A better way to define and describe Morlet wavelets for time-frequency analysis

Michael X Cohen

Radboud University and Radboud University Medical Center; Donders Institute for Neuroscience

Correspondence: mikexcohen@gmail.com
*Funding: MXC is funded by an ERC-StG 638589*

Abstract

Morlet wavelets are frequently used for time-frequency analysis of non-stationary time series data, such as neuroelectrical signals recorded from the brain. The crucial parameter of Morlet wavelets is the width of the Gaussian that tapers the sine wave. This width parameter controls the trade-off between temporal precision and frequency precision. It is typically defined as the "number of cycles," but this parameter is opaque, and often leads to uncertainty and suboptimal analysis choices, as well as being difficult to interpret and evaluate. The purpose of this paper is to present alternative formulations of Morlet wavelets in time and in frequency that allow parameterizing the wavelets directly in terms of the desired temporal and spectral smoothing (as full-width at half-maximum). This formulation provides clarity on an important data analysis parameter, and should facilitate proper analyses, reporting, and interpretation of results. MATLAB code is provided.

Motivation for time-frequency analysis

Many biological and physical systems exhibit rhythmic processes. Rhythmic temporal structure embedded in time series data can be extracted and quantified using the Fourier transform or other spectral analysis methods.

However, the Fourier transform has a "soft assumption" of signal stationarity, which means that the spectral and other features of the signal remain constant over time. To be sure, the Fourier transform is a perfect representation of the time series, regardless of its temporal dynamics. However, the power spectrum resulting from the Fourier transform is easily visually interpretable only for stationary signals; non-stationarities are encoded in the phase spectrum, which is typically impossible to interpret visually.

Signal non-stationarities are the primary motivation for time-frequency analyses, in which the power spectrum is computed over short windows of time. The primary assumption here is that the signal is roughly stationary over some shorter (sliding) time window. For neural time series data, this assumption is typically a few hundreds of milliseconds. An example of a "static" and "dynamic" spectral representation of a non-stationary signal is presented in Figure 1.
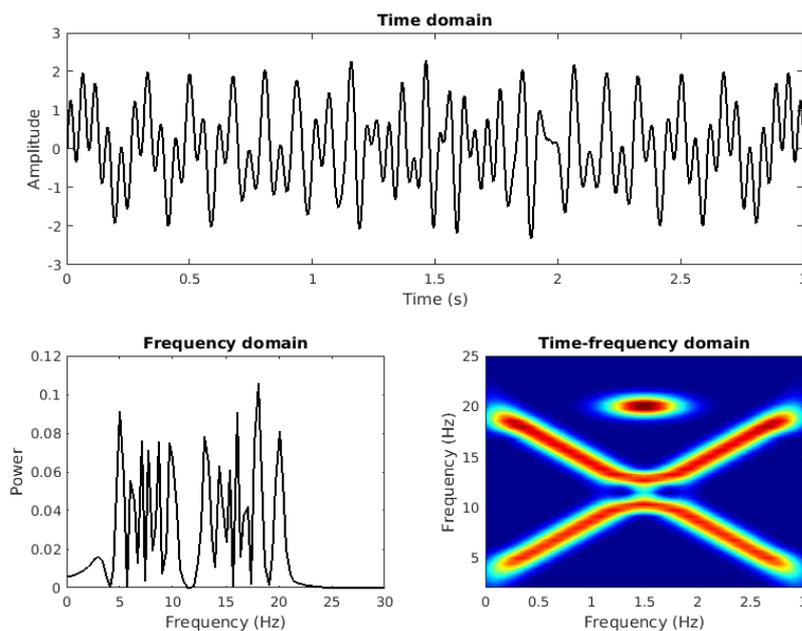
Figure 1. In the presence of signal non-stationarities (panel A), the "static" power spectrum from the Fourier transform can be difficult to interpret (panel B). In these cases, a time-frequency analysis (panel C) is often insightful.

