NeuroSpec 2.0 User Guide

David M. Halliday

Contents

1 Introduction

This guide provides an overview of the NeuroSpec statistical signal processing archive version 2.0. The archive consists of a number of MATLAB functions for performing time and frequency domain analyses of time series and/or point process data, and plotting the results. NeuroSpec 2.0 implements a multivariate spectral analysis, it includes routines for two channel auto spectral and cross spectral (coherence, phase, cumulant density) analysis, and a number of extensions including comparison of spectra, a stationarity test for spectra, comparison of coherence, system identification (gain, phase and impulse response), and pooled analysis (pooled spectra, pooled coherence, pooled phase, pooled cumulant density). Pooled analysis incorporates extended difference of spectra and extended difference of coherence tests. Four categories of routine are included in the software:

1. Core routines for processing two channels of time series and/or spike train data.
2. Extensions for comparison of spectra or comparison of coherence and pooled analysis.
3. Plotting routines.
4. Demonstration scripts.

2 Software requirements

These routines require the MATLAB system to be installed. Version 5.x or higher is required to run NeuroSpec 2.0. They have been tested using PC versions 5.3, 6.1 and 7.0.1. No additional toolboxes are required, only the base system is needed. If you have an older version of MATLAB please read the FAQ page on the NeuroSpec web site.

3 Installation, quick start guide & getting help

To install the archive:

1. Download and Unzip the file `neurospec20.zip`.
2. Install the directory `neurospec20` in your MATLAB path. See the MATLAB Help system for details of how to do this (In MATLAB 7.x the Set Path option is accessed from the File menu).
3. To get started straight away switch MATLAB to directory `neurospec20_demos` and run the demonstration scripts, starting with `sp2_type0_demo1`.

Once installed, additional help is available via the MATLAB help system. The command `helpwin` should display an entry: `neurospec20`, which in turn should display a hyperlinked list of files, similar to that in Appendix A. Help for individual commands is available using the `help` or `helpwin` commands followed by the function name. The MATLAB source for the demonstration scripts in directory `neurospec20_demos` should provide additional guidance in the use of the software.

4 Format of raw data in MATLAB

This section describes the format that raw data should be input to the NeuroSpec routines. NeuroSpec can process either time series or point process (spike train) data. The formats specified below should readily accommodate most types of experimental and/or simulated data. We adopt the MATLAB convention of specifying multi-channel data in column format. *Note*: Failure to specify input data in this format may give incorrect results.

4.1 Time series data

Time series data is assumed to represent regularly sampled data with a known sampling rate. Integer or real values can be processed. Time series data should be in column format. Each column should represent one signal, with one sample per row. The sampling rate is input to the analysis - this, in conjunction with the FFT segment length, defines the spacing of the Fourier frequencies returned after analysis. Both single column and 2D matrices can be processed. Each column in a 2D matrix should contain data from one signal. Each row in a 2D matrix should contain data samples at the same time instant from several channels. See the script `sp2_type0_demo1` for an example of processing matrix data. Two single columns of data passed to a NeuroSpec core routine should have the same number of data points in each column and be from simultaneously sampled signals.

4.2 Point process (spike train) data

Point Process (Spike Train) data should be in column format. Each row should specify the time of a spike (or event) as an integer multiple of the sampling rate. Times in floating point format should first be converted to integer values using an appropriate sampling rate. If analysing hybrid data, spike trains should be specified with the same sampling rate as the time series data. Spikes times should be specified in ascending order of occurrence. Since it is unlikely that two or more spike trains will have identical numbers of spikes (or events), these will generally be stored in separate variables. To reduce the number of workspace variables, spike trains can also be stored as elements of a MATLAB structure.

5 The NeuroSpec 2.0 core routines

All spectral analysis is done using the function `sp2_fn2a` which implements a weighted periodogram based spectral estimation for two channels of data. This function estimates the two auto spectra, coherence and phase in the frequency domain, and the cumulant density in the time domain. It works on both time series and/or point process data. The routine can optionally calculate the gain and impulse response of a linear transfer function model, and implement a Periodogram COV test for stationarity. The function `sp2_fn2a` is not designed to be called directly from within MATLAB. Three core routines provide the interface between your raw data and `sp2_fn2a`. They are:

1. `sp2a2_m1` For processing two time series.
2. `sp2a_m1` For processing hybrid data (one time series and one spike train).
3. `sp2_m1` For processing two spike trains.

Each of these routines supports three distinct types of data analysis, referred to as type 0, type 1, and type 2. Type 0 analysis is identical to that implemented in NeuroSpec 1.0, type 1 and type 2 analysis are new to NeuroSpec 2.0. The spectral estimation procedure is based on weighted periodogram estimates, for a particular analysis each periodogram is calculated from discrete Fourier transforms (DFT) containing S points, where S is a power of 2. Each DFT segment consists of T data points, where $T \leq S$. If $T < S$, then zeros are appended to the T data points to create a segment of length S.

The routines assume that the first channel of data is the reference (input) signal, and the second channel of data is the response (output) signal. This is important for estimates of phase, cumulant density and the optional impulse response. One of the options allows the reference to be inverted to the second channel (see below).

5.1 Type 0 analysis

Type 0 analysis is illustrated schematically in Figure 1a, it should be used for a single unbroken stretch of data. The data are split into L non overlapping segments, each containing T data points, analysed with the same DFT segment length, $S = T$. The total number of samples analysed is $R = LT$. Only complete segments are analysed, data points at the end of the record that do not make a complete segment are *not* included in the analysis. The segment length S is specified as an input argument to the three core routines, as a power of 2.

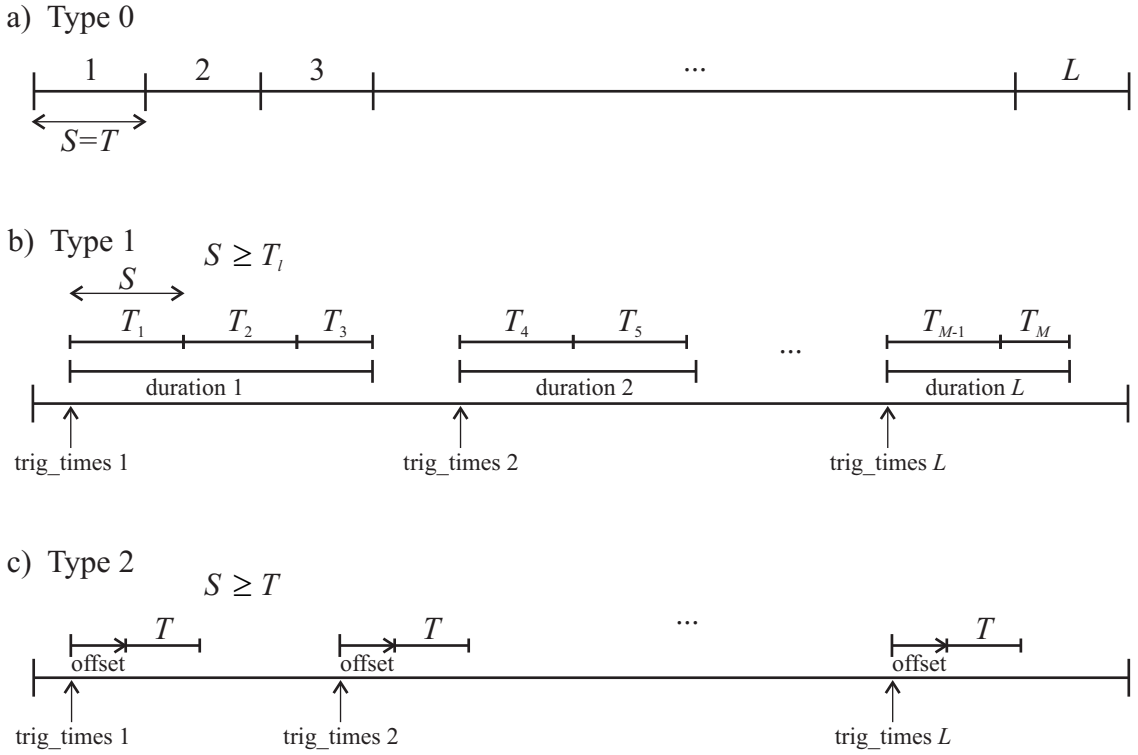

Figure 1: Illustration of Type 0, Type 1 and Type 2 analysis in NeuroSpec . See text for further details

5.2 Type 1 analysis

Type 1 analysis is illustrated schematically in Figure 1b, this allows specific sections from a single record to be analysed. Each section is specified by two parameters: a trigger time (input parameter `trig_times` in routines) and a duration (input parameter `duration` in routines). The L triggers illustrated in Fig. 1b generate M segments of data for analysis, each segment contains T_l points ($l = 1 \ldots M$). The DFT segment length, S, is a fixed power of 2 (as in Type 0 analysis). Each section of data is subdivided into the maximum possible number of segments that contain S data points, thus $T_l \leq S; (l = 1 \ldots M)$. In the example illustrated in Fig. 1b, the first section is split into three segments for analysis: $T_1 = T_2 = S, T_3 \leq S$. The second section contains two segments: $T_4 = T_5 = S$. The data left at the end of section 2 is not included as it contains too few points. The minimum number of points allowed in any segment is 5% of the segment length S. The last section illustrated in Fig.1 b, section L, is split into two segments: $T_{M-1} = S$, $T_M \leq S$. Segments where $T_l \leq S$ have zeros appended to the data points to create a segment of length S points that is input to the DFT. The aim of Type 1 analysis is to extract the maximum possible number of segments containing S data points ($T_l = S$) from the record under analysis. The parameters `trig_times` and `duration` should be input to the routines as column vectors of equal length, both specified as an integer number of samples.

5.3 Type 2 analysis

Type 2 analysis is illustrated schematically in Figure 1c. Type 2 analysis should be used for time-frequency analysis, based on a protocol of averaging over repeat trials. As in Type 1, data for analysis is taken only from specific parts of the record. The start of each trial is defined by the values in the column vector `trig_times`. A fixed `offset` value is added to each trigger value, this defines the start sample of each segment used for analysis. Each segment contains the same number of data points, T, specified by scalar parameter `seg_pts` in the routines. The DFT length of each segment is S, $S \geq T$, this should normally be specified as the next highest power of 2 (e.g. if `seg_pts` = 250, then `seg_pwr` should be set to 8). The input parameter `offset` can be either a single scalar value, or a column vector specifying a

range of offset values. If is specified as a vector, a separate analysis is done for each value in `offset`. A vector of offset values can be used to explore time dependency by moving a window (containing T points) across each trial. The three parameters `trig_times`, `offset` and `seg_pts` are specified as an integer number of samples.

6 Overview of Type 0 analysis

This section gives an overview of the syntax and arguments, including the use of options, for type 0 analysis. Suppose you have in MATLAB the following variables:

`dat1, dat2` Two time series of equal length.

`sp1, sp2` Two spike trains.

Suppose the sampling rate for all signals is 1000/sec, in MATLAB set: `rate=1000`. Define the segment length for the FFT as 2^{10} (1024 points), in MATLAB: `seg_pwr=10`. The segment length in NeuroSpec is always specified as a power of 2.

6.1 Time series analysis

To undertake a type 0 analysis of the two time series use the command:

```
[f,t,cl]=sp2a2_m1(0,dat1,dat2,rate,seg_pwr);
```

The first input argument specifies the type of analysis, for NeuroSpec 2.0 this should always be 0, 1 or 2. Options are included by specifying an option list in a text string. For example, if you wish to perform a linear de-trend on the data before analysis use:

```
[f,t,cl]=sp2a2_m1(0,dat1,dat2,rate,seg_pwr,'t2');
```

If you wish to rectify both channels and then perform a linear de-trend prior to analysis use:

```
[f,t,cl]=sp2a2_m1(0,dat1,dat2,rate,seg_pwr,'t2 r2');
```

For further details on the use of options, see the source file `sp2a2_m1.m`. The order in which the options are specified is not important, however, only a single option string should be used, multiple options should be separated by spaces. To undertake further analysis of the data, using the comparison of spectra, comparison of coherence or pooled spectra extensions an optional fourth output parameter should be included:

```
[f,t,cl,sc]=sp2a2_m1(0,dat1,dat2,rate,seg_pwr,'t2 r2');
```

6.2 Point process (spike train) analysis

To undertake a type 0 analysis of the point process data use the command:

```
[f,t,cl]=sp2_m1(0,sp1,sp2,sec_tot,rate,seg_pwr);
```

Analysis options are specified as above by including an option list in a text string. The optional output, `sc`, can be included as above. The 4^{th} input parameter `sec_tot` specifies the duration of the record in seconds. *Note:* This parameter is only required for the `sp2_m1` core routine. This is a change from the convention used in the equivalent routine in NeuroSpec 1.0.

6.3 Hybrid (time series/point process) analysis

To undertake a type 0 analysis of the hybrid data sp1 and dat2 use the command:

```
[f,t,cl]=sp2a_m1(0,sp1,dat2,rate,seg_pwr);
```

The spike train data is the second parameter, the time series is the third parameter. The routine assumes the spike train is the reference (input) signal for phase, cumulant density and optional linear transfer function estimates, to invert this the `'i'` option should be used. Rectification and de-trending are only available for time series data, to specify these use options with no arguments: `'t r'`.

7 Output from the NeuroSpec core routines

The output from the three `sp2*` functions has the same format in all cases, all return four variables:

1. `f` A matrix of frequency domain results.
2. `t` A matrix of time domain results.
3. `cl` A structure containing confidence limits and other analysis parameter.
4. `sc` An optional matrix of spectral coefficients.

For further details and the format of these variables see the `sp2*.m` source files. The fourth parameter is optional, it is required for the comparison of spectra, comparison of coherence or pooled spectra extensions described below.

8 Plotting of output from the NeuroSpec core routines

Four plotting functions are provided to plot the results from a single analysis with the `sp2*` core routines:

1. `psp2` Plotting of output from Type 0, Type 1 or a single offset in Type 2 analysis.
2. `psp2s` As psp2, includes optional system identification parameters (gain, impulse response).
3. `psp2_cov` As psp2, includes optional Periodogram COV test for weak stationarity.
4. `psp2_tf` Plotting of Type 2 time-frequency analysis using a range of offset values.

For example, assuming that `[f,t,cl]` are returned from a type 0, type 1 or single offset type 2 analysis, then the results can be plotted using a command like:

```
psp2(f,t,cl,100,200,100,0.5,'Good data 1')
```

For a description of the arguments see the file `psp2.m`. The frequency range (4^{th} parameter) is specified in Hz, the lag ranges (5^{th}, 6^{th} parameters) are specified in ms. The frequency range should not exceed the nyquist frequency. The number of time domain lags should not exceed the DFT segment length. Passing values to `psp2` which exceed these limits will generate an error message. The function `psp2` generates 5 plots: two autospectra, coherence, phase and cumulant density. The syntax for calling `psp2s` is identical to `psp2`, except that an additional two plots are generated illustrating the estimated gain ($\log_{10}$ scale) and impulse response. The syntax for calling `psp2_cov` is identical to `psp2`, except that an additional two plots are generated illustrating the Periodogram COV test for each channel.

Plotting of time frequency analysis from Type 2 analysis over a range of offset values is done using the function `psp2_tf`. Individual analyses at a single offset value can be plotted using `psp2`. See the demonstration script for Type 2 analysis (described below) for further details.

9 Demonstration scripts for Type 0, Type 1 and Type 2 analysis

Demonstration scripts are included as part of NeuroSpec for Type 0, Type 1, and Type 2 analysis. These scripts are called `sp2_type0_demo1`, `sp2_type1_demo1` and `sp2_type2_demo1`, they are located in the directory `neurospec20_demos`.

9.1 Type 0 analysis

The script `sp2_type0_demo1` demonstrates the use of type 0 analysis (see Fig. 1a) on artificially generated time series and spike train data. When run it generates 8 figures, which are reproduced below along with a brief description. Since a different set of random data is generated each time the script is run, the precise form of the plots you generate may differ from those shown here. However, the general form should be similar. They are provided as a guide to getting started with NeuroSpec 2.0. Also included are some comments regarding interpretation.

