Source Information Flow Toolbox (SIFT)

An Electrophysiological Information Flow Toolbox for EEGLAB
Theoretical Handbook and User Manual

Tim Mullen

Swartz Center for Computational Neuroscience,
Institute for Neural Computation, and
Department of Cognitive Science
University of California, San Diego

`tim@sccn.ucsd.edu`

Release 0.1a
Dec 15, 2010

1. Table of Contents

2. Introduction

Mapping the structural and active functional properties of brain networks is a key goal of basic and clinical neuroscience and medicine. The novelty and importance of this transformative research was recently emphasized by the U.S. National Institute of Health in their 2010 announcement for the Human Connectome Project:

> *Knowledge of human brain connectivity will transform human neuroscience by providing not only a qualitatively novel class of data, but also by providing the basic framework necessary to synthesize diverse data and, ultimately, elucidate how our brains work in health, illness, youth, and old age.*

The study of human brain connectivity generally falls under one or more of three categories: *structural, functional,* and *effective* (Bullmore and Sporns, 2009).

Structural connectivity denotes networks of anatomical (e.g., axonal) links. Here the primary goal is to understand what brain structures are *capable* of influencing each other via direct or indirect axonal connections. This might be studied *in vivo* using invasive axonal labeling techniques or noninvasive MRI-based diffusion weighted imaging (DWI/DTI) methods.

Functional connectivity denotes (symmetrical) correlations in activity between brain regions during information processing. Here the primary goal is to understand what regions are functionally related through correlations in their activity, as measured by some imaging technique. A popular form of functional connectivity analysis using functional magnetic resonance imaging (fMRI) has been to compute the pairwise correlation (or partial correlation) in BOLD activity for a large number of voxels or regions of interest within the brain volume.

In contrast to the symmetric nature of functional connectivity, *effective connectivity* denotes asymmetric or causal dependencies between brain regions. Here the primary goal is to identify which brain structures in a functional network are (causally) influencing other elements of the network during some stage or form of information processing. Often the term "information flow" is used to indicate directionally specific (although not necessarily causal) effective connectivity between neuronal structures. Popular effective connectivity methods, applied to fMRI and/or electrophysiological (EEG, iEEG, MEG) imaging data, include dynamic causal modeling, structural equation modeling, transfer entropy, and Granger-causal methods.

Contemporary research on building a human 'connectome' (complete map of human brain connectivity) has typically focused on structural connectivity using MRI and diffusion-weighted imaging (DWI) and/or on functional connectivity using fMRI. However, the brain is a highly dynamic system, with networks constantly adapting and responding to environmental influences so as to best suit the needs of the individual. A complete description of the human connectome necessarily requires accurate mapping and modeling of transient directed information flow or causal dynamics within distributed anatomical networks. Efforts to examine transient dynamics of effective connectivity (causality or directed information flow) using fMRI are complicated by low temporal resolution, assumptions regarding the spatial stationarity of the hemodynamic response, and smoothing transforms introduced in standard fMRI signal processing (Deshpande et al., 2009a; Deshpande et al., 2009b). While electro- and magneto-encephalography (EEG/MEG) affords high temporal resolution, the traditional approach of estimating connectivity between EEG electrode channels (or MEG sensors) suffers from a high risk of false positives from volume conduction and non-brain artifacts. Furthermore, severe limitations in spatial resolution when using surface sensors further limits the physiological interpretability of observed connectivity. Although precisely identifying the anatomical locations of sources of observed electrical activity (the inverse problem) is mathematically ill-posed, recent improvements in source separation and localization techniques may allow approximate identification of such anatomical coordinates with sufficient accuracy to yield anatomical insight invaluable to a wide range of cognitive neuroscience and neuroengineering applications (Michel et al., 2004). In limited circumstances it is also possible to obtain human intracranially-recorded EEG (ICE, ECoG, iEEG) that, although highly invasive, affords high spatiotemporal resolution and (often) reduced susceptibility to non-brain artifacts.

Once activity in specific brain areas have been identified using source separation and localization, it is possible to look for transient changes in dependence between these different brain source processes. Advanced methods for non-invasively detecting and modeling distributed network events contained in high-density EEG data are highly desirable for basic and clinical studies of distributed brain activity supporting behavior and experience. In recent years, Granger Causality (GC) and its extensions have increasingly been used to explore 'effective' connectivity (directed information flow, or causality) in the brain based on analysis of prediction errors of autoregressive models fit to channel (or source) waveforms. GC has enjoyed substantial recent success in the neuroscience community, with over 1200 citations in the last decade (Google Scholar). This is in part due to the relative simplicity and interpretability of GC – it is a data-driven approach based on linear regressive models requiring only a few basic *a priori* assumptions regarding the generating statistics of the data. However, it is also a powerful technique for system identification and causal analysis. While many landmark studies have applied GC to invasively recorded local field potentials and spike trains, a growing number of studies have successfully applied GC to non-invasively recorded human EEG and MEG data (as reviewed in (Bressler and Seth, 2010)). Application of these methods in the EEG source domain is also being seen in an increasing number of studies (Hui and Leahy, 2006; Supp et al., 2007; Astolfi et al., 2007; Haufe et al., 2010).