Overview of methods for time-frequency analysis

There are several time-frequency analysis methods, most of which produce qualitatively or quantitatively similar results (Bruns 2004; Cohen 2014). The three most commonly used are the short-time fast Fourier transform, complex wavelet convolution, and filter-Hilbert. These three methods have in common that they involve isolating a narrow temporal window of data and then extracting its frequency spectrum.

A key parameter in time-frequency analysis is the one that governs the trade-off between temporal precision and spectral precision; it is not possible to have simultaneously arbitrarily good precision in both time and in frequency. Furthermore, the resulting smoothness is often advantageous for averaging data across stimulus repetitions and individuals; thus, arbitrarily good precision can be detrimental in applied data analysis of biological systems.
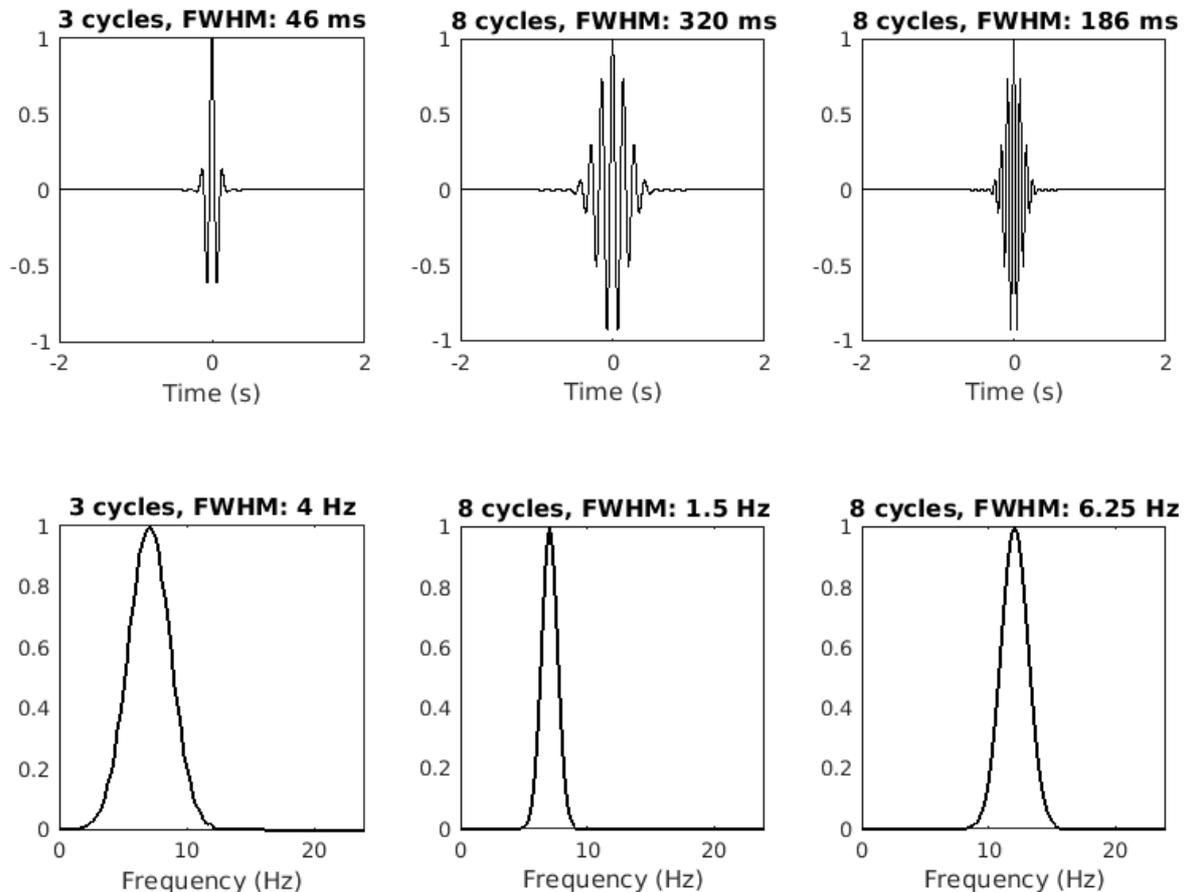
The key argument of this paper is that this crucial parameter is created and reported in a way that obscures both the signal-processing and the theoretical assumptions that are imposed on the data, and that shape the results. *The key upshot of this paper* is two alternative methods for creating and reporting Morlet wavelets in a way that makes this key parameter transparent and easily interpretable.

