cycle phase shift. It's reasonable to consider as well the *time domain* representation of the Hilbert Transform. To do so, let's write the Hilbert Transform as

$$H(x) = \left\{ \begin{array}{rl} -ix & \text{for positive frequencies of } x \\ 0 & \text{for 0 frequency of } x \\ ix & \text{for negative frequencies of } x \end{array} \right\} = x(f)\big(-i\,\mathrm{sgn}(f)\big),$$

where we have written $x(f)$ to make the frequency dependence of x explicit, and the sgn (pronounced "sign") function is defined as

$$\mathrm{sgn}(f) = \begin{cases} 1 & \text{if } f > 0, \\ 0 & \text{if } f = 0, \\ -1 & \text{if } f < 0. \end{cases}$$

In the frequency domain, we perform the Hilbert Transform by multiplying the signal $x(f)$ by a constant (either i or $-i$ depending on the frequency f of x).

To convert the Hilbert Transform in the frequency domain to the Hilbert Transform in the time domain, we take the inverse Fourier transform. Looking up the inverse Fourier transform of $-i\,\mathrm{sgn}(f)$, we find

$$\text{Inverse Fourier transform}\{-i\,\mathrm{sgn}(f)\} = \frac{1}{\pi t}.$$

Let's represent the inverse Fourier transform of $x(f)$ as $x(t)$.

Now, let's make use of an important fact. Multiplication of two quantities in the frequency domain corresponds to convolution of these two quantities in the time domain (see chapter 4). The convolution of two signals x and y is

$$x \star y = \int_{-\infty}^{\infty} x(\tau)y(t-\tau)d\tau.$$

So, in the time domain, the Hilbert Transform becomes

$$H(x) = x(t) \star \frac{1}{\pi t} = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{x(\tau)}{t-\tau} d\tau. \tag{7.6}$$

This time domain representation of the Hilbert Transform is equivalent to the frequency domain representation. However, the time domain representation is much less intuitive. Compare (7.6) to the statement, "The Hilbert Transform is a 90-degree phase shift in the frequency domain." The latter, we propose, is much more intuitive.