In the last decade an increasing number of effective connectivity measures, closely related to Granger's definition of causality, have been proposed. Like classic GC, these measures can be derived from (multivariate) autoregressive models fit to observed data time-series. These measures can describe different aspects of network dynamics and thus comprise a complementary set of tools for effective connectivity or causal analysis.

Several toolboxes affording various forms of Granger-causal (or related) connectivity analysis are currently available in full or beta-release. Table 1 provides a list of several of these toolboxes, along with the website, release version, and license. Although these toolboxes provide a number of well-written and useful functions, they generally lack integration within a more comprehensive framework for EEG signal processing (with the exception of TSA, which does integrate into the Biosig EEG/MEG processing suite). Furthermore, most of these either implements only one or two (often bivariate) connectivity measures, lacks tools for sophisticated visualization, or lacks robust statistics or multi-subject (group) analysis. Finally, with the exception of E-Connectome, none of these toolboxes directly support analysis and/or visualization of connectivity in the EEG source domain. These are all issues that our Source Information Flow Toolbox (SIFT), combined with the EEGLAB software suite, attempts to address.

Table 1. A list of free Matlab-based toolboxes for connectivity and graph-theoretical analysis in neural data.

Toolbox Name	Primary Author	Release	Website	License
Granger Causality Connectivity Analysis (GCCA) Toolbox	Anil Seth	2.6.1	http://www.informatics.sussex.ac.uk/users/anils/index.htm	GPL 3
Time-Series Analysis (TSA) Toolbox	Alois Schloegl	3.00	http://biosig-consulting.com/matlab/tsa/	GPL 2
E-Connectome	Bin He	1.0	http://econnectome.umn.edu/	GPL 3
Brain-System for Multivariate AutoRegressive Timeseries (BSMART)	Jie Cui	beta	http://www.brain-smart.org/	--

SIFT is an open-source Matlab (The Mathworks, Inc.) toolbox for analysis and visualization of multivariate information flow and causality, primarily in EEG/iEEG/MEG datasets following source separation and localization. The toolbox supports both command-line (scripting) and graphical user interface (GUI) interaction and is integrated into the widely used open-source EEGLAB software environment for electrophysiological data analysis (sccn.ucsd.edu/eeglab). There are currently four modules: data preprocessing, model fitting and connectivity estimation, statistical analysis, and visualization. A fifth group analysis module, affording connectivity analysis and statistics over a cohort of datasets, will be included in the upcoming beta release. First methods implemented include a large number of popular frequency-domain granger-causal and coherence measures, obtained from adaptive multivariate autoregressive models, surrogate and analytic statistics, and a suite of tools for interactive visualization of information flow dynamics across time, frequency, and (standard or personal MRI co-registered) anatomical source locations.

In this paper, I will outline the theory underlying multivariate autoregressive modeling and granger-causal analysis. Practical considerations, such as data length, parameter selection, and non-stationarities are addressed throughout the text, and useful tests for estimating statistical significance are outlined. This theory section is followed by a hands-on walkthrough of the use of the SIFT toolbox for analyzing source information flow dynamics in an EEG dataset. Here we will walk through a typical data processing pipeline culminating with the demonstration of some of SIFT's powerful tools for interactive visualization of time- and frequency-dependent directed information flow between localized EEG sources in an anatomically-coregistered 3D space. Theory boxes through the chapter will provide additional insight on various aspects of model fitting and parameter selection.

3. Multivariate Autoregressive Modeling

Assume we have an *M*-dimensional time-series of length *T* (e.g., *M* channels of EEG data, with *T* time points per channel)*:* $X := x_1 \ldots x_T$ where $x_t = [x_{1t} \ldots x_{Mt}]'$. We can represent the multivariate process at time *t* as a stationary, stable vector autoregressive (VAR, MVAR, MAR) process of order *p* (Henceforth we will denote this as a VAR[*p*] process):

$$x_t = v + \sum_{k=1}^{p} A_k x_{t-k} + u_t \qquad \text{(Eq 3.1)}$$

Here $v = [v_1 \ldots v_M]'$ is an (*M* x 1) vector of intercept terms (the mean of *X*), A_i are (*M* x *M*) model coefficient matrices and u_t is a zero-mean white noise process with nonsingular covariance matrix Σ.