Advantages and assumptions of Morlet wavelets for time-frequency analysis

A Morlet wavelet is defined as a sine wave tapered by a Gaussian (Figure 2, top row). For time-frequency analysis, a complex Morlet wavelet is used, in which the Gaussian tapers a complex sine wave. The complex Morlet wavelet is then convolved with the time series signal, and the result of convolution is a complex-valued signal from which instantaneous power and phase can be extracted at each time point. Wavelet convolution can be conceptualized as a "template-matching" procedure, in which each time point in the signal is compared against a template (the Gaussian-windowed sine wave), and the result of the convolution is a time series of "similarities" between the signal and the wavelet.

The tapering Gaussian has one parameter that defines its width (also called its shape or deviation). A wider Gaussian leads to decreased temporal precision but increased spectral precision, and vice-versa for a narrower Gaussian. This parameter is typically defined as the "number of cycles," but the purpose of this paper is to argue that it would be better to define the Gaussian width as the full-width at half-maximum (FWHM), which is the distance in time between 50% gain before the peak to 50% gain after the peak.

There are several advantages of Morlet wavelets for time-frequency analysis. One is that the Morlet wavelet is Gaussian-shaped in the frequency domain (Figure 2, bottom row). The absence of sharp edges minimizes ripple effects that can be misinterpreted as oscillations (this is a potential danger associated with plateau-shaped filters). Second, the results of Morlet wavelet convolution retain the temporal resolution of the original signal. Third, wavelet convolution is more computationally efficient and requires less code compared to other methods, because it involves the smallest number of computations, most of which are implemented using the fast Fourier transform.

Figure 2. *Three Morlet wavelets in the time domain (upper row) and in the frequency domain (lower row) with two frequencies and different time-frequency trade-off parameters. The argument of this paper is that reporting this parameter in terms of full-width at half-maximum (FWHM) in the time and/or frequency domains is more informative than number of cycles.*

There are two key assumptions of Morlet wavelet-based time-frequency analysis. The first is that the signal is roughly sinusoidal. Some researchers have noted that brain oscillations are probably not shaped like pure sine waves (Cole and Voytek 2017; Jones 2016). On the other hand, the overwhelming success and ubiquity of time-frequency analysis methods indicates that minor violations of this assumption are not deleterious for practical data analysis, hypothesis-testing, data mining, and feature discovery. The sinusoidal assumption is therefore considered valid, and additional detailed waveform analyses can be insightful in specific instances.

The second assumption of Morlet wavelet analysis concerns stationarity of the signal. Interpreting the results of Morlet wavelet convolution relies on the assumption that the signal is stationary within the time window that the wavelet has non-zero energy. Generally, this can be taken as the FWHM of the Gaussian that is used to create the Morlet wavelet. Any signal non-stationarities that are smaller than this FWHM will produce results that are difficult to interpret (similar to the "static" power spectrum in Figure 1).

Appropriate selection of the width of the wavelet is a combination of signal processing considerations, and theoretical/speculative considerations of the system from which the signal was measured. This parameter is therefore important for time frequency data analysis, and yet is often

selected and reported in a way that obscures the assumption underlying the data analysis. The key goal of this paper is to present two different methods of formulating and describing Morlet wavelets in a way that makes this assumption transparent and easily interpretable.

Typical definition of Morlet wavelets

A complex Morlet wavelet w can be defined as the product of a complex sine wave and a Gaussian window:

$$w = e^{2i\pi ft} e^{\frac{-t^2}{2\sigma^2}} \qquad (1)$$

where i is the imaginary operator ($i = \sqrt{-1}$), f is frequency in Hz, and t is time in seconds. To avoid introducing a phase shift, t should be centered at $t=0$, for example, by defining t from -2 to +2 seconds. σ is the width of the Gaussian, which is defined as

$$\sigma = \frac{n}{2\pi f} \qquad (2)$$

The parameter that defines the time-frequency precision trade-off is n, which is often referred to as the "number of cycles." For neurophysiology data such as EEG, MEG, and LFP, typical values of n range from 2 to 15 over frequencies between 2 Hz and 80 Hz. Other applications may require a different number of cycles.