Figure 2 shows an analysis of two uncorrelated signals, generated from a normal distribution. These signals can be considered as an approximation to white noise, and thus have a flat power spectrum over the entire frequency range. The sampling rate for the data is assumed to be 1 ms, thus the analysis has a nyquist frequency of 500 Hz. The spectral estimates have local fluctuations which are comparable in magnitude to the 95% confidence limit. Most of the coherence values lie below the 95% confidence limit, thus no significant correlation is evident. In this case the phase has no valid interpretation, and is randomly distributed over the range $[-\pi, +\pi]$. The time domain cumulant density estimate fluctuates about the null value, zero, with most values inside the upper and lower 95% confidence limits, again supporting the interpretation of uncorrelated data.

Figure 3 illustrates a type 0 analysis of correlated data, with the data generated by combining white noise signals sample by sample. In this case the coherence is constant around 0.5, indicating that channel 1 can account for around 50% of the variability in channel 2 across the entire frequency range. The phase is constant around 0 radians, indicating synchronous signals, and the cumulant has a large peak at time zero indicating that the time span of the correlation is very narrow, as we would expect for point wise combination of white noise signals (a property of white noise is that values in adjacent time bins are independent).

Figure 4 shows the results of introducing a delay of 10 samples (which equates to 10 ms assuming 1 ms sampling) into the correlated data. The phase and cumulant density estimates change to reflect this delay, the phase has constant slope (equal to the delay) and the peak in the cumulant has moved to +10 ms lag. The phase estimate is constrained over the range $\pm\pi$, a discontinuity appears as the phase cycles repeatedly through this range.

Figure 5 illustrates the results of applying an 11 point moving average filter to one channel of white noise data to create the second channel. This analysis also includes the optional systems identification parameters ($\log_{10}$ gain and impulse response). The plots reflect the characteristics of the filtering process. The input signal has a flat power spectrum, the output spectrum is a monotonically decreasing function of frequency. The coherence tails of with frequency as the filter bandwidth is reached, and the input signal no longer influences the output signal. The $\log_{10}$ gain estimate, which, like the phase estimate, is valid over the frequency range where there is significant coherence, shows the low pass filter characteristics of the moving average filter. The phase is dominated by the delay in the moving average filter. The cumulant density gives an indication of the span of dependency of the filter in the time domain, in this case from 0 to 10 ms. The impulse response is very similar to the cumulant density estimate, this similarity reflects the fact that the 'system' has a white noise input.

Figure 6 shows the gain, phase and coherence plots from figure 5 (heavy lines) with estimated point-wise confidence limits (thin lines) included on each plot. These point-wise confidence limits give an indication of the variability associated with the parameter estimates. For example, the coherence has an estimates value of 0.45 at the lowest frequency of 0.98 Hz, the point-wise confidence intervals for this are [0.35 0.55]. The point-wise confidence limits, in this case, are relative constant around ± 0.1 at frequencies up to 50 Hz. This give some useful insight into the interpretation of coherence estimates.

The core routines in NeuroSpec 2.0 have the option to calculate a *Periodogram COV* test as a means of assessing weak stationarity within individual signals. Figure 7 illustrates the same data as in figure 5 processed with this option and plotted using routine `psp2_cov`, the Periodogram COV test for each signal is plotted in row 2, directly below the corresponding spectrum estimates. For both signals, the Periodogram COV values are inside the 95% confidence limits, suggesting that the assumption of weak stationarity is reasonable in this case (as would be expected using this type of simulated data).

Figure 8 illustrates results from analysis of two spike trains, generated independently to have the same mean rate (10 spikes/sec). The first spike train has a coefficient of variation (COV) of 0.1, the second a COV of 0.2. Periodic spike trains have point process spectral estimates that contain harmonic components. The spectral estimates in figure 8 have components that are harmonics of the fundamental periodicity of 10 Hz. The first spike train (upper left) has more harmonic components than the second, as a consequence of it's lower COV. Since they have been generated independently, these spike trains are uncorrelated, this is reflected in the coherence, phase and cumulant density plots.

Figure 9 illustrates the results from analysis of hybrid data: one spike train and one time series. The two sequences have been generated independently, and thus exhibit no correlation. The spike train is assumed to be the input (or reference signal), its spectral estimate is shown at top left, the spectrum of the time series is shown at top right.

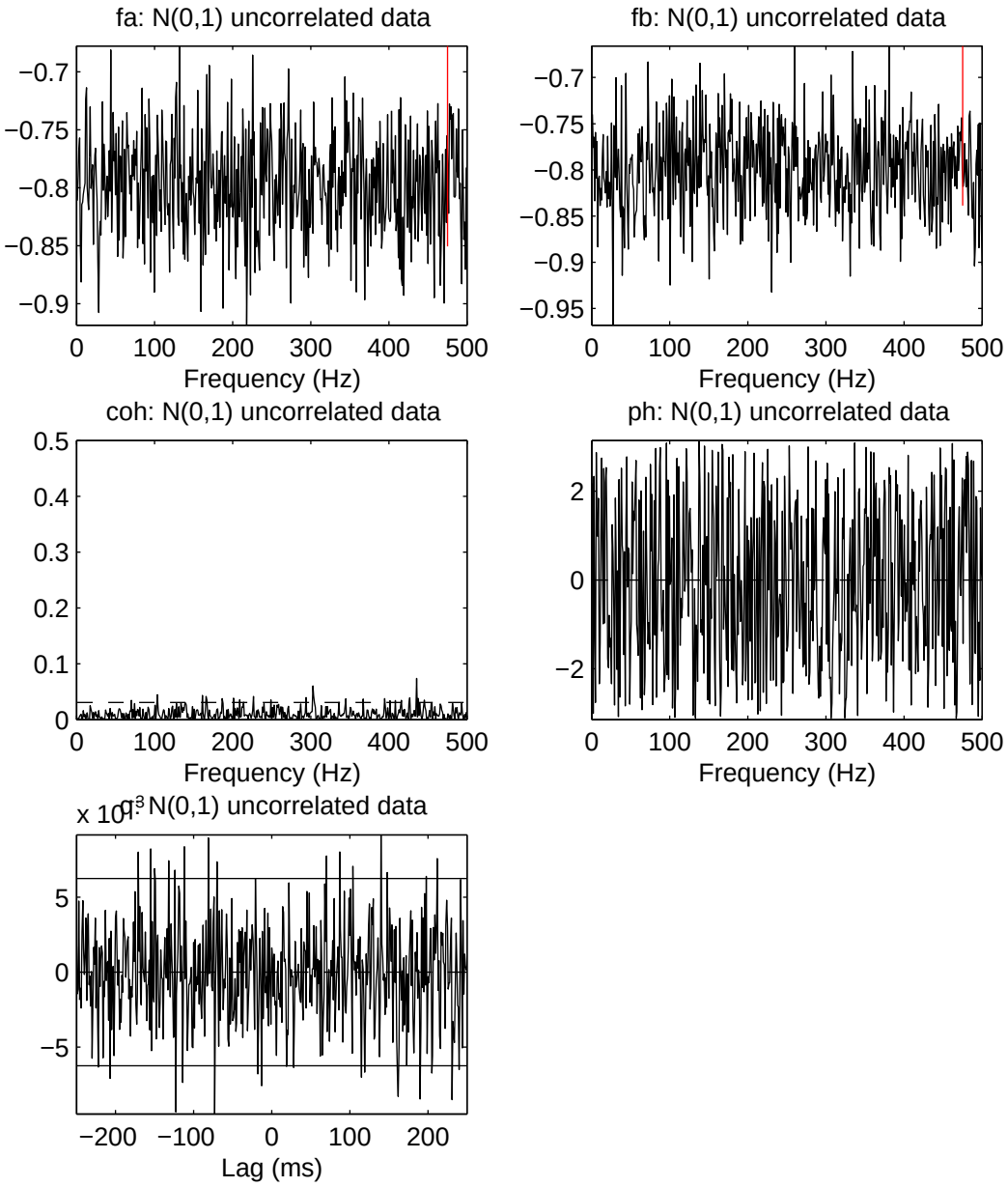

Figure 2: *Type 0 analysis.* Plot generated by `psp2` of output from `sp2a2_m1` for Type 0 analysis of uncorrelated white noise. Top two graphs are plots of the power spectrum of each signal (plotted as $\log_{10}$), for channel 1 (left) and channel 2 (right). The magnitude of the 95% confidence limit is indicated by the the vertical bar in the top right of each plot (highlighted in red in this figure). The two centre plots show the coherence (left) and phase (right) estimates. The horizontal dashed line in the coherence estimate is the upper 95% confidence limit based on the assumption of independence. The abscissa in the top four plots is labelled in Hz (up to the presumed nyquist frequency of 500 Hz). The lower plot shows the cumulant density estimate, plotted against time lag in ms. The horizontal dashed line at zero indicates the null value, the solid horizontal lines indicate the upper and lower 95% confidence limits based on the assumption of independence.

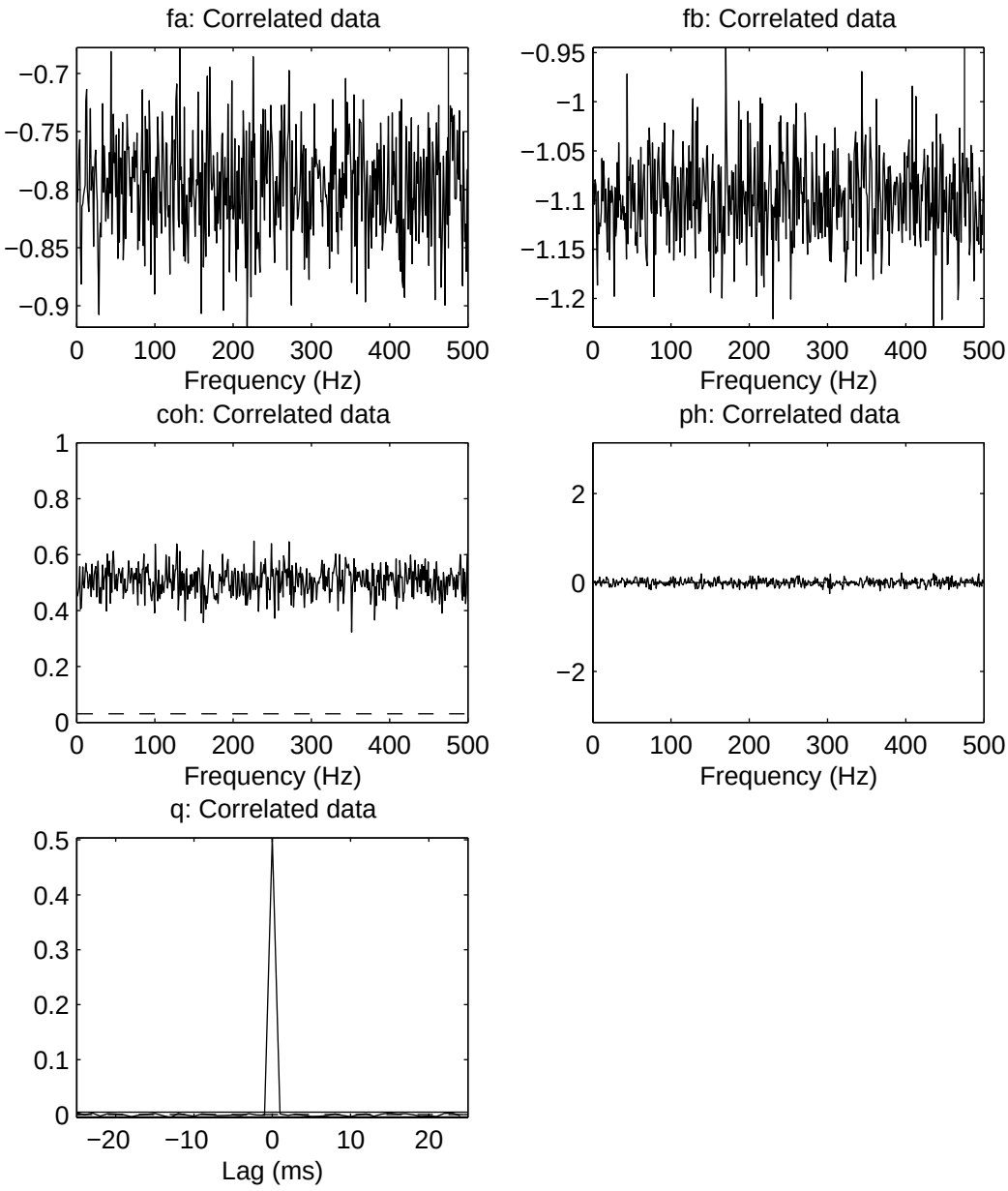

Figure 3: *Type 0 analysis.* Plot generated by `psp2` of output from `sp2a2_m1` for Type 0 analysis of correlated signals. Layout as in Figure 2.

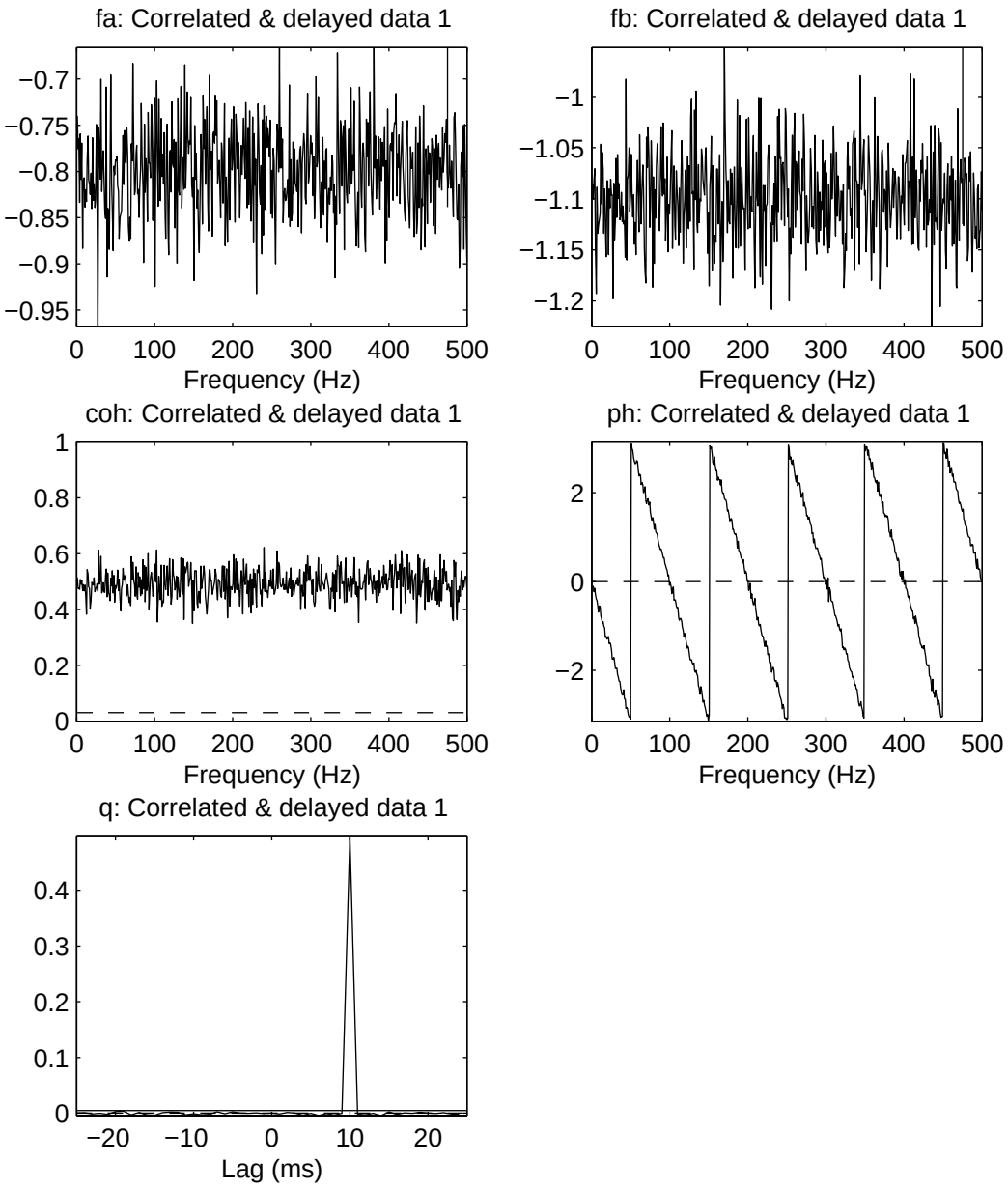

Figure 4: *Type 0 analysis.* Plot generated by `psp2` of output from `sp2a2_m1` for Type 0 analysis of correlated and delayed signals. Layout as in Figure 2.