3.1. Stationarity and Stability

We assume two basic conditions regarding the data *X* and its associated VAR[*p*] model: *stationarity* and *stability*. A stochastic process *X* is *weakly stationary* (or *wide-sense stationary* (WSS)) if its first and second moments (mean and covariance) do not change with time. In other words $E(x_t) = \mu$ for all *t* and $E[(x_t - \mu)(x_{t-h} - \mu)'] = \Gamma(h) = \Gamma(-h)'$ for all *t* and *h*=0,1,2, ... where *E* denotes expected value. A VAR[*p*] process is considered *stable* if its reverse characteristic polynomial has no roots in or on the complex unit circle. Formally, x_t is stable if

$$\det(I_{Mp} - Az) \neq 0 \text{ for } |z| \leq 1 \text{ where}$$

$$\mathbf{A} := \underset{(Mp \times Mp)}{\begin{bmatrix} A_1 & A_2 & \cdots & A_p \\ I_M & 0 & \cdots & 0 \\ \vdots & \ddots & & \vdots \\ 0 & \cdots & I_M & 0 \end{bmatrix}}$$

Equivalently, x_t is stable if all eigenvalues of **A** have modulus less than 1 (Lütkepohl, 2006). A stable process is one that will not diverge to infinity ("blow up"). An important fact is that stability implies stationarity – thus it is sufficient to test for stability to ensure that a VAR[*p*] process is both stable and stationary. SIFT performs a stability test by analyzing the eigenvalues of **A**.

3.2. The Multivariate Least-Squares Estimator

A parametric VAR model can be fit using a number of approaches including multivariate least-squares approaches (e.g., MLS, ARFIT), lattice algorithms (e.g., Vieira-Morf), or state-space models (e.g., Kalman filtering). Here we will briefly outline the multivariate least-squares algorithm (multichannel Yule-Walker) and encourage the interested reader to consult (Schlögl, 2000; Lütkepohl, 2006; Schlögl, 2006) for more details on this and other algorithms (several of which are implemented in SIFT).

To derive the multivariate least-squares estimator, let us begin with some definitions:

$$\begin{aligned}
&X := (x_1, \ldots, x_T) && (M \times T),\\
&B := (\nu, A_1, \ldots, A_p) && (M \times (Mp+1)),\\
&Z_t := \begin{bmatrix} 1 \\ x_t \\ \vdots \\ x_{t-p+1} \end{bmatrix} && ((Mp+1) \times 1),\\
&Z := (Z_0, \ldots, Z_{T-1}) && ((Mp+1) \times T),\\
&U := (u_1, \ldots, u_T) && (M \times T)
\end{aligned}$$

Our VAR[*p*] model (Eq 3.1) can now be written in compact form:

$$X = BZ + U \qquad \text{(Eq 3.2)}$$

Here *B* and *U* are unknown. The multivariate (generalized) least-squares (LS, GLS) estimator of *B* is the estimator $\hat{B}$ that minimizes the variance of the innovation process (residuals) *U*. Namely,

$$\hat{B} = \arg\min_B S(B)$$

where $S(B) = \text{tr}[(X - BZ)'\Sigma^{-1}(X - BZ)]$.

It can be shown (Lütkepohl, 2006) that the LS estimator can be obtained by

$$\hat{B} = XZ'(ZZ')^{-1} \qquad \text{(Eq 3.3)}$$

This result can be derived in several ways, however a simple approach follows from post-multiplying

$$x_t = BZ_{t-1} + u_t$$

by Z'_{t-1} and taking expectations:

$$E(x_t Z'_{t-1}) = BE(Z_{t-1} Z'_{t-1}) \qquad \text{(Eq 3.4)}$$

Estimating $E(x_t Z'_{t-1})$ by

$$\frac{1}{T}\sum_{t=1}^{T} x_t Z'_{t-1} = \frac{1}{T} XZ'$$

we obtain the normal equations

$$\frac{1}{T} XZ' = \hat{B}\frac{1}{T} ZZ'$$

and thus, $\hat{B} = XZ'(ZZ')^{-1}$.

The reader may note that $\hat{B}$ is simply the product of X and the Moore-Penrose pseudoinverse of Z: $\hat{B} = XZ^{\dagger}$ where $Z^{\dagger} = \mathrm{pinv}(Z)$. The reader familiar with univariate autoregressive model fitting might also note that (Eq 3.4) is very similar to the well-known system of Yule-Walker equations. Hence, this can be considered an extension to the multivariate case of the Yule-Walker algorithm for univariate AR model fitting.