There is nothing "wrong" with equations 1 and 2. However, the relationship between the number of cycles and the temporal-spectral smoothing is unclear, partly because most people do not think about time in terms of "number of cycles," and partly because the effect of n on the temporal smoothing depends on the frequency of the wavelet (see Figure 2). Many researchers and students struggle with how to specify and interpret this parameter, which leads to confusion and possibly suboptimal data analyses.

A better definition of the Morlet wavelet

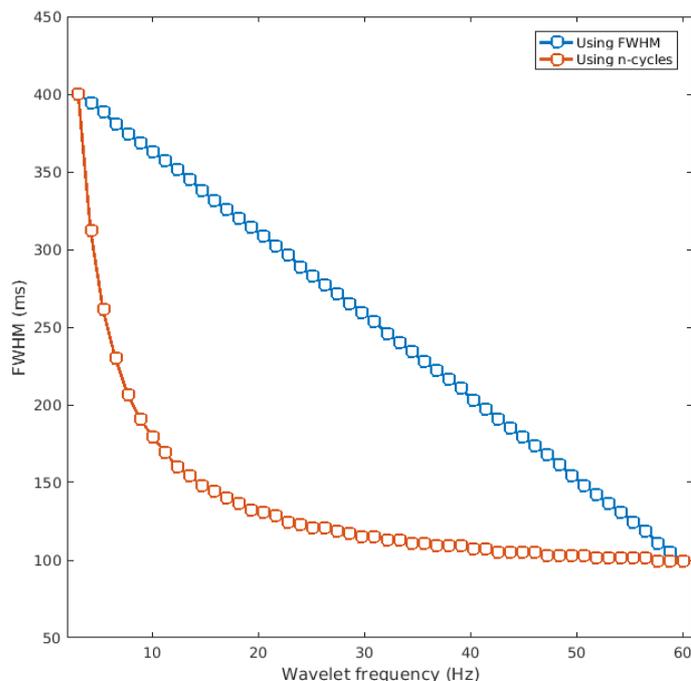A definition of the complex Morlet wavelet that provides more clarity is

$$w = e^{2i\pi ft} e^{\frac{-4ln(2)t^2}{h^2}} \qquad (3)$$

The new parameter here is h, which is the FWHM in seconds.

Equation 3 is not fundamentally different from equations 1-2. It is simply a way of rewriting the formula to make the temporal smoothing parameter easier to define, understand, and report. Rather than specifying a number of cycles (which might feel arbitrary and mysterious), one can specify a FWHM in seconds (assuming that t is defined in seconds), which is a more natural unit to think about in terms of desired temporal smoothing, and in terms of the assumption of the duration of stationarity in the system under investigation.

The h parameter in equation 3 is analytical; its actual implemented value depends on the data sampling rate. For this reason, it is recommended to compute and examine the empirical FWHM that is implemented in software. When the sampling rate is considerably higher than the frequency of interest, the specified and empirical widths should match closely.

The empirical FWHM can be obtained by subtracting the time point where the post-peak Gaussian is closest to 50% gain (if the Gaussian is normalized to a peak amplitude of 1, then this is the value closest to .5) from the time point where the pre-peak Gaussian is closest to 50% gain. (This algorithm is presented in the MATLAB code at the end of this document.) Figure 3 shows the relationship between the empirical FWHM derived from equations 1-2 vs. equation 3.

Figure 3. *The empirical FWHM of the Gaussian (y-axis) used to define complex Morlet wavelets over a range of frequencies (x-axis), defined using equations 1-2 vs. equation 3.*

Recommendation for minimum wavelet FWHM

As mentioned earlier, there are two considerations for selecting the width of a wavelet for time-frequency analysis: signal processing and system quasi-stationary. The minimum width of a wavelet should be derived only from signal processing considerations.