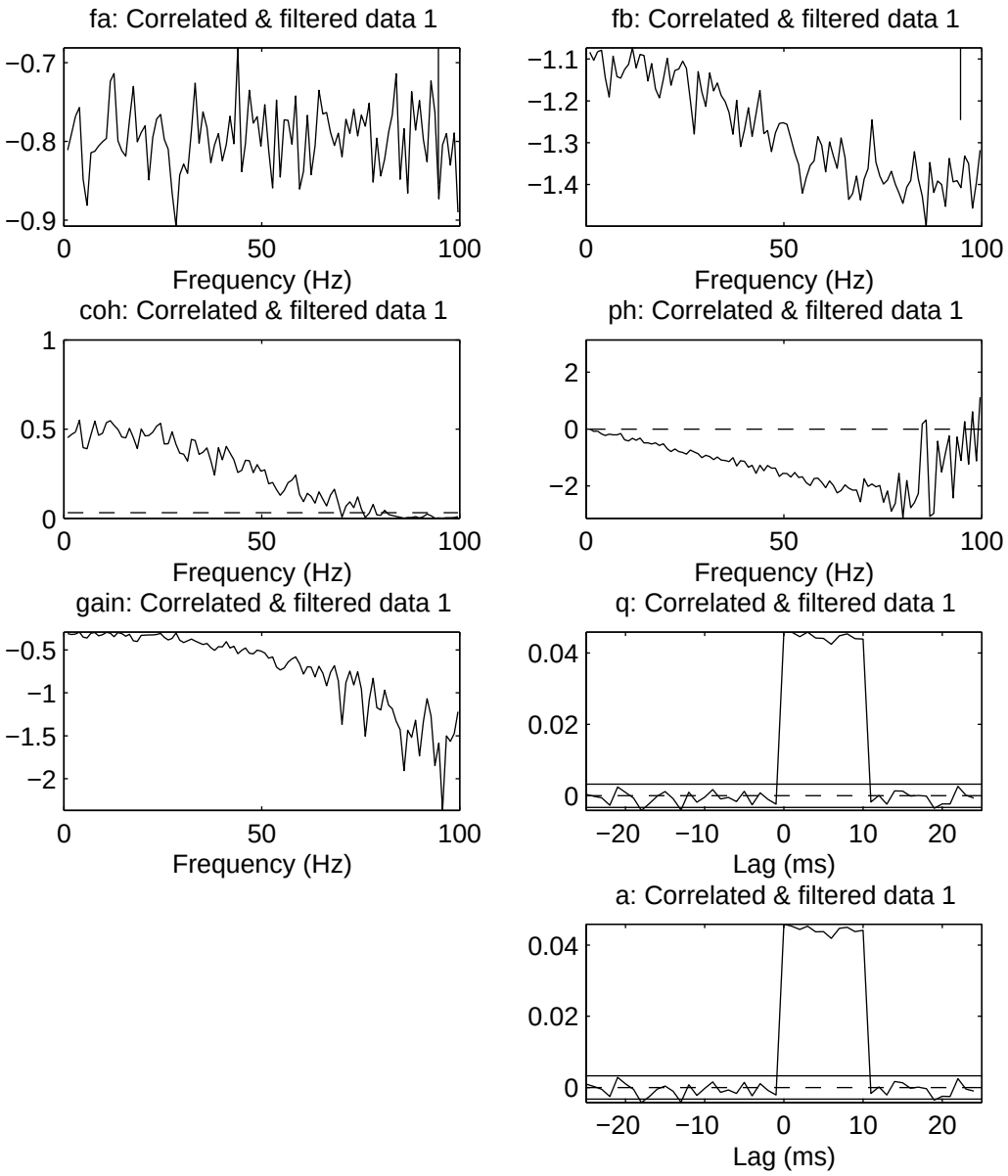

Figure 5: *Type 0 analysis.* Plot generated by `psp2s` of output from `sp2a2_m1` for Type 0 analysis of linearly filtered white noise. Figure includes plots of gain (3^{rd} row left, plotted as $\log_{10}$) and impulse response (bottom row). Confidence limits for other plots as described in Figure 2. Confidence limits for estimated impulse response have a similar interpretation to those for the cumulant density estimate.

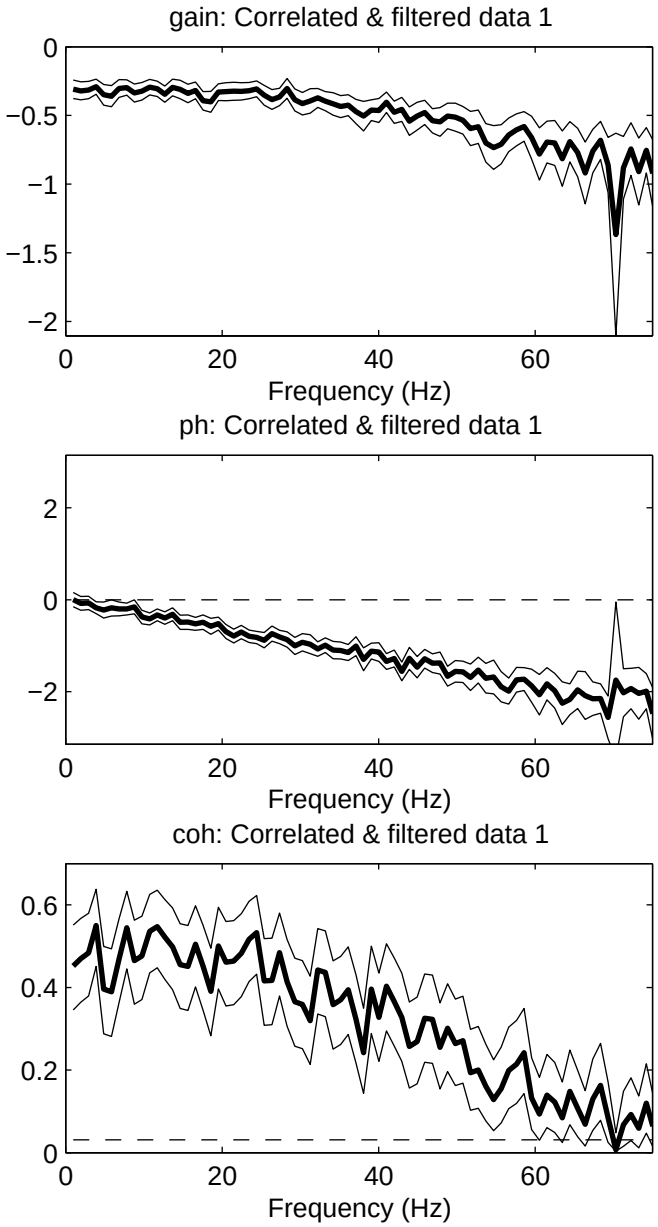

Figure 6: *Type 0 analysis.* Gain, phase and coherence from Figure 5, plotted with point wise confidence limits. Heavy lines are parameters estimates, thin lines are upper and lower 95% point wise confidence limits.

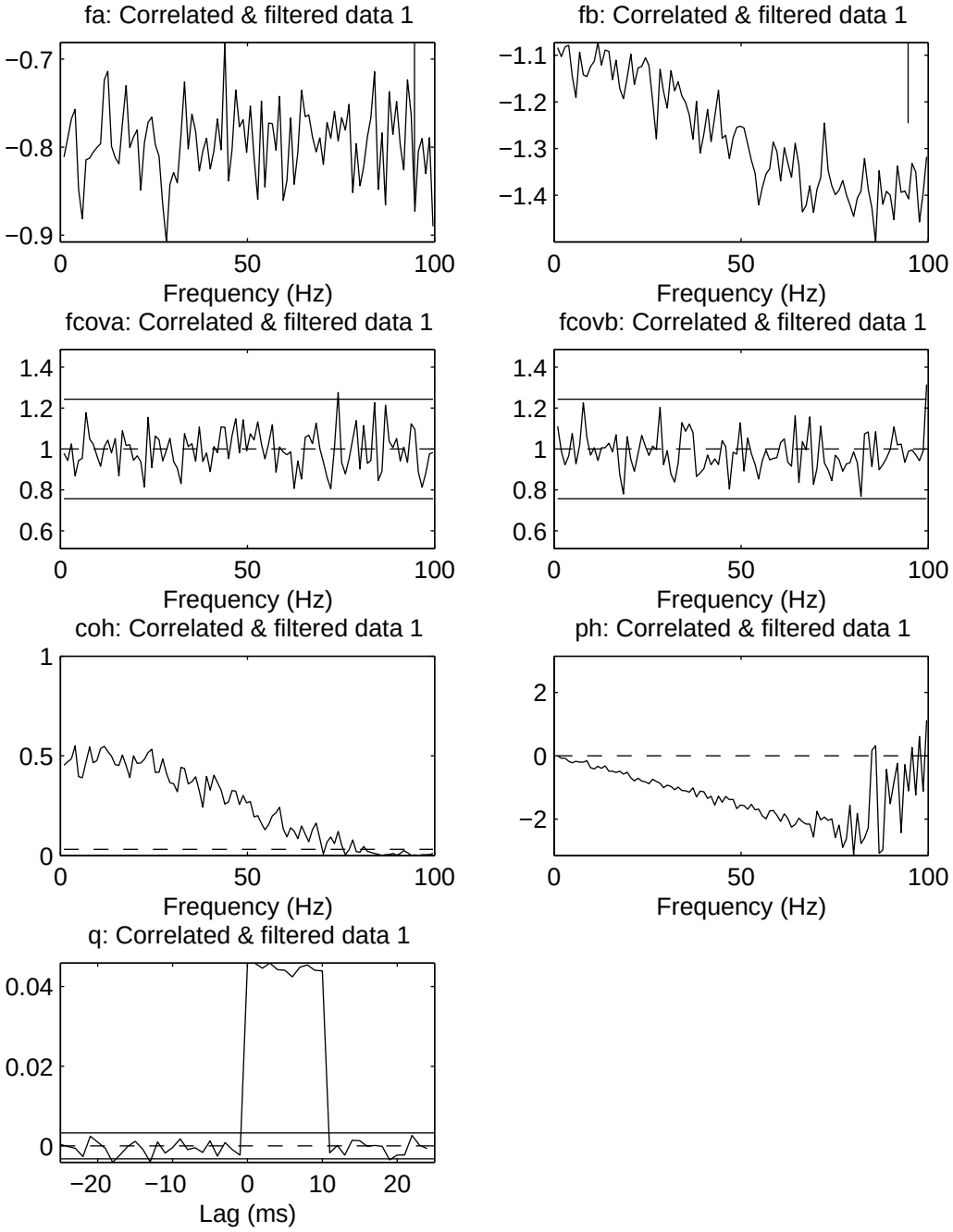

Figure 7: Plot generated by `psp2_cov` of output from `sp2a2_m1` for Type 0 analysis of linearly filtered white noise. Analysis and figure includes Periodogram COV plots for each channel (2^{nd} row). The dotted line at zero is the *NULL* value for the test, the solid horizontal lines indicate the upper and lower 95% confidence limits for the test. Confidence limits for other plots as described in Figure 2.

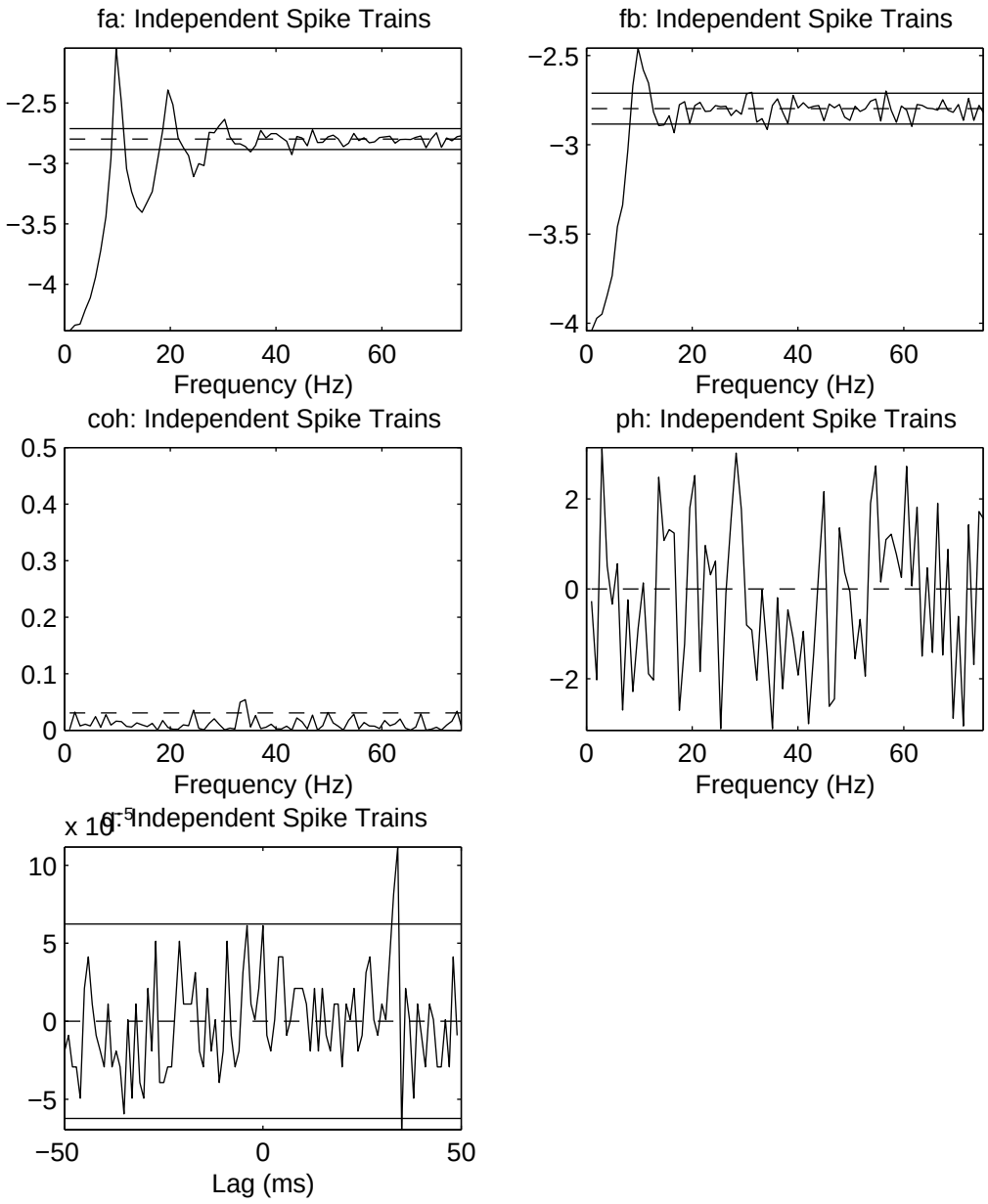

Figure 8: *Type 0 analysis.* Plot generated by `psp2` of output from `sp2_m1` for Type 0 analysis of independent spike trains. Layout as in Figure 2.