Although asymptotically optimal, the LS algorithm often suffers from sub-optimal performance when even moderate sample sizes are available, as compared to more robust modified LS algorithms (e.g., the stepwise least-squares ARFIT algorithm) or non-LS algorithms (e.g., the Vieira-Morf lattice algorithm). A detailed empirical performance comparison of these and other algorithms can be found in (Schlögl, 2006). For this reason, SIFT abandons the LS algorithm in favor of these more robust algorithms. The SIFT functions **`pop_est_fitMVAR()`** and **`est_fitMVARKalman()`** provide access to various model-fitting approaches.

3.3. Frequency-Domain Representation

Electrophysiological processes generally exhibit oscillatory structure, making them well suited for frequency-domain analysis (Buzsaki, 2006). A suitably fit autoregressive model provides an idealized model for the analysis of oscillatory structure in stochastic time series (Burg, 1967; Zetterberg, 1969; Burg, 1975; Neumaier and Schneider, 2001). From the AR coefficients, we can obtain a number of useful quantities including the *spectral density matrix* and the *transfer function* of the process. From these and related quantities we can obtain power spectra, coherence and partial coherence, Granger-Geweke causality, directed transfer function, partial directed coherence, phase-locking value, and a number of other quantities increasingly being used by the neuroscience community to study synchronization and information flow in the brain (Pereda et al., 2005; Schelter et al., 2006).

To obtain our frequency-domain representation of the model, we begin with our VAR[*p*] model from (Eq 3.1). For simplicity, we will assume the process mean is zero:

$$x_t = \sum_{k=1}^{p} A_k x_{t-k} + u_t$$

Rearranging terms we get

$$u_t = \sum_{k=0}^{p} \hat{A}_k x_{t-k} \text{ where } \hat{A}_k = -A_k \text{ and } \hat{A}_0 = -I$$

Z-transforming both sides yields:

$U(f) = A(f)X(f)$ where

$$A(f) = \sum_{k=0}^{p} \hat{A}_k e^{-i2\pi fk}$$

Premultiplying by $A(f)^{-1}$ and rearranging terms we obtain:

$$X(f) = A(f)^{-1}U(f) = H(f)U(f)$$

Here *X(f)* is the (*M* x *M*) spectral matrix of the multivariate process, *U(f)* is a matrix of random sinusoidal shocks and *A(f)*$^{-1}$ = *H(f)* is the *transfer matrix* of the system. Note that *H(f)* transforms the noise input (*U*) into the structured spectral matrix. This should give us a hint that analysis of *H(f)* (and *A(f)*) will help us in identifying the structure of the modeled system (including information flow dynamics). The spectral density matrix of the process (which contains the auto-spectrum of each variable (at frequency *f*) on the diagonals and the cross-spectrum on the off-diagonals) is given by:

$$S(f) = X(f)X(f)^* = H(f)\Sigma H(f)^*$$

As we shall see in Section 4.3. , from $S(f), A(f), H(f)$ and Σ, we can derive a number of frequency-domain quantities relevant to the study of oscillations, information flow, and coupling in neural systems.

3.4. Modeling non-stationary data using adaptive VAR models

In section 3. we stated that data stationarity is a necessary precondition for accurate VAR estimation. However, it is well-known that neural data, including EEG and Local Field Potentials (LFPs), can be highly non-stationary, exhibiting large fluctuations in both the mean and variance over time. For instance, a record of EEG data containing evoked potentials (EPs) is a classic example of a non-stationary time series (both the mean and variance of the series changes dramatically and transiently during the evoked response). Another example would be EEG data collected during slow-wave sleep, which exhibits slow fluctuations in the mean EEG voltage over time. A number of algorithms have been proposed for fitting VAR models to non-stationary series. In the neuroscience community the most popular approaches include segmentation (overlapping sliding-window) approaches (Jansen et al., 1981; Florian and Pfurtscheller, 1995; Ding et al., 2000), state-space (Kalman filtering) approaches (Schlögl, 2000; Sommerlade et al., 2009), and non-parametric methods based on minimum-phase spectral matrix factorization (Dhamala et al., 2008). All of these approaches are currently – or soon to be made – accessible in SIFT. Here we will briefly outline the concepts behind each modeling approach.