It is recommended to set the FWHM no lower than one cycle at the frequency of the sine wave used to create the wavelet. For example, a 10 Hz wavelet should not have less than a 100 millisecond FWHM. This leads to the simple formula to compute the minimum FWHM of *1/f*, where *f* is the frequency of the sine wave that creates the wavelet. This corresponds to ~2⅔ cycles (the *n* parameter in equation 2).

Note that a FWHM corresponding to one cycle actually means that more than one cycle will contribute to the wavelet, because there is still non-zero energy beyond the 50% gain boundaries used to define FWHM.

Also note that this is a recommendation for the *minimum* wavelet length. It is often sensible to have wavelets longer than the minimum, depending on a desired amount of temporal or spectral smoothing. One should also keep in mind that wavelets with narrower Gaussians (smaller FWHM)

have wider spectral energy windows (Cohen 2014), so creating wavelets using the minimum bound is not necessarily optimal.

Alternative: defining the wavelet in the frequency domain

The discussion above concerns defining the wavelet in the time domain. This is a natural approach in many situations, but in other situations it might be more natural to start from a desired amount of spectral smoothing. Thus, one can specify the desired spectral smoothing, which means defining the wavelet in the frequency domain instead of in the time domain. The formula is slightly different.

$$g = e^{-0.5(x/s)^2} \tag{4}$$
$$x = z - f \tag{5}$$
$$s = \frac{h(2\pi - 1)}{4\pi} \tag{6}$$

where z is a vector of frequencies in Hz from 0 to the sampling rate in the number of steps as the desired length of the spectrum (e.g., the length of the time series signal, or the length of the result of convolution), f is the peak frequency of the wavelet, and h is the FWHM in Hz.

The Gaussian in equation 4 creates the amplitude shape in the frequency domain; the time-domain wavelet can be obtained by the inverse Fourier transform of this envelope. The empirical FWHM in the time domain Morlet wavelet can be obtained by applying the FWHM estimation procedure to the magnitude of the complex time series. An example is shown in Figure 4.
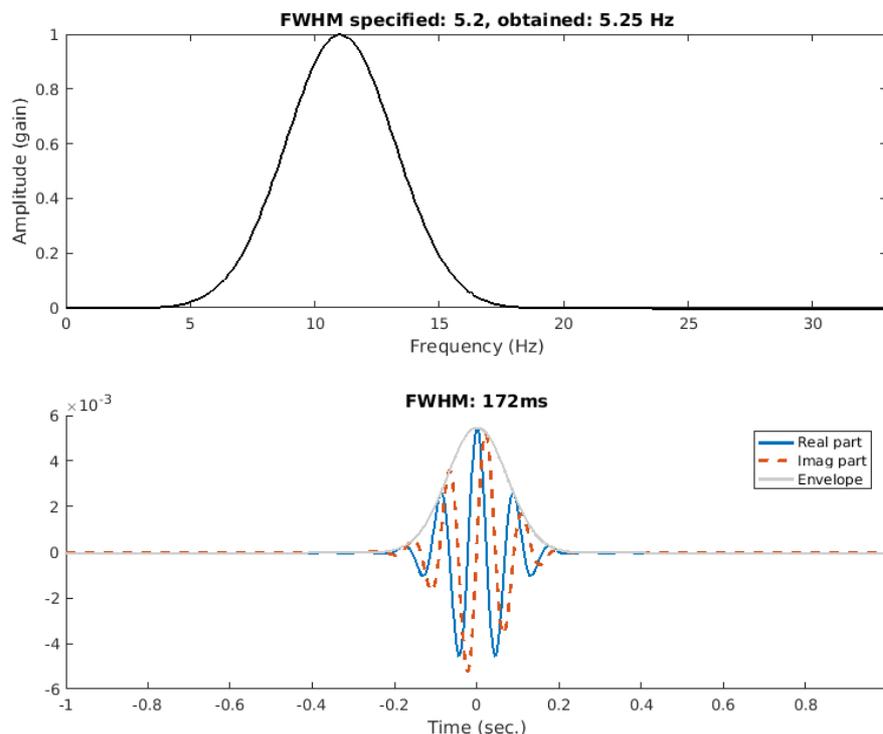
Figure 4. *An example of specifying the wavelet shape in the frequency domain (top panel) and computing its inverse Fourier transform to obtain a time-domain Morlet wavelet (bottom panel). The temporal FWHM is defined using the envelope.*