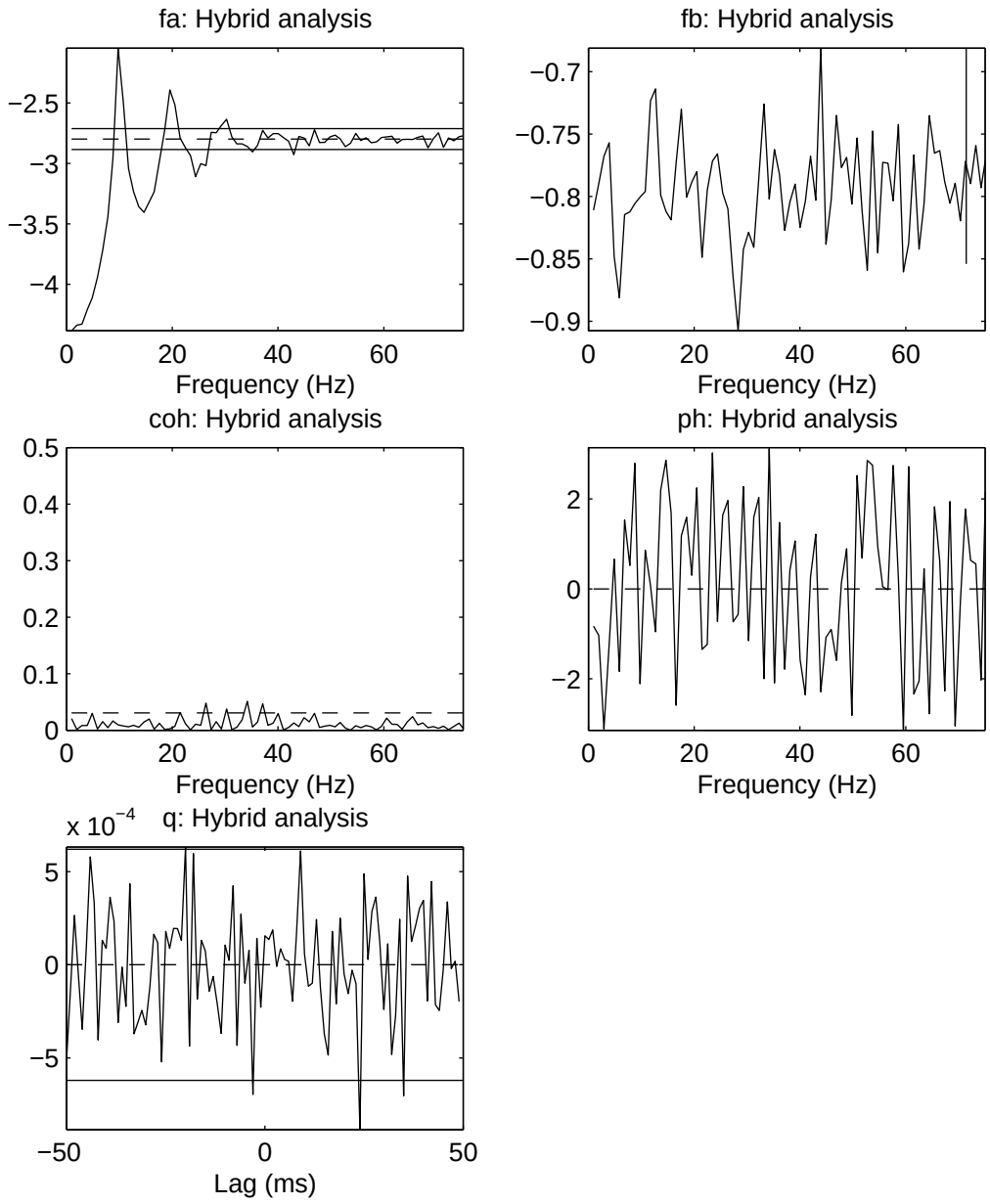

Figure 9: *Type 0 analysis.* Plot generated by `psp2` of output from `sp2a_m1` for Type 0 analysis of one spike train and one time series. Layout as in Figure 2.

9.2 Type 1 analysis

The script `sp2_type1_demo1` demonstrates the use of type 1 (see Fig. 1b) analysis on artificially generated time series and spike train data. When run it generates 6 figures, based on the same combinations of artificially generated data as the Type 0 demonstration script described above. Two of the six figures are illustrated here. Figure 10 illustrates the analysis of the 11 point moving average filtered data (*cf* Figure 5). Figure 11 shows the results for analysis of the two independent spike trains (*cf* Figure 8). For the particular run illustrated here, the 100000 data samples were split into 187 segments of data, with the number of data samples in each segment ranging from 60 to 1014 samples.

9.3 Type 2 analysis

The script `sp2_type2_demo1` demonstrates the use of type 2 analysis (see Fig. 1c) to undertake a time-frequency analysis on two channels of artificially generated time series data. The data consisted of 50 trials of independent white noise, each trial was 1000 samples (1000 ms) in duration. Two sines waves were added to both channels within each trial: 10 Hz from 300-400 ms, and 25 Hz from 500 - 600 ms. The amplitude of both sine waves were set to give a signal to noise ratio (SNR) within each channel of $-15dB$. The time frequency analysis was done by moving a sliding window of duration 250 samples (250 ms) across each trial with offset values ranging from 0 to 750 samples, with an increment of 50 samples (16 offset values in total). The five figures produced by the script are included below, along with some brief comments regarding interpretation. Figures 12, 13, 14 illustrate the output of the plotting function `psp2_tf`. Figure 12 shows the two time-dependent spectral estimates and the time-dependent coherence estimate. The time dependent coherence plot is reproduced in Figure 13, along with the time-dependent phase estimate, both plots in this figure include scale bars. The scale bar for the coherence plot has a horizontal line at the value of the upper 95% confidence limit (0.059 in this case). The contour plot for the time dependent coherence has the lowest contour height set at this 95% confidence level, regions where the estimated coherence is below this level are indicated by a single colour in the contour plot. For this example, the majority of the time-frequency plane falls below this significance level, as expected. Two regions of significant coherence are highlighted in figures 12 and 13. These are at offset values 100-350 ms around 10 Hz, and at offset values 350 - 550 ms around 25 Hz. The offset values give the start of each data segment in the analysis (see fig 1c), a more representative indication of timing may be derived by converting these offset values to the centre of each segment. The 10 Hz component results from data segments centred at 225-475 ms, the 25 Hz component from data segments centred at 475 - 675 ms. These time values encompass the actual values used to generate the data.

Figure 14 illustrates the third figure generated by the plotting function `sp2_tf`. This is the time dependent cumulant density. This function describes how the correlation as a function of time changes with the value of offset. The offset value in this plot is the same as the plots in figures 12, 13. The vertical axis is the time lag between the two signals. The main feature in this plot is the horizontal band around 0 ms time lag extending from 100 to 500 ms offset values. This feature, which has positive values, is indicating the temporal synchronization between the two signals that results from the common sinusoidal modulation. The scale bar for this time dependent cumulant density includes the upper and lower 95% confidence limits. Since the confidence limits for the cumulant density depend on the data analysed, it is unlikely that they will be constant as the offset changes. Thus, the limits for all offset values are included in the plot, as a guide to interpretation.

Figures 15 and 16 illustrate how a single section at a fixed offset value can be plotted from a type 2 analysis using the function `psp2`. Figure 15 is for an offset of 300 ms, which highlights the 10 Hz common sinusoidal modulation. Figure 16 is for an offset of 500 ms, this highlights the 25 Hz common sinusoidal modulation.

A time-frequency Type 2 analysis, as illustrated in figures 12, 13, 14 requires a range of offset values to be used. The plotting functions will only work with a minimum of 3 offset values, although in practice a greater number would normally be required to reliably detect any trend. This should be taken into account when designing experimental protocols. Results from a single offset Type 2 analysis can be plotted using `psp2`, and variants, as shown in figures 15, 16.

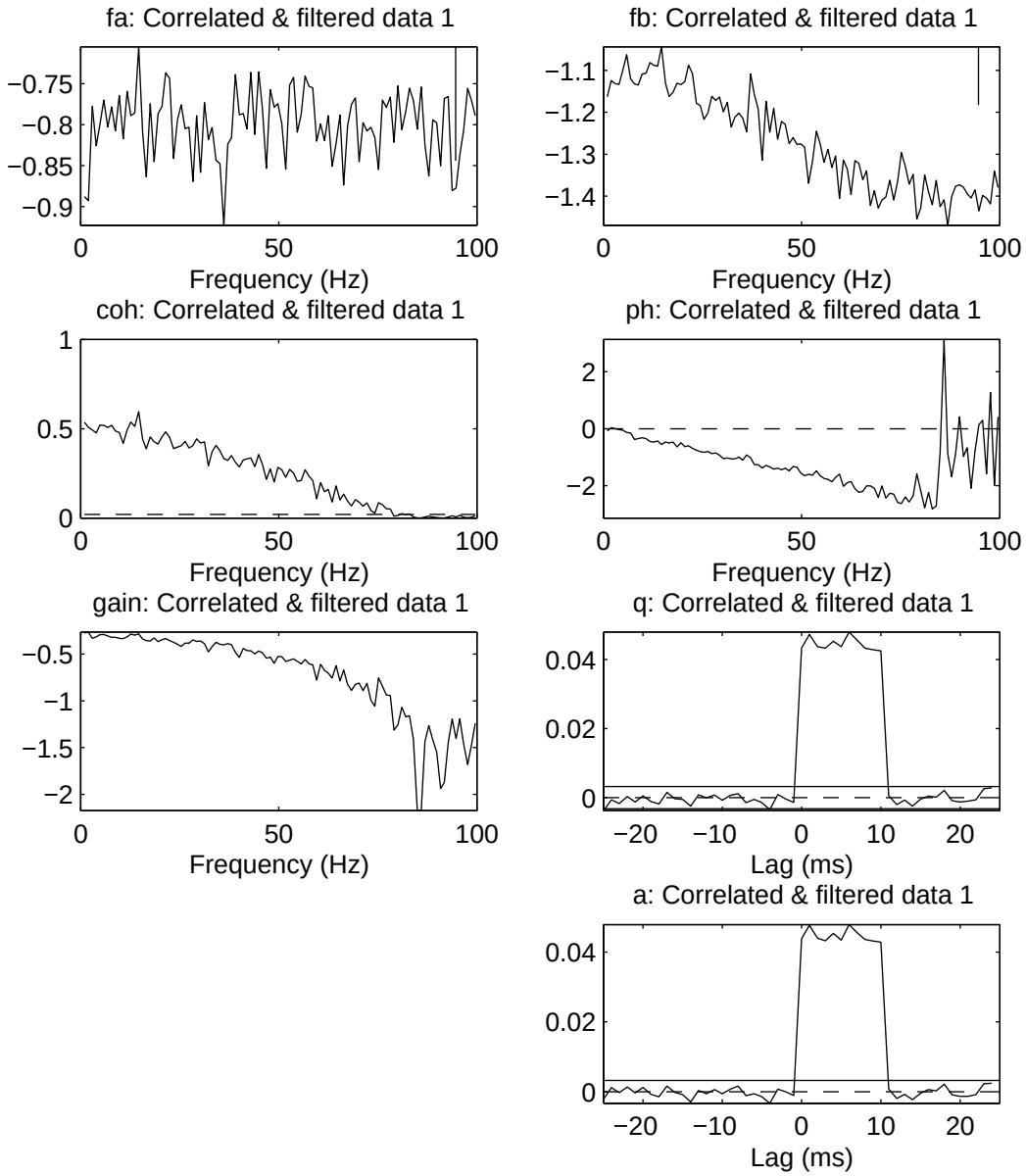

Figure 10: *Type 1 analysis.* Plot generated by `psp2s` of output from `sp2a2_m1` for Type 1 analysis of linearly filtered white noise. Layout same as Figure 5.

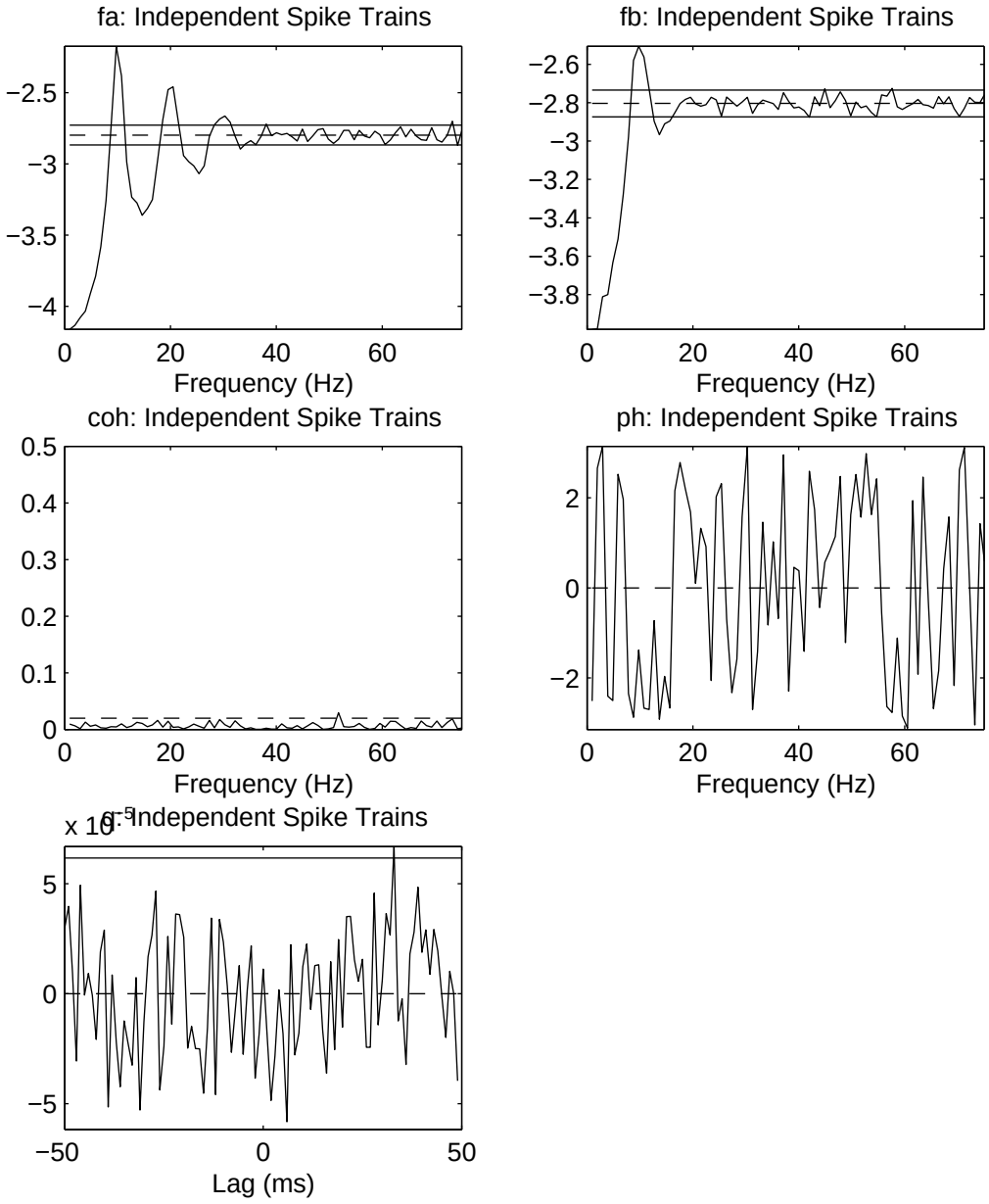

Figure 11: *Type 1 analysis.* Plot generated by `psp2` of output from `sp2_m1` for Type 1 analysis of independent spike trains. Layout as in Figure 8.