3.4.1. Segmentation-based Adaptive VAR (AMVAR) models

A segmentation-based AMVAR adopts an approach rather similar to the concept behind short-time fourier transforms or other windowing techniques. Namely, we extract a sliding window of length *W* from the multivariate dataset, and fit our VAR[*p*] model to this data. We then increment the window by a (small) quantity *Q* and repeat the procedure until the start of the window is greater than *T-W*. This produces *floor((T-W)/Q+1)* VAR coefficient matrices which describe the evolution of the VAR[*p*] across time. The concept here is that by using a sufficiently small window, the data will be *locally stationary* within the window and suitable for VAR modeling. By using highly overlapping windows (small *Q*) we can obtain coefficients that change relatively smoothly with time. Figure 1 shows a schematic of the sliding-window AMVAR approach.

One concern here is whether sufficient data points are available to accurately fit the model. In the general case, we have M^2p coefficients (free parameters) to estimate, which requires a minimum of M^2p data samples. However, in practice, we would like to have at least 10 times as many data points as free parameters (Schlögl and Supp, 2006; Korzeniewska et al., 2008). When multiple realizations (e.g., experimental trials) are available, we can assume that each trial is a random sample from the same stochastic process and average covariance matrices across trials to reduce the bias of our model coefficient estimator (Ding et al., 2000). For the LS algorithm, explained in section 3. , this yields the modified estimator:

$$\hat{B} = E(X^{(i)}Z'^{(i)})E(Z^{(i)}Z'^{(i)})^{-1} \qquad \text{(Eq 3.5)}$$

Where $X^{(i)}$ and $Z^{(i)}$ denote matrices *X* and *Z* for the *i*th single-trial and the expected value is taken across all trials. This approach effectively increases the number of samples available for a sliding window of length *W* from *W* to *WN,* where *N* is the number of trials/realizations. This allows us to potentially use very small windows (containing as few as *p+1* sample points) while still obtaining a good model fit.

When using short windows with multi-trial data, an important preprocessing step is to pointwise subtract the ensemble mean and divide by the ensemble standard deviation (ensemble normalization). This ensures that the ensemble mean is zero and the variance is one, at every time point. This can dramatically improve the local stationarity of the data (Ding et al., 2000). An important result of this is that we are essentially modeling dependencies in the residual time-series after removing the event-related potential (ERP) from the data. The fact that this preprocessing step has become common practice in published applications of AMVAR analysis to neural data suggests that there is, in fact, rich task-relevant information present in the so-called “residual noise” component of the EEG which cannot be inferred from the ERP itself (Ding et al., 2000; Bressler and Seth, 2010). This fits under the model that mean-field electrophysiological measures such as LFPs and EEG measure a sum of (potentially oscillatory) ongoing activity and evoked responses where the amplitude and phase of the evoked response depends largely on the phase of the ongoing oscillations (Kenet et al., 2005; Wang et al., 2008). Analyzing the phase structure of the stationary ongoing oscillations may provide a deeper insight into the state of the underlying neural system than the analysis of the evoked responses themselves.

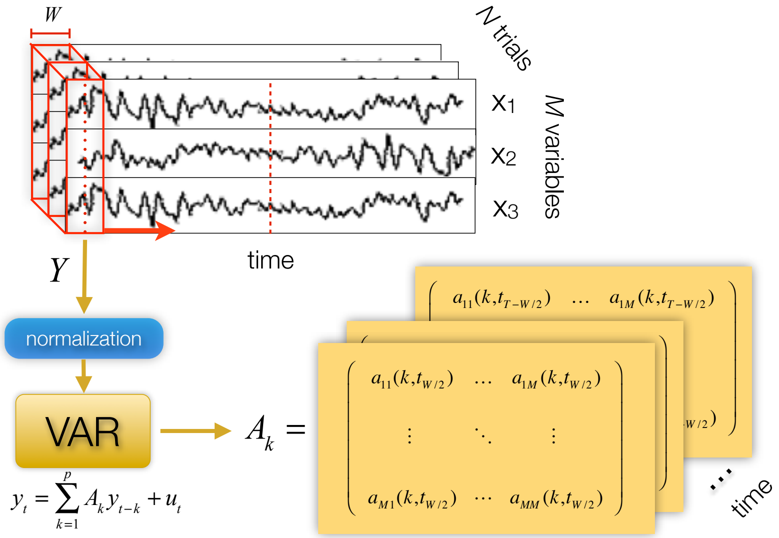

W
N trials
X1
X2
X3
M variables
time
Y
normalization
VAR
$A_k =$
$a_{11}(k,t_{T-W/2})$ … $a_{1M}(k,t_{T-W/2})$
$a_{11}(k,t_{W/2})$ … $a_{1M}(k,t_{W/2})$
⋮ ⋱ ⋮
$a_{M1}(k,t_{W/2})$ … $a_{MM}(k,t_{W/2})$
$y_t = \sum_{k=1}^{p} A_k y_{t-k} + u_t$
…
time

Figure 1. Schematic of sliding-window AMVAR modeling. W is the window length, T is the length of each trial in samples, N is the number of trials.