Reporting

It is sensible and useful to report the FWHM in both the time and the frequency domains. This will facilitate interpretation of the results and replication of analysis methods. Here is a suggestion for describing the wavelets:

> We implemented time-frequency analysis by convolving the signal with a set of complex Morlet wavelets, defined as complex sine waves tapered by a Gaussian. The frequencies of the wavelets ranged from X Hz to X Hz in X steps. The full-width at half-maximum (FWHM) ranged from XXX to XXX ms with increasing wavelet peak frequency. This resulted in a spectral FWHM range of XX Hz to XX Hz.

Conclusion

The goal of this paper is to provide an alternative way to conceptualize, define, and report Morlet wavelets for time-frequency analysis. No new methods, formulas, or algorithms are presented here; instead, the idea is to encourage researchers and students to make a minor adjustment to their existing analysis pipeline in a way that makes the findings, analysis methods, assumptions, and results, easier to understand, communicate, and interpret. MATLAB code to reproduce the figures is presented as a supplement.

References

Bruns, Andreas. 2004. "Fourier-, Hilbert- and Wavelet-Based Signal Analysis: Are They Really Different Approaches?" *Journal of Neuroscience Methods* 137 (2): 321–32.

Cohen, M. X. 2014. *Analyzing Neural Time Series Data: Theory and Practice*. MIT Press.

Cole, Scott R., and Bradley Voytek. 2017. "Brain Oscillations and the Importance of Waveform Shape." *Trends in Cognitive Sciences* 21 (2): 137–49.

Jones, Stephanie R. 2016. "When Brain Rhythms Aren't 'rhythmic': Implication for Their Mechanisms and Meaning." *Current Opinion in Neurobiology* 40: 72–80.

MATLAB code to reproduce figure 1

```
%%% generate signal
srate = 1000;
time  = (0:3*srate)/srate;
pnts  = length(time);

f1 = [3 20];
f2 = [20 3];

transgaus = exp( -(time-mean(time)).^2 / .2 );

signal = sin(2*pi*time.*linspace(f1(1),mean(f1),pnts)) + ...
         sin(2*pi*time.*linspace(f2(1),mean(f2),pnts)) + ...
         sin(2*pi*time*20).*transgaus;


%%% static power spectrum
hz = linspace(0,srate/2,floor(pnts/2)+1);
powr = (2*abs(fft(signal)/pnts)).^2;


%%% time-frequency analysis
nfrex = 40;
frex = linspace(2,25,nfrex);
fwhm = linspace(.8,.7,nfrex);

% setup wavelet and convolution parameters
wavet = -2:1/srate:2;
halfw = floor(length(wavet)/2)+1;
nConv = pnts + length(wavet) - 1;

% initialize time-frequency matrix
tf = zeros(length(frex),pnts);

% spectrum of data
dataX = fft(signal,nConv);

% loop over frequencies
for fi=1:length(frex)

    % create wavelet
    waveX = fft( exp(2*1i*pi*frex(fi)*wavet).*exp(-(4*log(2)*wavet).^2/fwhm(fi).^2),nConv );
    waveX = waveX./max(waveX); % normalize

    % convolve
    as = ifft( waveX.*dataX );
    % trim and reshape
    as = as(halfw:end-halfw+1);

    % power
    tf(fi,:) = abs(as).^2;
end


%%% plotting

figure(1), clf
subplot(211)
plot(time,signal,'k','linew',2)
xlabel('Time (s)'), ylabel('Amplitude')
```

```
    set(gca,'xlim',time([1 end]))
    title('Time domain')


    subplot(223)
    plot(hz,powr(1:length(hz)),'k','linew',2)
    set(gca,'xlim',[0 30])
    xlabel('Frequency (Hz)'), ylabel('Power')
    title('Frequency domain')


    subplot(224)
    contourf(time,frex,tf,40,'linecolor','none')
    xlabel('Frequency (Hz)'), ylabel('Frequency (Hz)')
    title('Time-frequency domain')
```