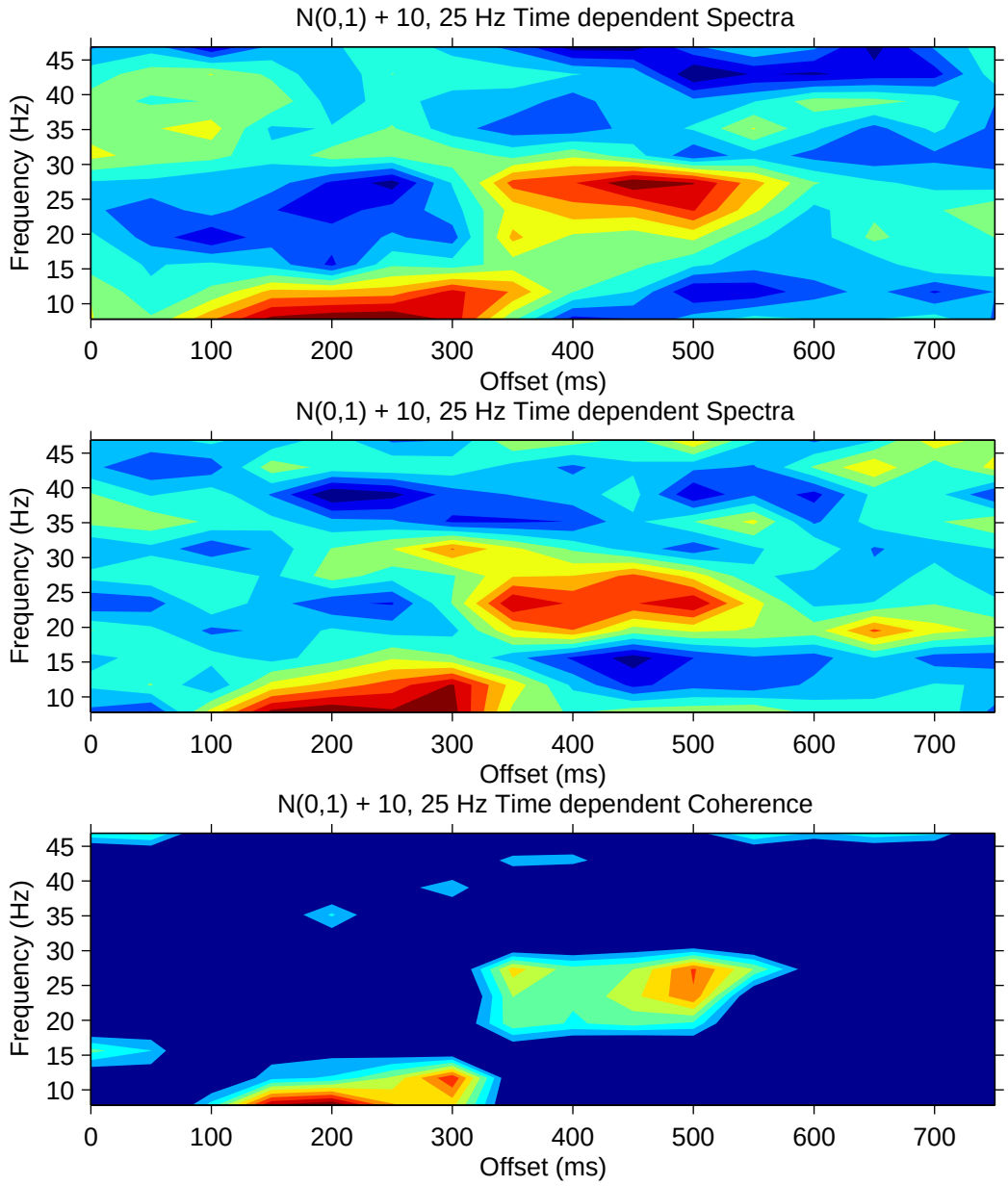

Figure 12: *Type 2 analysis.* Plot 1 of 3 generated by `sp2_tf` of output from `sp2a2_m1` for Type 2 analysis of uncorrelated white noise over repeat trials with two sine waves embedded within each trial (10 Hz from 300-400 ms; 25 Hz from 500 - 600 ms). Analysis done using 16 offset values from 0 to 750 ms in 50 ms increments. Upper and middle panel show contour plots of the two time-dependent spectral estimates. Lower panel shows a contour plot of the time dependent coherence estimate.

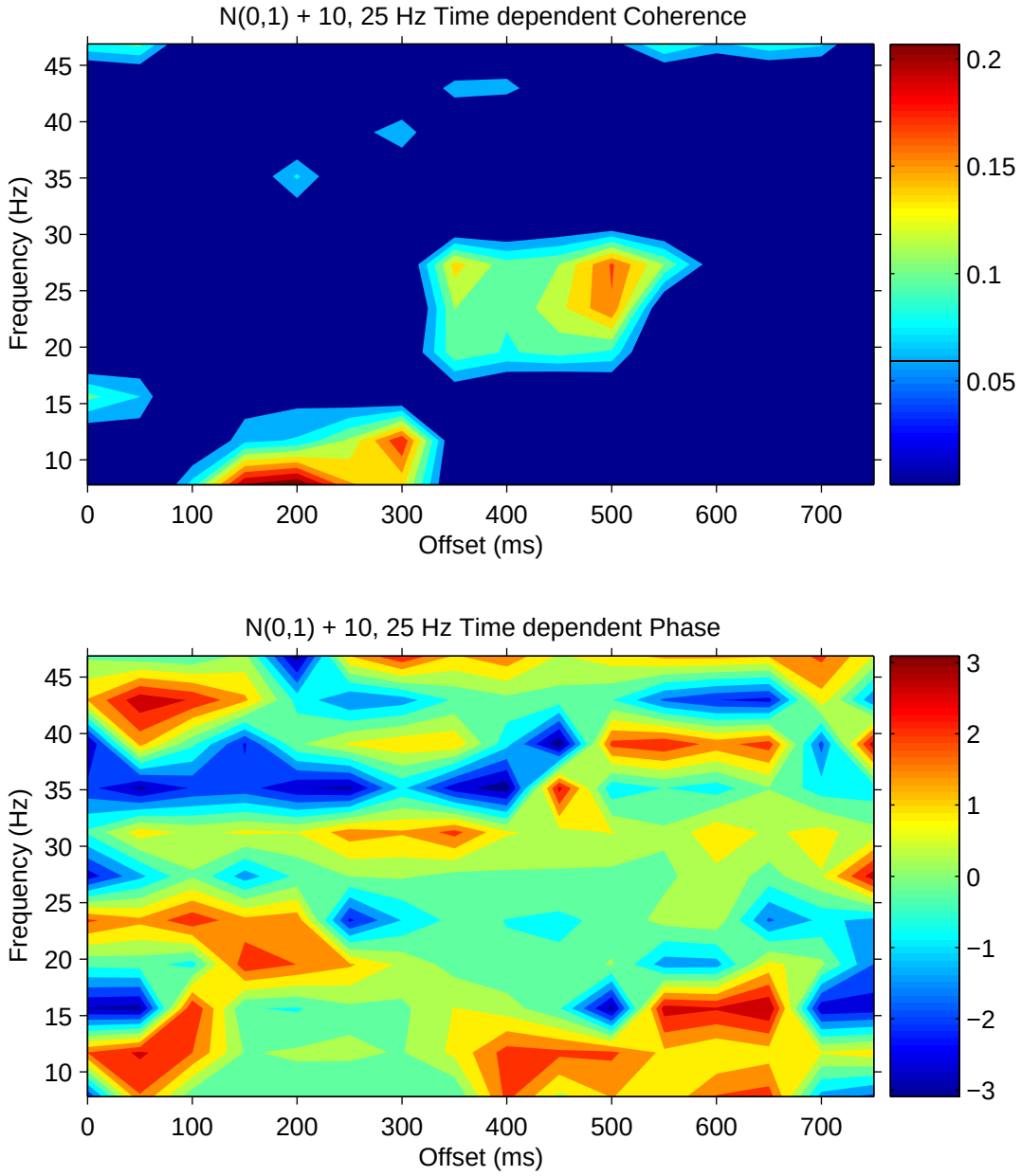

Figure 13: *Type 2 analysis.* Plot 2 of 3 generated by `sp2_tf` of output from `sp2a2_m1` for Type 2 analysis of uncorrelated white noise over repeat trials with two sine waves embedded within each trial (10 Hz from 300-400 ms; 25 Hz from 500 - 600 ms). Analysis done using 16 offset values from 0 to 750 ms in 50 ms increments. Upper panel shows contour plot of the time dependent coherence estimate. The solid horizontal line on the scale bar is the upper 95% confidence limit for the coherence estimate. All values below this level are indicated a single colour in the plot. Lower panel is a contour plot of the time dependent phase estimate.

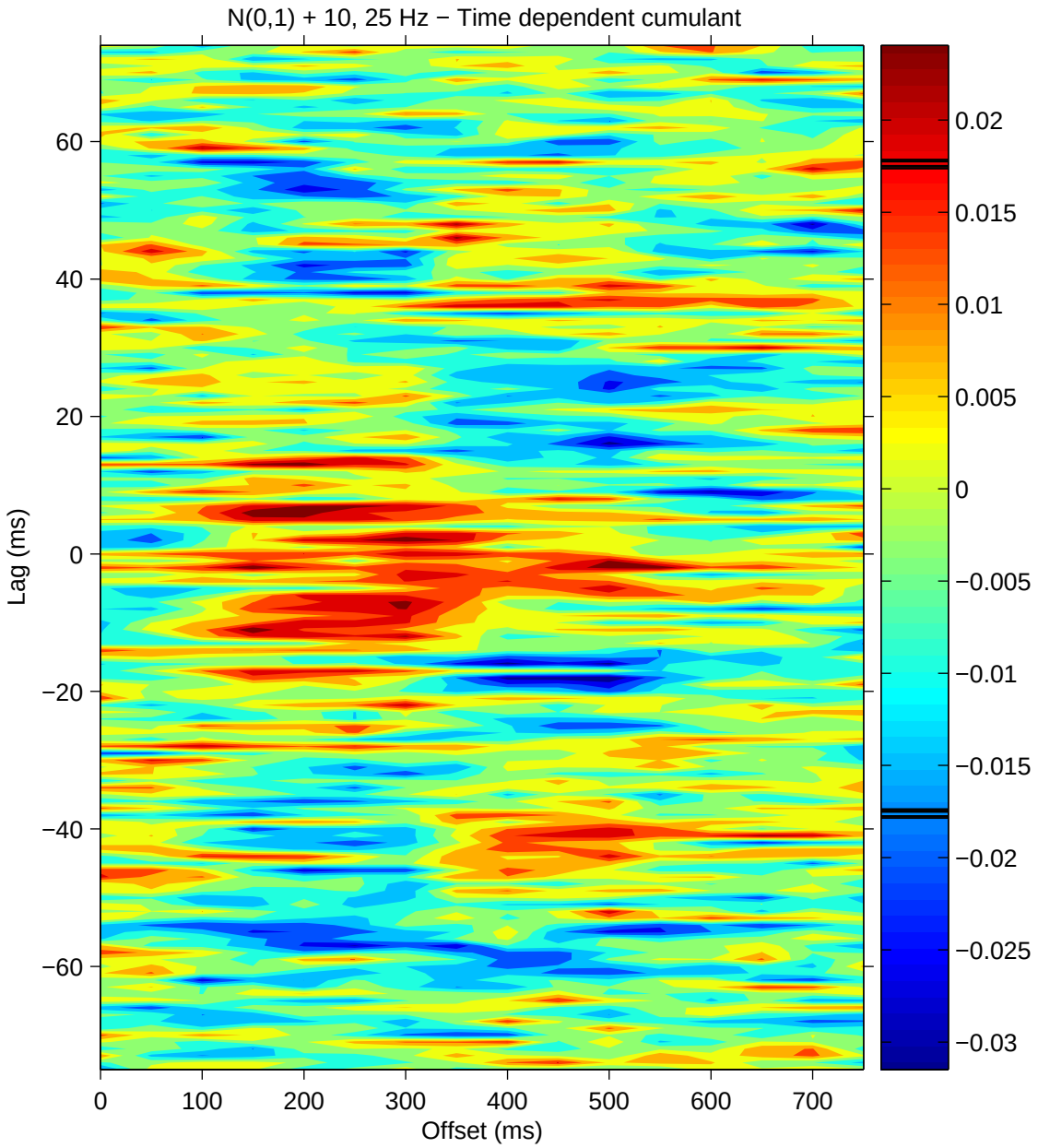

Figure 14: *Type 2 analysis.* Plot 3 of 3 generated by `sp2_tf` of output from `sp2a2_m1` for Type 2 analysis of uncorrelated white noise over repeat trials with two sine waves embedded within each trial (10 Hz from 300-400 ms; 25 Hz from 500 - 600 ms). Analysis done using 16 offset values from 0 to 750 ms in 50 ms increments. Plot of time dependent cumulant density estimate. The solid horizontal lines on the scale bar are the upper and lower 95% confidence limits for all of the 16 offset values included in the analysis.

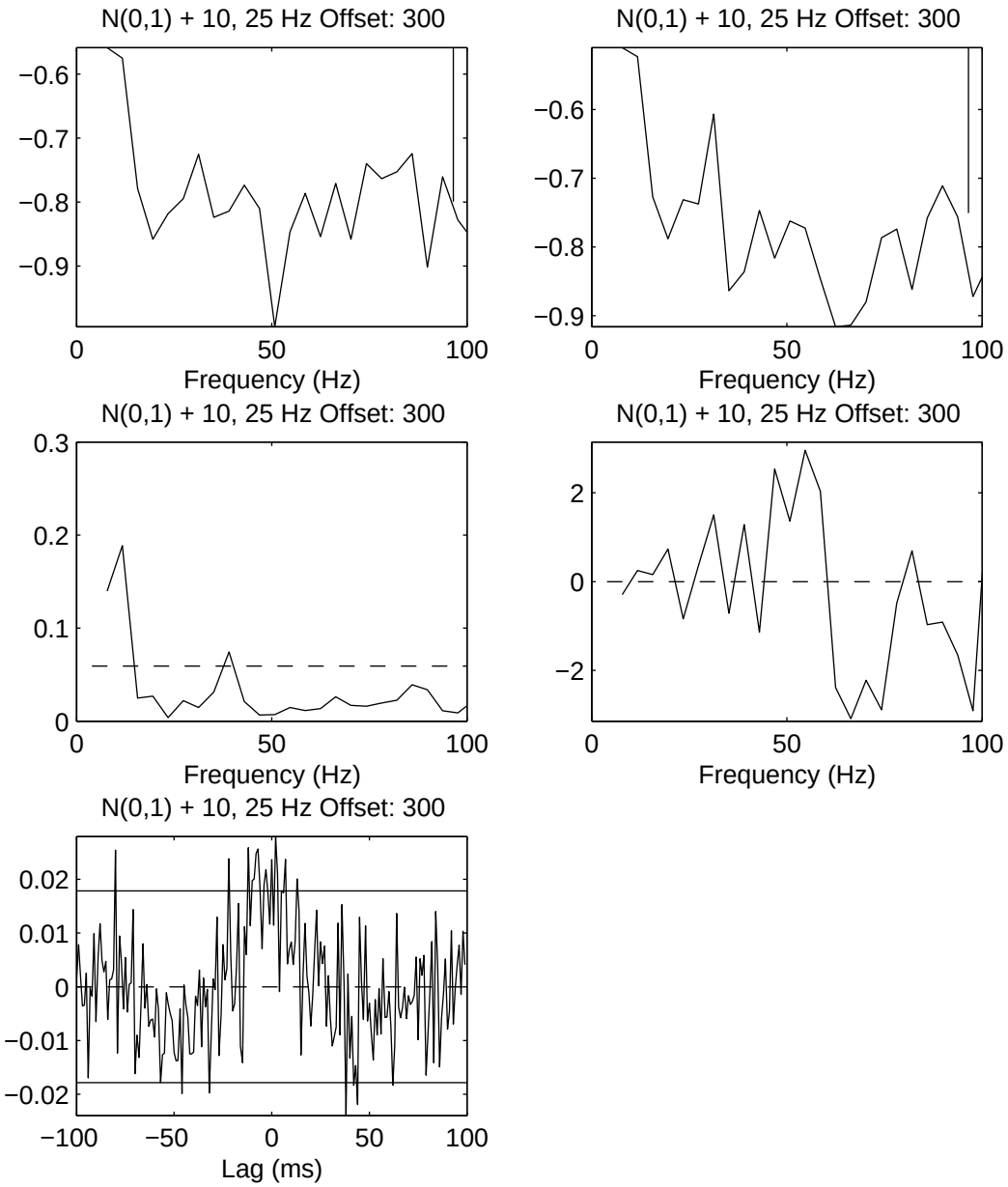

Figure 15: *Type 2 analysis.* Plot generated by `psp2` of of single section from output of `sp2a2_m1`, for Type 2 analysis of uncorrelated white noise over repeat trials with two sine waves embedded within each trial (10 Hz from 300-400 ms; 25 Hz from 500 - 600 ms). Figure is for a single offset value of 300 ms. Layout and confidence intervals are the same as for Figure 2.