3.5. Model order selection

Parametric VAR model fitting really involves only one parameter: the model order. The most common approach for model order selection involves selecting a model order that minimizes one or more *information criteria* evaluated over a range of model orders. Commonly used information criteria include, Akaike Information Criterion (AIC), Schwarz-Bayes Criterion (SBC) – also known as the Bayesian Information Criterion (BIC) – Akaike's Final Prediction Error Criterion (FPE), and Hannan-Quinn Criterion (HQ). A detailed comparison of these criteria can be found in Chapter 4.3 of (Lütkepohl, 2006). In brief, each criterion is a sum of two terms, one that characterizes the entropy rate or prediction error of the model, and a second term that characterizes the number of freely estimated parameters in the model (which increases with increasing model order). By minimizing both terms, we seek to identify a model that is both parsimonious (does not overfit the data with too many parameters) while also accurately modeling the data. The criteria implemented in SIFT are defined in Table 2.

Table 2. Information criteria for model order selection implemented in SIFT. Here $\hat{T} = TN$ is the total number of samples (data points) used to fit the model

Estimator	Formula
Schwarz-Bayes Criterion (Bayesian Information Criterion)	$SBC(p) = \ln\left\|\tilde{\Sigma}(p)\right\| + \frac{\ln(\hat{T})}{\hat{T}} pM^2$
Akaike Information Criterion	$AIC(p) = \ln\left\|\tilde{\Sigma}(p)\right\| + \frac{2}{\hat{T}} pM^2$

Akaike's Final Prediction Error	$FPE(p) = \left\|\tilde{\Sigma}(p)\right\| + \left(\frac{\hat{T}+Mp+1}{\hat{T}-Mp-1}\right)^{M}$ and its logarithm (used in SIFT) $\ln(FPE(p)) = \ln\left\|\tilde{\Sigma}(p)\right\| + M\ln\left(\frac{\hat{T}+Mp+1}{\hat{T}-Mp-1}\right)$
Hannan-Quinn Criterion	$HQ(p) = \ln\left\|\tilde{\Sigma}(p)\right\| + \frac{2\ln(\ln(\hat{T}))}{\hat{T}}pM^{2}$

For a given information criterion, *IC*, we select the model order that minimizes *IC*:

$$p_{sel} = \underset{p}{\arg\min}\, IC(p)$$

Here, the first term, $\ln\left|\tilde{\Sigma}(p)\right|$ is the logarithm of the determinant of the estimated noise covariance matrix (prediction error) for a VAR model of order *p* fit to the *M*-channel data, where *TN* is the total number of datapoints used to fit the model (*T* samples per trial x *N* trials). The key difference between the criteria is how severely each penalizes increases in model order (the second term). AIC and SBC are the most widely used criteria, but SBC more heavily penalizes larger model orders. For moderate and large *TN,* FPE and AIC are essentially equivalent (see Lutkepohl (2006) p. 148 for a proof); however, FPE may outperform AIC for very small sample sizes. HQ penalizes high model orders more heavily than AIC but less than SBC. Both SBC and HQ are *consistent* estimators, which means that $\lim_{N\to\infty}\Pr\{p_{sel} = p_{true}\} = 1$. This cannot be said of AIC and FPE. However, under small sample conditions (small *N*), AIC/FPE may outperform SBC and/or HQ in selecting the true model order (Lütkepohl, 2006). When modeling EEG data, it is common for AIC and FPE to show no clear minimum over a reasonable range of model orders. In this case, there may be a clear "elbow" in the criterion plotted as a function of increasing model order, which may suggest a suitable model order.

When selecting a model order for neural connectivity analysis, it is important to consider the dynamics of the underlying physiological system. In particular, one should consider the maximum expected time lag between any two variables included the model. If we have reason to expect a time lag of τ seconds between any two brain processes, we should make sure to select a model order of $p \geq \tau F_s$ where F_s is the process sampling rate in Hz. Additionally, we should consider that the multivariate spectrum of a *M*-dimensional VAR[*p*] model has *Mp*/2 frequency components (peaks) distributed amongst the *M* variables (there are *Mp* complex-conjugate roots of the characteristic equation of the model). This means that we can observe *p*/2 frequency peaks between each pair of variables (Florian and Pfurtscheller, 1995; Schlögl and Supp, 2006). Thus a reasonable lower bound on the model order might be twice the number of expected frequencies plus one (for the zero-Hz peak). Tests performed by Jansen (1981) and Florian and Pfurtscheller (1995) demonstrated that a potentially optimal model order for modeling EEG spectra was p=10, although little spectral differences were identified for model orders between 9 and 13. A key point, however, is that this was identified for a sampling rate of 128 Hz and it is known that the optimal model order depends significantly on the sampling rate of the process (Zetterberg, 1969).