MATLAB code to reproduce figure 2

```matlab
% simulation parameters
freq1 = 7; % of the wavelet
freq2 = 12;
srate = 1000;
time  = -2:1/srate:2;
pnts  = length(time);

% define number of cycles
numcycles = [ 3 8 ];

% create the sine wave and two Gaussian windows
sinwave1 = cos(2*pi*freq1*time);
sinwave2 = cos(2*pi*freq2*time);
gauswin1 = exp( -time.^2 / (2* (numcycles(1)/(2*pi*freq1))^2 ) );
gauswin2 = exp( -time.^2 / (2* (numcycles(2)/(2*pi*freq1))^2 ) );
gauswin3 = exp( -time.^2 / (2* (numcycles(2)/(2*pi*freq2))^2 ) );


% create the two wavelets
morletwave1 = sinwave1 .* gauswin1;
morletwave2 = sinwave1 .* gauswin2;
morletwave3 = sinwave2 .* gauswin3;


% compute the empirical temporal FWHM in seconds (later converted to ms)
midp = dsearchn(time',0);
fwhmT(1) = time(midp-1+dsearchn(morletwave1(midp:end)',.5)) -
time(dsearchn(morletwave1(1:midp)',.5));
fwhmT(2) = time(midp-1+dsearchn(morletwave2(midp:end)',.5)) -
time(dsearchn(morletwave2(1:midp)',.5));
fwhmT(3) = time(midp-1+dsearchn(morletwave3(midp:end)',.5)) -
time(dsearchn(morletwave3(1:midp)',.5));


% normalized power spectra of the wavelets
powspect1 = abs(fft(morletwave1)/pnts).^2;
powspect1 = powspect1 ./ max(powspect1);

powspect2 = abs(fft(morletwave2)/pnts).^2;
powspect2 = powspect2 ./ max(powspect2);

powspect3 = abs(fft(morletwave3)/pnts).^2;
powspect3 = powspect3 ./ max(powspect3);


hz = linspace(0,srate/2,floor(pnts/2)+1);

idx1 = dsearchn(hz',freq1);
idx2 = dsearchn(hz',freq2);
fwhmF(1) = hz(idx1-1+dsearchn(powspect1(idx1:end)',.5)) - hz(dsearchn(powspect1(1:idx1)',.5));
fwhmF(2) = hz(idx1-1+dsearchn(powspect2(idx1:end)',.5)) - hz(dsearchn(powspect2(1:idx1)',.5));
fwhmF(3) = hz(idx2-1+dsearchn(powspect3(idx2:end)',.5)) - hz(dsearchn(powspect3(1:idx2)',.5));


%%% plotting
figure(2), clf
subplot(231)
plot(time,morletwave1,'k')
```

```
    xlabel('Time (s)'), axis square
    title([ num2str(numcycles(1)) ' cycles, FWHM: ' num2str(fwhmT(1)*1000) ' ms' ])

    subplot(232)
    plot(time,morletwave2,'k')
    xlabel('Time (s)'), axis square
    title([ num2str(numcycles(2)) ' cycles, FWHM: ' num2str(fwhmT(2)*1000) ' ms' ])

    subplot(233)
    plot(time,morletwave3,'k')
    xlabel('Time (s)'), axis square
    title([ num2str(numcycles(2)) ' cycles, FWHM: ' num2str(fwhmT(3)*1000) ' ms' ])




    subplot(234)
    plot(hz,powspect1(1:length(hz)),'k','linew',2)
    set(gca,'xlim',[0 freq2*2])
    xlabel('Frequency (Hz)'), axis square
    title([ num2str(numcycles(1)) ' cycles, FWHM: ' num2str(fwhmF(1)) ' Hz' ])

    subplot(235)
    plot(hz,powspect2(1:length(hz)),'k','linew',2)
    set(gca,'xlim',[0 freq2*2])
    xlabel('Frequency (Hz)'), axis square
    title([ num2str(numcycles(2)) ' cycles, FWHM: ' num2str(fwhmF(2)) ' Hz' ])

    subplot(236)
    plot(hz,powspect3(1:length(hz)),'k','linew',2)
    set(gca,'xlim',[0 freq2*2])
    xlabel('Frequency (Hz)'), axis square
    title([ num2str(numcycles(2)) ' cycles, FWHM: ' num2str(fwhmF(3)) ' Hz' ])
```