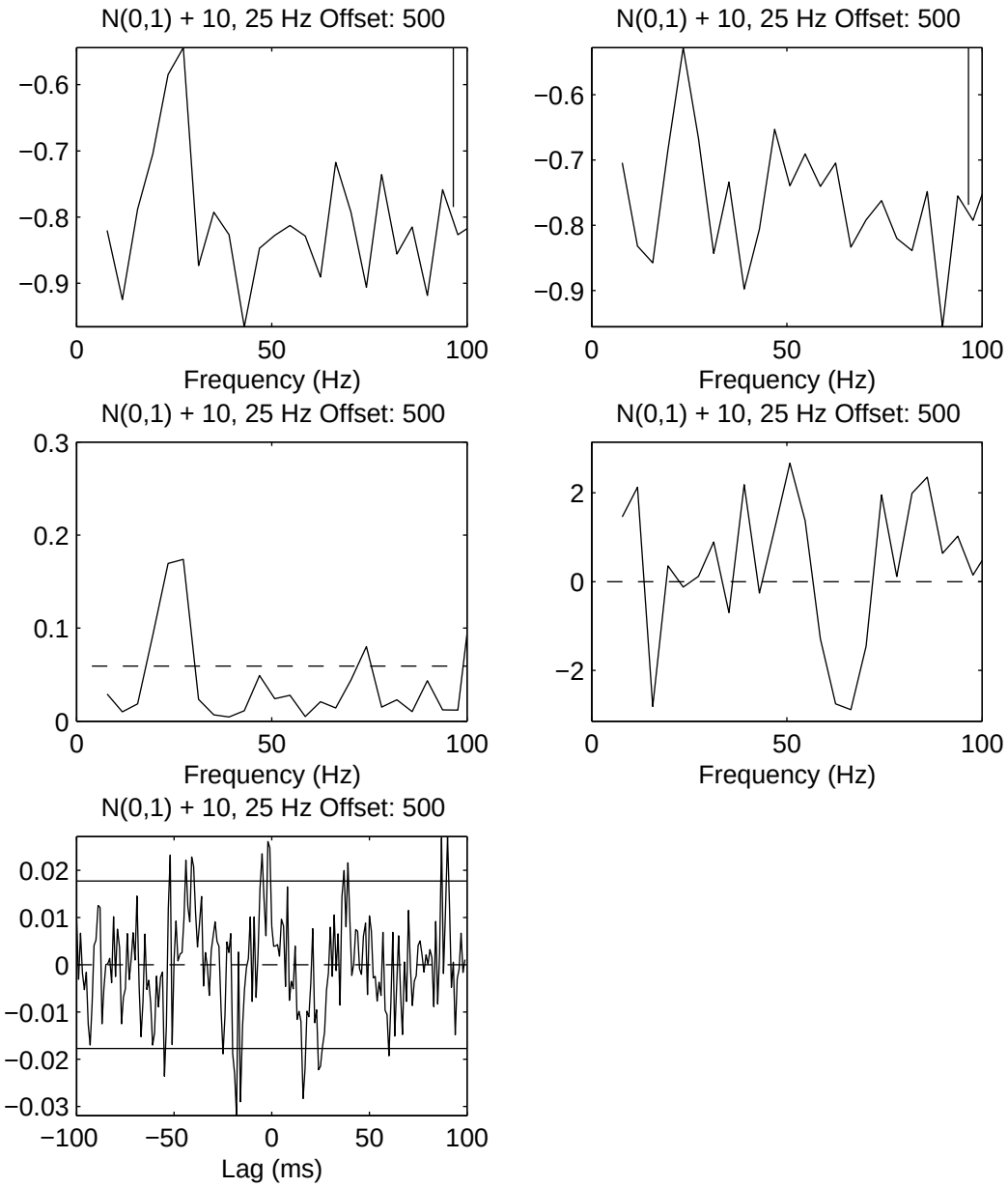

Figure 16: *Type 2 analysis.* Plot generated by `psp2` of single section from output of `sp2a2_m1`, for Type 2 analysis of uncorrelated white noise over repeat trials with two sine waves embedded within each trial (10 Hz from 300-400 ms; 25 Hz from 500 - 600 ms). Figure is for a single offset value of 500 ms. Layout and confidence intervals are the same as for Figure 2.

10 Comparison of spectra and comparison of coherence

10.1 Comparison of spectra

NeuroSpec 2.0 includes routines that can be used to assess and plot if there are statistically significant differences between two spectral estimates. The two spectral estimates should be derived independently, this generally means they should be calculated from data sampled at different times. Applying the test to two spectral estimates derived from simultaneously sampled data is not recommended. Any correlation between the two signals will invalidate the assumption of independence, in addition confidence limits will not be valid. It is difficult to interpret the results of such a comparison in a meaningful way. A typical protocol would involve recording the response of a system over two distinct time periods, possibly in response to two different sets of input conditions. The comparison of spectra test can then be used to determine if the output characteristics of the system have changed.

Comparison of spectra is undertaken at each frequency, thus the two spectra should have identical Fourier frequencies. This is achieved if the two records have the same sampling rate and DFT segment length, this is checked by the routine. The routine allows comparison of spectra that have been estimated using a different type of analysis (Type 0, 1, 2). However caution should be exercised in doing such a comparison, principally because different zero padding regimes will alter the 'effective' resolution (or bandwidth) of the estimate.

The function to calculate the difference of spectra is `sp2_compf`. Assuming that `sc1`, `cl1` and `sc2`, `cl2` are the spectral coefficient matrices and confidence limit structures related to the spectra to be compared, the comparison is done using the command:

```
[f3,cl3]=sp2_compf(sc1,cl1,2,sc2,cl2,2);
```

The 3rd and 6th parameters indicate which column of the spectral coefficient matrix is to be used. The parameters should take only the values 1 or 2; 1 to use the input spectrum, 2 to use the output spectrum. The output of this routine has a similar format to the output of the NeuroSpec core routines. The matrix `f3` contains the result of the comparison, the structure `cl3` contains the parameters required to plot the results. The comparison of spectra test is calculated using a log ratio of the two spectral estimates. The ratio is calculated as $log_{10}(\texttt{sc1}/\texttt{sc2})$. The null value, based on the hypothesis of equal spectra, is zero ($log_{10}(1) = 0$). Confidence limits are set using an F-distribution. This test can support comparison of spectra estimated from different record lengths (unlike the comparison of coherence test described below). The results of the test are plotted using the function `psp_compf1`:

```
psp_compf1(f3,cl3,200)
```

The third parameter controls the frequency range over which to plot the results. An optional fourth parameter can be used to scale the y axis. A demonstration script is included to illustrate the use of the comparison of spectra and the comparison of coherence tests. The comparison of spectra test is applied to two spectra generated independently by applying moving average low pass filters of lengths 11 points and 21 points to separate white noise records. The demonstration script is `sp2_comp_demo1`, it generates four figures. The first two figures illustrate a two channel analysis of each white noise and the filtered version (using routines `sp2a2_m1` and `psp2`). The third figure, generated by `psp_compf1`, is reproduced in Figure 17. This shows the two spectral estimates and the results of the comparison of spectra test (lower plot). The test exceeds the upper 95% confidence limit over the frequency range 20 Hz to 50 Hz, reflecting the reduction in signal bandwidth with a 21 point moving average filter compared with the 11 point filter. The fourth figure generated by `sp2_comp_demo1` illustrates the comparison of coherence test, described in the next section.

10.2 Comparison of coherence

The function to compare two coherence estimates is `sp2_compcoh`. The coherence estimates must be independent, *i.e.* they should be calculated from bivariate data sampled at different times. Any correlation between the two bivariate data sets will invalidate the assumption of independence, in addition confidence limits will not be valid. It is difficult to interpret the results of such a comparison in a meaningful way. Comparison of coherence is undertaken at each frequency, thus the two estimates should have identical Fourier frequencies. The routine checks that both estimates have identical sampling rates and

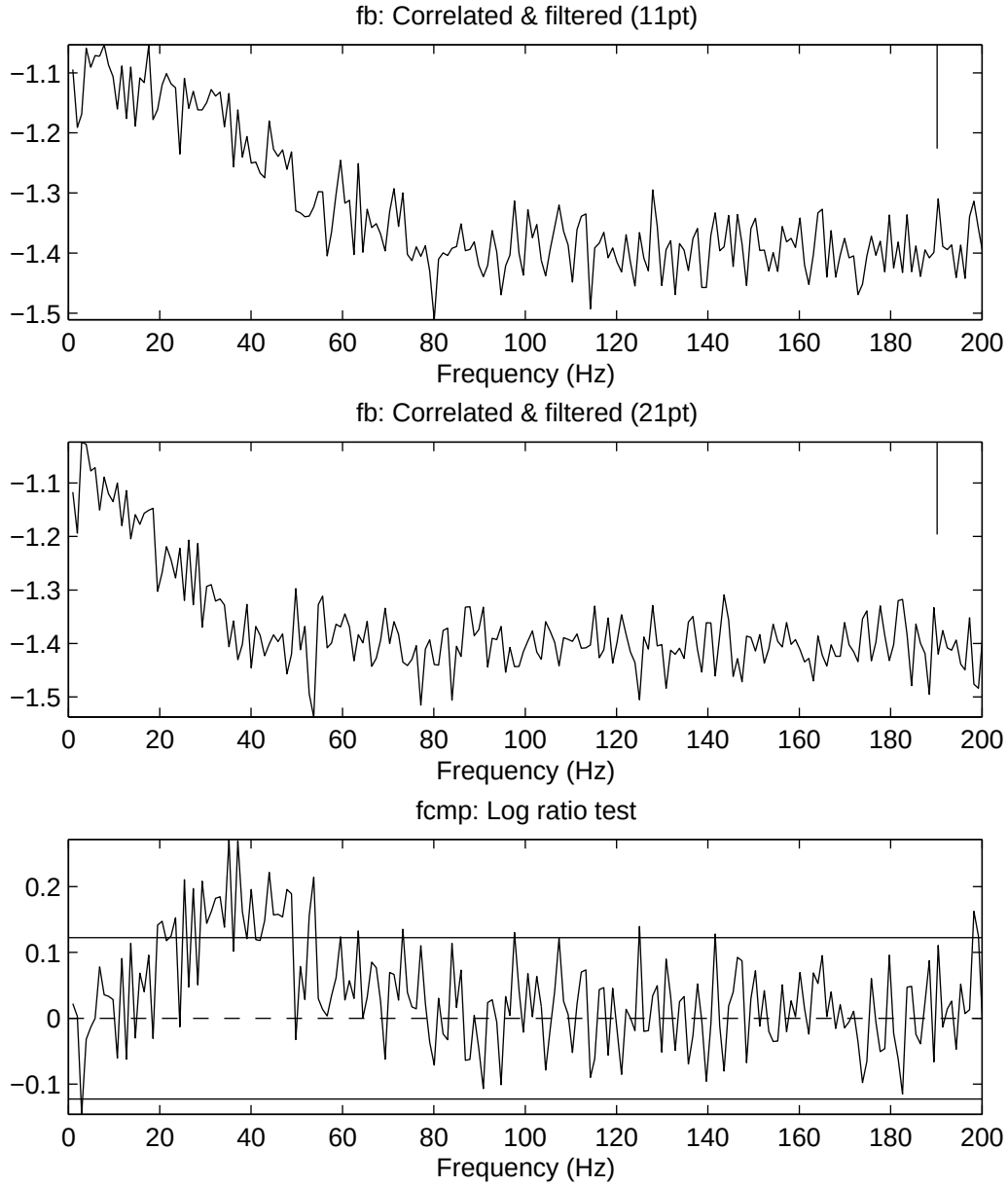

Figure 17: *Comparison of spectra test.* Plot generated by `psp_compf1` of output from `sp2_compf` for comparison of spectra test. (Top) Spectral estimate for a white noise sequence filtered by an 11 point moving average filter. (Middle) Spectral estimate for white noise sequence filtered by a 21 point moving average filter. (Lower) Difference of spectra test applied to the two estimates. The horizontal dashed line at zero indicates the null value, the solid horizontal lines indicate the upper and lower 95% confidence limits based on the hypothesis of equal values at each Fourier frequency.

DFT segment lengths. In addition, the comparison of coherence test, has the additional restriction that both estimates are derived from the same number of segments (Comparison of coherence estimates with different numbers of segments can be done using the pooled coherence technique described below). The comments in the previous section regarding comparison of estimates based on different types of analysis are also applicable to this test.

The syntax for this function is

```
[f4,cl4]=sp2_compcoh(sc1,cl1,sc2,cl2);
```

where `sc1`, `cl1` and `sc2`, `cl2` are the spectral coefficient matrices and confidence limit structures for the two coherence estimates. This test has the form of a subtraction, where the estimate in `sc2` is subtracted from that in `sc1`. The null value, based on the hypothesis of equal coherence estimates, is zero. Upper and lower 95% confidence limits are included. Values of the test which exceed the upper limit indicate that the coherence estimate in `sc1` exceeds that in `sc2`, values below the lower 95% confidence limit indicate that the `sc1` estimate is smaller than the `sc2` estimate. The results of the test are plotted using the function `psp_compcoh1`:

```
psp_compcoh1(f4,cl4,200,1)
```

The third parameter controls the frequency range over which to plot the results, the optional fourth parameter controls the scaling of the y axis.

Figure 18 illustrates the fourth figure generated by the demonstration script `sp2_comp_demo1`. This uses the same data as the difference of spectra test illustrated in Fig. 17. The two coherence estimates were generated independently, from the input and output of different moving average filters driven by white noise. The difference of coherence test exceeds the upper 95% confidence limit over the frequency range 20 Hz to 60 Hz, indicating a significant difference in the two coherence estimates over this frequency range.

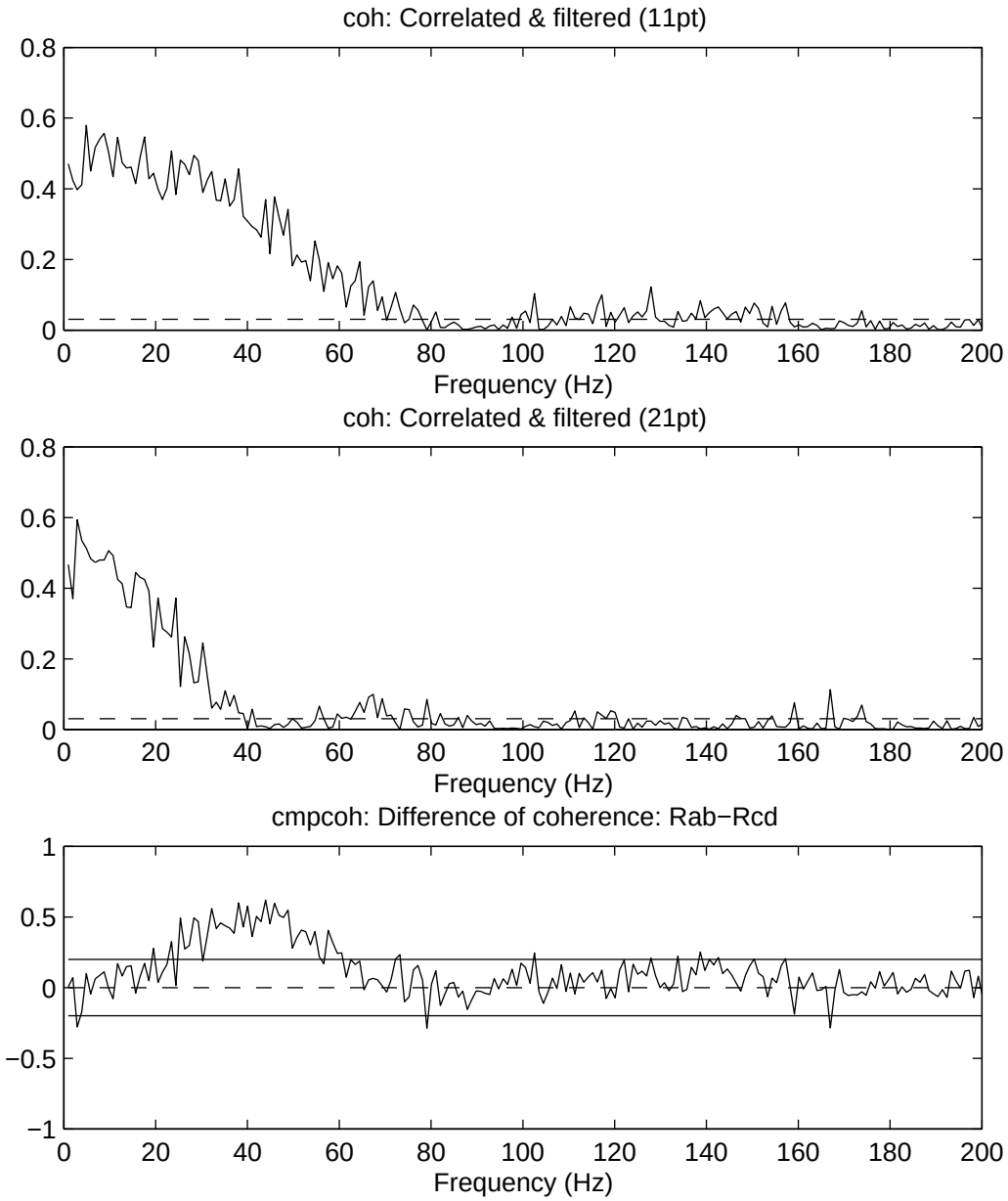

Figure 18: *Comparison of coherence test.* Plot generated by `psp_compcoh1` of output from `sp2_compcoh` for comparison of spectra test. (Top) Coherence estimate between the input and output of an 11 point point moving average filter driven by white noise. The horizontal dashed line in the coherence estimate is the upper 95% confidence limit based on the assumption of independence. (Middle) Coherence estimate between the input and output of a 21 point point moving average filter driven by white noise. (Lower) Difference of coherence test applied to the two estimates. The horizontal dashed line at zero indicates the null value, the solid horizontal lines indicate the upper and lower 95% confidence limits based on the hypothesis of equal coherence values at each Fourier frequency.