The principle motivation behind heavy penalization of high model orders in an information criterion is to improve forecasting performance by reducing over-fitting. However, forecasting is not necessarily the ultimate goal of our neural modeling approach. Furthermore, selecting a too-small model order can severely impair our frequency resolution (merging peaks together) as well as our ability to detect coupling over long time lags. Where there is a question as to a suitable model order, it is often better to err on the side of selecting a larger model order. As such, a criterion such as HQ, which often shows a clear minimum but affords intermediate penalization between AIC and SBC may represent an optimal choice for neural data.

In general, it is good practice to select a model order by examining multiple information criteria and combining this information with additional expectations and knowledge specific to the physiological properties of the neural system being analyzed. When possible spectra and coherence obtained from fitted VAR models should be compared with those obtained from non-parametric methods (such as wavelets) to validate the model. Model order selection is often an iterative process wherein, through model validation, we determine the quality of our model fit, and, if necessary, revise our model specification until the data is adequately modeled.

Model order selection is implemented in SIFT using **`pop_est_selModelOrder()`**.

3.6. Model Validation

There a number of criteria which we can use to determine whether we have appropriately fit our VAR model. SIFT implements three commonly used categories of tests: (1) checking the residuals of the model for serial and cross-correlation (whiteness tests), (2) testing the consistency of the model, and (3) check the stability/stationarity of the model. These can be accessed through the SIFT GUI using **`pop_est_validateMVAR()`**

3.6.1. Checking the whiteness of the residuals

Recall the compact model definition from (Eq 3.2): $X = BZ + U$. Here we can regard the VAR[*p*] model coefficients *B* as a filter which transforms innovations (random white noise), *U*, into observed, structured data *X*. Consequently, for coefficient estimates $\hat{B}$, we can obtain the residuals $\hat{U} = X - \hat{B}Z$. If we have adequately modeled the data, the residuals should be small and uncorrelated (white). Correlation structure in the residuals means there is still some correlation structure in the data that has not been described by our model. Checking the whiteness of residuals typically involves testing whether the residual autocorrelation coefficients up to some desired lag *h* are sufficiently small to ensure that we cannot reject the null hypothesis of white residuals at some desired significance level.

3.6.1.1. Autocorrelation Function (ACF) Test

The (M x M) lag l autocovariance matrix of the residuals is given by $C_l = E[\hat{u}_t \hat{u}'_{t-l}]$. We denote the autocovariances up to lag l as $\mathbf{C}_h = (C_1, \ldots, C_h)$. The lag l autocorrelation matrix is given by $R_l = D^{-1} C_l D^{-1}$ where D is a (M x M) diagonal matrix, the diagonal elements being the square root of the diagonal elements of C_0. We are generally interested in testing the (white noise) null hypothesis $H_0 : \mathbf{R}_h = (R_1, \ldots, R_h) = 0$ against the alternative $H_1 : \mathbf{R}_h \neq 0$. A simple test, based on asymptotic properties of univariate white noise processes, involves rejecting the hypothesis that $\hat{U}$ is white noise at the 5% level if $|R_l| > \pm 2 / \sqrt{\hat{T}}$ for any lag l (excluding the diagonal elements of R_0 which are always 1). $\hat{T} = TN$ is the total number of samples used in estimating the covariance. However, since this is a pointwise significance test at the 5% level, in practice we expect one in twenty coefficients to exceed $2/\sqrt{\hat{T}}$ in absolute value even if $\hat{U}$ is white. A reasonable corrected statistic is thus the probability of a coefficient exceeding the 5% significance bounds:

$$\rho = \frac{\operatorname{count}\left(|\mathbf{R}_h| > \pm 2 / \sqrt{\hat{T}}\right)}{\operatorname{count}(\mathbf{R}_h)} = \frac{\operatorname{count}\left(|\mathbf{R}_h| > \pm 2 / \sqrt{\hat{T}}\right)}{M^2(h+1) - M}$$

If $\rho < 0.05$, or equivalently $1\text{-}\rho > 0.95$, then we cannot reject the null hypothesis at the 5% level and we accept that the residuals are white.