MATLAB code to reproduce figure 3

```matlab
% specify frequencies
frex = linspace(3,60,50);
fwhm = linspace(.4,.1,length(frex)); % in ms
ncyc = linspace(3.2,16,length(frex));


% parameters for complex Morlet wavelets
srate = 1024;
wavtime  = -2:1/srate:2;
midp = dsearchn(wavtime',0);


% outputs
empfwhm = zeros(length(frex),2);

% loop over frequencies
for fi=1:length(frex)

    % create the Gaussian using the FWHM formula (equation 3)
    gwin = exp( (-4*log(2)*wavtime.^2) ./ fwhm(fi)^2 );

    % measure the empirical fwhm
    empfwhm(fi,1) = wavtime(midp-1+dsearchn(gwin(midp:end)',.5)) -
wavtime(dsearchn(gwin(1:midp)',.5));



    % create the Gaussian using the n-cycles formula (equations 1-2)
    s    = ncyc(fi) / (2*pi*frex(fi));
    gwin = exp( -wavtime.^2 ./ (2*s^2) );

    % empirical FWHM
    empfwhm(fi,2) = wavtime(midp-1+dsearchn(gwin(midp:end)',.5)) -
wavtime(dsearchn(gwin(1:midp)',.5));
end

figure(3), clf

plot(frex,empfwhm*1000,'o-','markersize',8,'markerfacecolor','w','linew',2)
xlabel('Wavelet frequency (Hz)'), ylabel('FWHM (ms)')
legend({'using FWHM';'Using n-cycles'})
set(gca,'xlim',[frex(1)-1 frex(end)+1])
```

MATLAB code to reproduce figure 4

```matlab
% specify wavelet parameters
peakf = 11;
fwhm  = 5.2;

% specify simulation details
npnts = 8001;
srate = 1000;


% vector of frequencies
hz = linspace(0,srate,npnts);

% frequency-domain Gaussian
s  = fwhm*(2*pi-1)/(4*pi); % normalized width
x  = hz-peakf;             % shifted frequencies
fx = exp(-.5*(x/s).^2);    % gaussian

% empirical FWHM
idx = dsearchn(hz',peakf);
empfwhmF = hz(idx-1+dsearchn(fx(idx:end)',.5)) - hz(dsearchn(fx(1:idx)',.5));


% time-domain wavelet
morletwavelet = fftshift( ifft(fx) );
time = (-floor(npnts/2):floor(npnts/2))/srate;

% FWHM of wavelet in the time domain
midp = dsearchn(time',0);
mw_amp = abs(morletwavelet);
mw_amp = mw_amp./max(mw_amp);
empfwhmT = time(midp-1+dsearchn(mw_amp(midp:end)',.5)) - time(dsearchn(mw_amp(1:midp)',.5));


%%% plotting
figure(5), clf
subplot(211)
plot(hz,fx,'k','linew',2)
set(gca,'xlim',[0 peakf*3])
xlabel('Frequency (Hz)'), ylabel('Amplitude (gain)')
title([ 'FWHM specified: ' num2str(fwhm) ', obtained: ' num2str(empfwhmF) ' Hz' ])

subplot(212), hold on
plot(time,real(morletwavelet),'linew',2)
plot(time,imag(morletwavelet),'--','linew',2)
h = plot(time,abs(morletwavelet),'linew',2);
set(h,'color',[1 1 1]*.8)
set(gca,'xlim',[-1 1])
legend({'Real part';'Imag part';'Envelope'})
xlabel('Time (sec.)')
title([ 'FWHM: ' num2str(empfwhmT*1000) 'ms' ])
```