11 Pooled spectra and pooled coherence

NeuroSpec 2.0 includes routines to undertake a population analysis across several independent pairs of signals. The population analysis constructs estimates of the same parameters as the core routines, except these are now population, or pooled parameters. The framework also includes four additional parameters that extend the scope of the population analysis. These four additional parameters are

1. A χ^2 extended difference of spectra test on the set of input spectra.
2. A χ^2 extended difference of spectra test on the set of output spectra.
3. A χ^2 extended difference of coherence test.
4. A histogram of the fraction of individual coherence estimates that have significant values.

The three χ^2 tests are based on a *NULL* hypothesis of equal values of log input spectra, log output spectra or coherence within the population. A significant value for this test indicates that the *NULL* hypothesis is not valid. Like the difference of spectra and difference of coherence tests described above, this test is applied, and interpreted, at each frequency of interest. The coherence histogram indicates the proportion (normalized from 0 to 1) of individual coherence estimates that have significant values at each frequency. This is generated by counting the frequency bins where the individual coherence estimates exceed the estimated upper 95% confidence limit.

Calculation of pooled spectra is done iteratively, using two steps. First the spectral coefficients are generated for each pair of signals, these are stored in the optional fourth output, `sc`, from the core routines. Second, these coefficients are used either to create a new set, or update an existing set of the population parameters. The function to create or update the pooled parameter estimates is `pool_sc`. There are two valid ways to call this function:

```
[plf1,plv1]=pool_sc(sc1,cl1);
```

```
[plf1,plv1]=pool_sc(sc1,cl1,plf1,plv1);
```

where `sc1`, `cl1` are the spectral coefficient matrices and confidence limit structures to be included in the pooled analysis. The two returned variables `plf1`, `plv1` contain the updated results of the pooled analysis, `plf1` is a matrix with one row for each frequency, `plv1` is a structure containing scalar values related to the pooled analysis. The first syntax above is used to create a new pooled analysis, subsequent calls should use the lower syntax where `plf1`, `plv1` are passed as input arguments to be updated. Passing an empty matrix as a RHS argument (`plf1=[];`) will also create a new pooled analysis. Once all data sets in the population have been processed the data in the variables `plf1`, `plv1` is converted to a form suitable for plotting using the command:

```
[f,t,cl]=pool_sc_out(plf1,plv1);
```

where `f`, `t` and `cl` are similar (but not identical) to the output from the NeuroSpec core routines. The results can then be plotted using one of four plotting functions, which generate plots of the same five parameters as `psp2`, and include the following additional plots:

`psp2_pool6`	6 plots, including χ^2 difference of coherence test.
`psp2_pool8`	8 plots, as `psp2_pool6`, including coherence histogram & 2^{nd} pooled coherence.
`psp2_pool8_chi3`	8 plots, with 3 χ^2 extended difference tests & coherence histogram, no cumulant.
`psp2_pool9_chi3`	9 plots, as `psp2_pool8_chi3`, also include time domain pooled cumulant estimate.

The function `psp2_pool8_chi3` does not generate any time domain plots and therefore requires a reduced set of input parameters. The pooled coherence estimate plotted in `psp2_pool6`, `psp2_pool8_chi3` and `psp2_pool9_chi3` is based on the technique of pooling coherency estimates. The second pooled coherence estimate plotted in `psp2_pool8` is generated from pooled spectra. The relative merits of the two different pooled coherence estimates are discussed in the references section (below). The recommended method is the pooled coherency based estimate.

As with the comparison of spectra and comparison of coherence tests, a pooled spectral analysis should be done using the same analysis parameters on each data set. In practice this means using the same segment length on data sets with the same sampling rate. The function `pool_scf` includes a check on these parameters. Optional parameters are not checked for consistency, however, it is likely that similar options would be applicable to any data sets from the same population (such as rectification). Options which alter the effective resolution of estimates, such as hanning and data windowing, should be kept constant across all records in a population. Pooling can be done on Type 0, Type 1, or a single offset Type 2 analysis. Pooling over a range of offset values in a Type 2 analysis should be done separately at each offset. The function `pool_scf` includes a check that the inputs `sc` and `cl` have the correct number of dimensions. There is no restriction on pooling data from different types of analysis (Type 0, 1, 2), again, caution is advised in such a case.

A demonstration script, `sp2_pool_demo1`, is included to illustrate pooled analysis. The script generates ten bivariate data sets for pooling by filtering white noise data. The last data set has an additional 10Hz sinusoidal component included in both channels, which appears in the individual spectra and coherence estimate for the last set of data. The script generates four figures, one each using the plotting functions `psp2_pool6`, `psp2_pool8`, `psp2_pool8_chi3` and `psp2_pool9_chi3`. The 6 panel figure is reproduced in figure 19. The χ^2 difference of coherence test (lower left panel) highlights the difference at 10 Hz in the coherence estimates. The 10 Hz component is also present in the pooled spectra (top plots). Figure 20 shows the figure with 8 plots, generated by `psp2_pool8`, which includes the additional pooled coherence estimate derived from pooled spectra (3^{rd} row, left), and the coherence histogram plot (lower left). A small difference is evident in the two pooled coherence estimates at 10 Hz. This is the frequency where the individual estimates differ, which is highlighted in the χ^2 test. Differences in the two pooled coherence estimates may well arise because the null hypothesis of equal coherence is not valid. In most circumstances the pooled coherence estimate from figure 19 (middle row, left) should be used. The output from `psp2_pool8_chi3` is illustrated in figure 21, this 8 panel figure includes all three χ^2 tests, one for each set of spectra (input and output) and one for coherence. The χ^2 tests for each set of spectra are calculated in a similar way to the χ^2 extended difference of coherence test, and have the same confidence limit. The output from `psp2_pool9_chi3` is similar to that from `psp2_pool8_chi3`, except an additional plot showing the pooled cumulant density estimate is included.

11.1 Note about use of pooled analysis for multivariate data

A key assumption in calculating pooled spectra and coherence is that the data sets to be pooled are *independent*. Like the comparison of coherence test described above, this means that (in general) each pair of signals should have been sampled or generated at different times. Attempting to pool a number of bivariate data sets that have all been sampled at the same time is not recommended. Any correlation between the signals will invalidate the assumption of independence, and confidence limits will not be valid. This factor should be taken into account when designing experimental protocols with a view to undertaking pooled analysis on the data. If a number of simultaneously recorded signals are the only data available for analysis, then this could be analysed using higher order multivariate approaches (one such approach will be included in future releases of NeuroSpec). Attempting a pooled analysis by combining all possible pairwise combinations of signals will invalidate the assumption of independence, and is not strictly a population analysis since it represents only a single sample from the system under study.

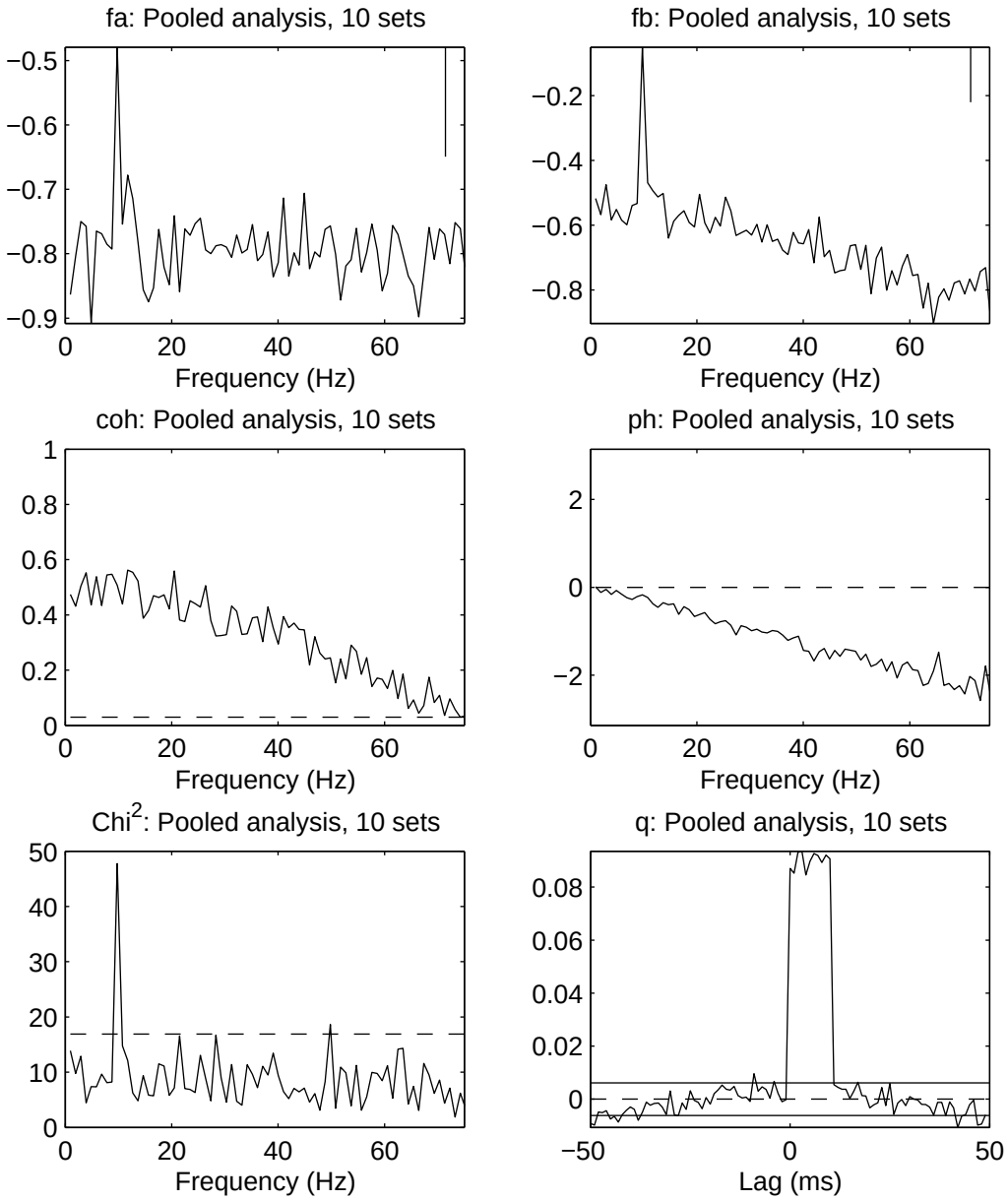

Figure 19: *Pooled analysis.* Plot generated by `psp2_pool6` of output from `pool_scf_out`, `pool_scf`. Results from analysis of 10 records, each record generated by filtering a white noise sequence with an 11 point moving average filter. Last record has an additional 10 Hz sine component added to each channel. Top two graphs are plots of the pooled spectral estimates (plotted as $\log_{10}$), for channel 1 (white noise, left) and channel 2 (filtered, right). The magnitude of the 95% confidence limit indicated by the the vertical bar in the top right of each plot. The centre two plots show the pooled coherence (left) and pooled phase (right) estimates. The horizontal dashed line in the coherence estimate is the upper 95% confidence limit based on the assumption of independence. The lower left plot is the χ^2 extended difference of coherence estimate. The horizontal dashed line is the upper 95% confidence limit based on the null hypothesis of equal coherence estimates in the population of records at each frequency. The isolated peak at 10 Hz suggests the null hypothesis is not valid at this frequency. The lower right plot shows the pooled cumulant density estimate, plotted against time lag in ms. The horizontal dashed line at zero indicates the null value, the solid horizontal lines indicate the upper and lower 95% confidence limits based on the assumption of independence.

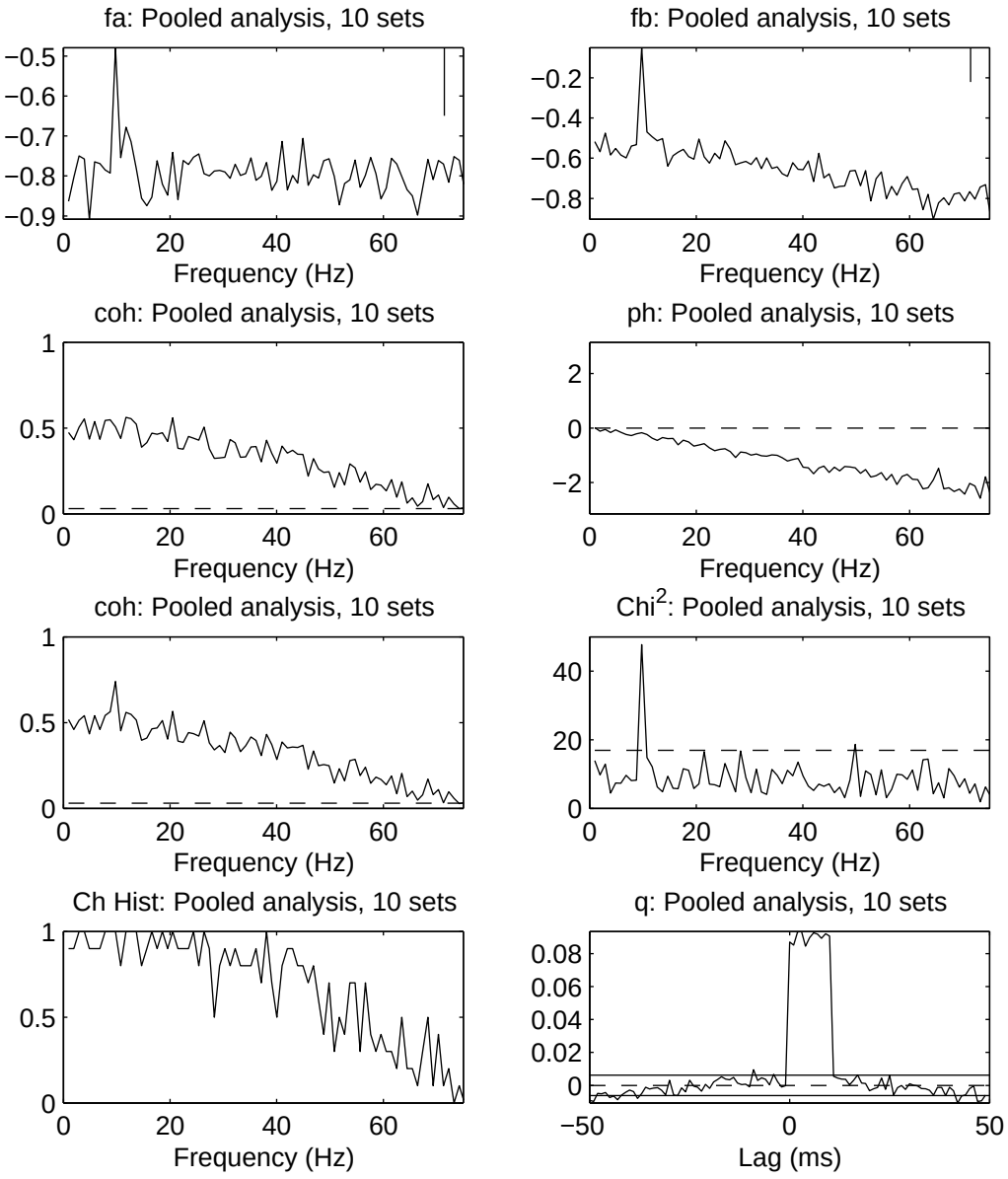

Figure 20: *Pooled analysis.* Results from same analysis as illustrated in figure 19, but plotted with `psp2_pool8`. First row illustrates two pooled spectral estimates, second row shows pooled coherence and pooled phase. Third row shows alternative (pooled spectra based) estimate of pooled coherence and χ^2 extended difference of coherence test. Fourth row illustrates the normalised histogram showing the fraction of individual coherence estimates that exhibit significant values at each frequency (left), and the pooled cumulant density estimate (right). Confidence limits as in figure 19.

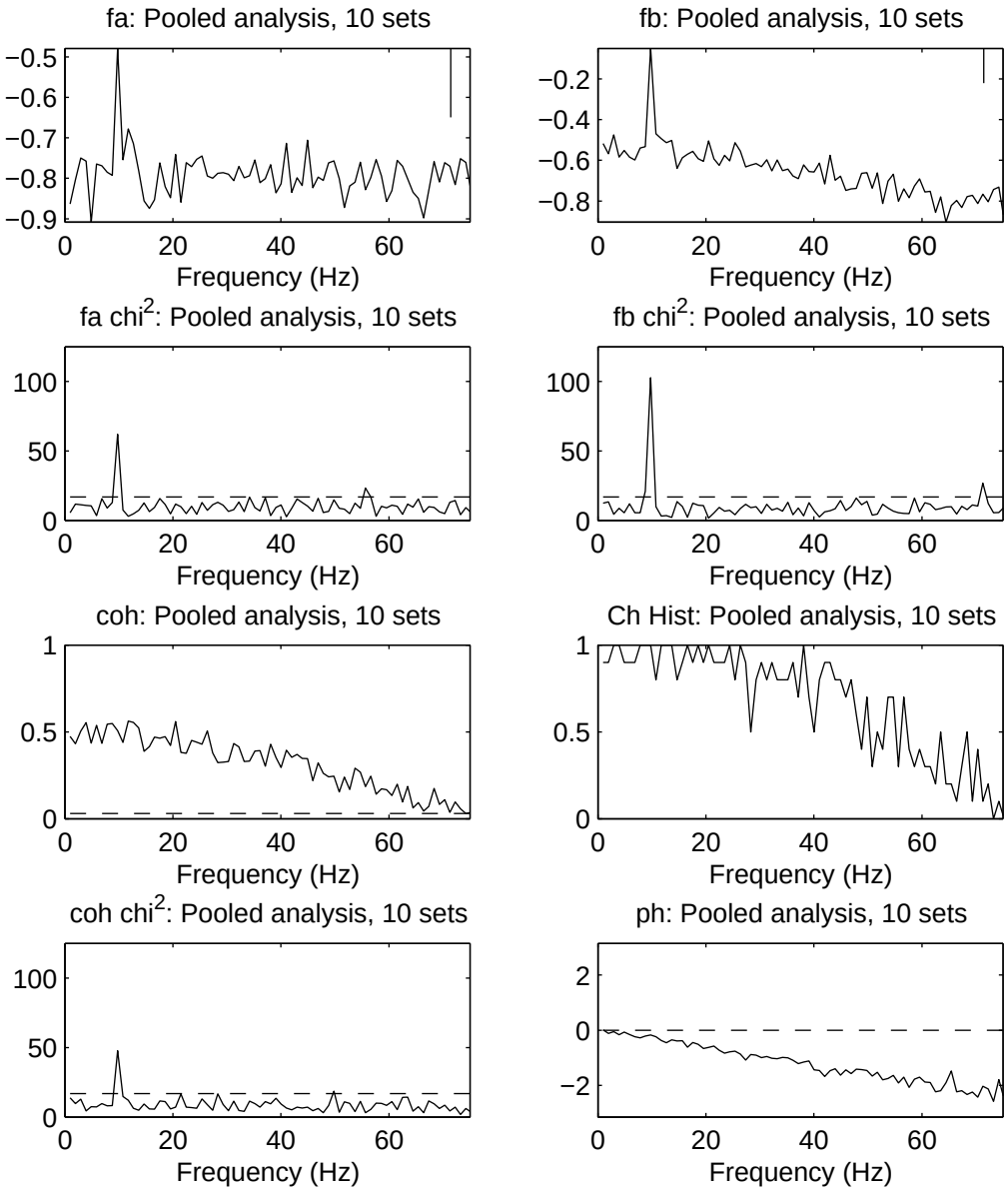

Figure 21: *Pooled analysis.* Results from same analysis as illustrated in figure 19, but plotted with `psp2_pool8_chi3`. First row illustrates two pooled spectral estimates, second row shows the two χ^2 tests for the population of input spectra (left) and output spectra (right). These tests highlight the difference at 10 Hz across the population of data sets in both channels of data. The third row shows pooled coherence (left) and the normalised histogram showing the fraction of individual coherence estimates that exhibit significant values at each frequency (right). Fourth row illustrates the χ^2 extended difference of coherence test (left), and the pooled phase estimate (right). Confidence limits as in figure 19, all three χ^2 tests have the same confidence limit (which depends on the number of records in the comparison).

12 Upgrading from NeuroSpec 1.0

NeuroSpec 1.0 implemented only Type 0 analysis (not Type 1 or Type 2, see section 5), without any of the additional extensions for comparison of spectra, comparison of coherence, and pooled analysis. The NeuroSpec 2.0 core routines include more analysis options: 3-point hanning smoothing window; Normalisation of each data segment to unit variance prior to DFT; Calculation of system identification parameters: log_{10} gain & impulse response; Tapering of data prior to DFT using split cosine taper of 5, 10, 25 or 50% at each end of segment; calculation of *Periodogram COV* test for each channel of data.

The naming of the core routines in NeuroSpec 2.0 is similar to that used in NeuroSpec 1.0. The command syntax used in NeuroSpec 2.0 is based on that used in NeuroSpec 1.0, however, the routines are not interchangeable. The three NeuroSpec 2.0 core routines (`sp2_m1`, `sp2a_m1`, `sp2a2_m1`) require the first parameter to specify the analysis type: 0, 1 or 2. The routine `sp2_m1` has an additional input parameter compared to the NeuroSpec 1.0 equivalent `sp2_m`: The 4^{th} parameter in `sp2_m1` specifies the duration of the record length in seconds.

12.1 Conversion of MATLAB scripts from NeuroSpec 1.0 to 2.0

Conversion of MATLAB scripts from NeuroSpec 1.0 to 2.0 Type 0 analysis requires the following changes

1. Change of function name to NeuroSpec 2.0 equivalent, by appending a '1' to the function name (*e.g.* `sp2a2_m` becomes `sp2a2_m1`).
2. Insert a '0' as the first input parameter, to indicate Type 0 analysis.
3. Include an additional 4^{th} parameter in `sp2_m1` specifying the record length in seconds.

12.2 Plotting

Plotting routines in NeuroSpec 2.0 retain the same names as in NeuroSpec 1.0. However, they have been modified to include additional functionality necessary for plotting the output from Type 1 and Type 2 analysis. The plotting routines in NeuroSpec 2.0 are backward compatible with NeuroSpec 1.0 analysis routine. It is recommended that you use the plotting routines supplied with NeuroSpec 2.0 for all plotting. The plotting routines from NeuroSpec 1.0 can safely be removed from the MATLAB path, or overwritten with the new routines, without any loss of functionality. No modifications are needed to calls to `psp2` (the only plotting function included in NeuroSpec 1.0) to upgrade MATLAB scripts from NeuroSpec 1.0 to 2.0.

13 Licensing

NeuroSpec is free software. It is covered by the GNU General Public Licence, a copy of which is included in the distribution file `neurospec20.zip`.

14 Further information

This user guide focuses on the practical aspects of using the NeuroSpec software. The references below can be consulted for further details of the analytical framework. Two tutorial reviews [8, 3] provide the basis for much of the analysis capabilities of NeuroSpec 2.0. Additional material and discussions can be found in [4, 5, 6, 7] . Time frequency analysis is discussed in [9]. Comparison of spectra is discussed in [2], P117. Comparison of coherence is described in [8], P7. Pooled spectra and coherence are described in [1, 5]. Details of how to download copies of these articles can be found on the NeuroSpec web site `www.neurospec.org`. Please refer to the articles below (as appropriate) in any scholarly publications resulting from the use of NeuroSpec software.

References

[1] Amjad, A.M., Halliday, D.M., Rosenberg, J.R. & Conway, B.A. (1997). An extended difference of coherence test for comparing and combining several independent coherence estimates — theory and application to the study of motor units and physiological tremor. *Journal of Neuroscience Methods* **73**, 69-79.

[2] Diggle, P.J. (1990) *Time series. A biostatistical introduction.* Clarendon Press, Oxford.

[3] Halliday, D.M., Rosenberg, J.R., Amjad, A.M., Breeze, P., Conway, B.A. & Farmer, S.F. (1995). A framework for the analysis of mixed time series/point process data - theory and application to the study of physiological tremor, single motor unit discharges and electromyograms. *Progress in Biophysics and molecular Biology* **64**, 237-278.

[4] Halliday, D.M. & Rosenberg, J.R. (1999). Time and frequency domain analysis of spike train and time series data. In: *Modern Techniques in Neuroscience Research*, (Eds. U. Windhorst & H. Johansson), Springer-Verlag, Ch 18, 503-543.

[5] Halliday, D.M. & Rosenberg, J.R. (2000). On the application, estimation and interpretation of coherence and pooled coherence. *Journal of Neuroscience Methods*, **100**, 173-174.

[6] Halliday, D.M. (2005) Spike train analysis for neural systems. In: *Modelling in the Neurosciences (2nd edition)* (Eds: Reeke, G.N. et al.), CRC Press, 555-579.

[7] Nielsen J.B., Conway B.A., Halliday D.M., Perreault M-C. & Hultborn H. (2005) Organization of common synaptic drive to motoneurones during fictive locomotion in the spinal cat. *Journal of Physiology*, **569**, 291-304.

[8] Rosenberg, J.R., Amjad, A.M., Breeze, P., Brillinger, D.R., & Halliday, D.M. (1989). The Fourier approach to the identification of functional coupling between neuronal spike trains. *Progress in Biophysics and molecular Biology* **53**, 1-31.

[9] Zhan, Y., Halliday, D.M., Liu, X. & Feng, J. (2006) Detecting time-dependent coherence between non-stationary electrophysiological signals - A combined statistical and time-frequency approach. *Journal of Neuroscience Methods*, **156**, 322-332.

15 Acknowledgements

NeuroSpec 2.0 has been written by David Halliday. Thanks are due to the many people who have contributed to the development of the framework: Jay Rosenberg, Bernie Conway, Abdul Majeed Amjad, Alex Rigas, David Murray-Smith, Joe Lau, Peter Breeze, Simon Farmer, Jens Nielsen, Yang Zhan, John-Stuart Brittain. The work has been supported by grants from the Wellcome Trust, BBSRC, EPSRC & MRC.

The URL of this document is: http://www.neurospec.org/neurospec20.pdf
The NeuroSpec home page is at: http://www.neurospec.org/
Contact: contact@neurospec.org

A Appendix A: List of files

This Appendix gives an overview of the `m` files included in NeuroSpec 2.0.

Core routines	
`sp2_m1.m`	Analysis of two spike trains.
`sp2a_m1.m`	Analysis of one spike train and one time series.
`sp2a2_m1.m`	Analysis of two time series.
`sp2_fn2a.m`	Function called by routines (Two channel weighted periodogram analysis).
Comparison of spectra and comparison of coherence	
`sp2_compf.m`	Comparison of spectra test.
`sp2_compcoh.m`	Comparison of coherence test.
Pooled spectra and coherence	
`pool_scf.m`	Create or update pooled spectral coefficients.
`pool_scf_out.m`	Convert pooled spectral coefficients to form suitable for plotting.
Main Plotting Routines	
`psp2.m`	Plots Type 0, Type 1 or Type 2 (single offset) analysis from core routines.
`psp2s.m`	As `psp2`, includes optional system identification parameters.
`psp2_cov.m`	As `psp2`, includes optional Periodogram COV test for each channel.
`psp2_tf.m`	Plotting of Type 2 time-frequency analysis over range of offset values.
`psp_compf1.m`	Plots results of comparison of spectra test.
`psp_compcoh1.m`	Plots results of comparison of coherence test.
`psp2_pool6.m`	Plots results of pooled analysis.
`psp2_pool8.m`	Plots results of pooled analysis (includes 2 additional panels).
`psp2_pool8_chi3.m`	Plots results of pooled analysis, includes 3 chi-squared tests (no time domain).
`psp2_pool9_chi3.m`	Plots results of pooled analysis, includes 3 chi-squared tests (with time domain).
Other Plotting Routines (used in)	
`psp_fa1.m`	Plots input log spectral estimate (`psp2`, `psp2s`, `psp2_pool*`)
`psp_fb1.m`	Plots output log spectral estimate (`psp2`, `psp2s`, `psp2_pool*`)
`psp_fcova1.m`	Plots Periodogram COV test for input spectral estimate.
`psp_fcovb1.m`	Plots Periodogram COV test for output spectral estimate.
`psp_ch1.m`	Plots coherence estimate (`psp2`, `psp2s`, `psp2_pool*`)
`psp_ph1.m`	Plots phase estimate (`psp2`, `psp2s`, `psp2_pool*`)
`psp_q1.m`	Plots cumulant density estimate (`psp2`, `psp2s`, `psp2_pool*`)
`psp_g1.m`	Plots log gain estimate (`psp2s`)
`psp_a1.m`	Plots impulse response estimate (`psp2s`)
`psp_chi1.m`	Plots χ^2 difference of coherence test (`psp2_pool*`)
`psp_fchia1.m`	Plots χ^2 difference of input spectra test (`psp2_pool8_chi3`, `psp2_pool9_chi3`)
`psp_fchib1.m`	Plots χ^2 difference of output spectra test (`psp2_pool8_chi3`, `psp2_pool9_chi3`)
`psp_chplf1.m`	Plots alternative pooled coherence estimate (`psp2_pool8`)
`psp_chst1.m`	Plots histogram of significant coherence values in population (`psp2_pool*`)
`psptf_fa1.m`	Plots time dependent input log spectral estimate (`psp2_tf`)
`psptf_fb1.m`	Plots time dependent output log spectral estimate (`psp2_tf`)
`psptf_ch1.m`	Plots time dependent coherence estimate (`psp2_tf`)
`psptf_ph1.m`	Plots time dependent phase estimate (`psp2_tf`)
`psptf_q1.m`	Plots time dependent cumulant density estimate (`psp2_tf`)
Plotting of individual estimates with point-wise confidence limits	
`psp_ch1cl.m`	Plots coherence estimate.
`psp_ph1cl.m`	Plots phase estimate.
`psp_g1cl.m`	Plots log gain estimate.
Demonstration scripts	
`sp2_type0_demo1.m`	Demonstrates Type 0 analysis.
`sp2_type1_demo1.m`	Demonstrates Type 1 analysis.
`sp2_type2_demo1.m`	Demonstrates Type 2 analysis.
`sp2_comp_demo1.m`	Demonstrates comparison of spectra and comparison of coherence.
`sp2_pool_demo1.m`	Demonstrates pooled spectral and coherence analysis.