Due to its simplicity, this sort of test enjoys much popularity. However, it is important to bear in mind that the 5% confidence intervals apply to individual coefficients (i.e., for univariate models) and although the R_i and R_j are asymptotically uncorrelated for $i \neq j$ this is not necessarily true for the elements of R_i. As such, this test may be misleading when considering the coefficients of a multivariate model as a group. Additionally, in small sample conditions (small $\hat{T}$), this test may be overly conservative such that the null hypothesis is rejected (residuals indicated as non-white) less often than indicated by the chosen significance level (Lutkepohl, 2006).

3.6.1.2. Portmanteau Tests

In the previous section, we noted that the simple asymptotic ACF test may yield misleading results when the coefficients are considered independently rather than as a group, derived from a multivariate process. In contrast, *portmanteau* tests are a powerful class of test statistics explicitly derived to test H_0 up to some lag h. SIFT implements three portmanteau test statistics: *Box-Pierce (BPP)*, *Ljung-Box (LBP)*, and *Li-McLeod (LMP)*. Under the null hypothesis, for large sample size and h, each of these test statistics approximately follow a χ^2-distribution with $M^2(h-p)$ degrees of freedom. A ρ-value can thus be obtained by comparing the test statistic with the c.d.f. of this distribution. If $1\text{-}\rho$ is greater than some value α (e.g., 0.05 for a 5% significance level), we cannot reject the null hypothesis and we accept that the residuals are white. Table 3 lists the three tests implemented in SIFT along with their test statistics and practical notes.

Table 3. Popular portmanteau tests for whiteness of residuals, implemented in SIFT. Here $\hat{T} = TN$ is the total number of samples used to estimate the covariance

Portmanteau Test	Formula (Test Statistic)	Notes
Box-Pierce (BPP)	$Q_h := \hat{T}\sum_{l=1}^{h}\mathrm{tr}\left(C_l' C_0^{-1} C_l C_0^{-1}\right)$	The original portmanteau test. Potentially overly-conservative. Poor small-sample properties.
Ljung-Box (LBP)	$Q_h := \hat{T}(\hat{T}+2)\sum_{l=1}^{h}(\hat{T}-l)^{-1}\,\mathrm{tr}\left(C_l' C_0^{-1} C_l C_0^{-1}\right)$	Modification of BPP to improve small-sample properties. Potentially inflates the variance of the test statistic. Slightly less conservative than LMP with slightly higher (but nearly identical) statistical power.
Li-McLeod (LMP)	$Q_h := \hat{T}\sum_{l=1}^{h}\mathrm{tr}\left(C_l' C_0^{-1} C_l C_0^{-1}\right) + \frac{M^2 h(h+1)}{2\hat{T}}$	Further modification of BPP to improve small-sample properties without variance inflation. Slightly more conservative than LBP. Probably the best choice in most conditions.

BPP is the classical portmanteau test statistic. It can be shown that in small sample conditions (small $\hat{T}$) its distribution under the null hypothesis diverges from the asymptotic χ^2 distribution. This can render it overly-conservative leading us to reject the null hypothesis of white residuals even when the model was appropriately fit.

The LBP statistic attempts to improve the small-sample properties of the test statistic. By adjusting each covariance coefficient by its asymptotic variance, it can be shown that under the null hypothesis, the LBP statistic has a small-sample distribution much closer to the asymptotic distribution than the BPP statistic. However, it can also be shown that the variance of the LBP statistic can be inflated to substantially larger than its asymptotic distribution.

Like LBP, the LMP statistic has better small-sample properties than BPP. However, unlike LBP, it does so without inflating its variance. Although less popular than LBP, it has been demonstrated that the variance of LMP is closer to its asymptotic variance whereas LBP is more sensitive with significance levels somewhat larger than expected when $\hat{T}$ is large. LMP is slightly conservative but the statistical power for LMP and LBP are nearly identical. Since, in practice, it is preferable to select the more conservative test among tests with comparable power, LMP may represent an ideal choice of test statistic for most applications.

The interested reader should consult (Lutkepohl, 2006) and (Arranz, n.d.)for additional details and references concerning checking the whiteness of residuals. The whiteness of residuals can be tested in SIFT using **`est_checkMVARWhiteness()`**

3.6.2. Checking the consistency of the model

To address the question of what fraction of the correlation structure of the original data is captured by our model, we can calculate the *percent consistency* (Ding et al., 2000). We generate an ensemble, of equal dimensions as the original data, using simulated data from the VAR model. For both the real and simulated datasets, we then calculate all auto- and cross-correlations between all variables, up to some predetermined lag. Letting $\mathbf{R_r}$ and $\mathbf{R_s}$ denote the vectorized correlation matrices of the real and simulated data, respectively, the percent consistency index is given by

$$\mathrm{PC} = \left(1 - \frac{\|\mathbf{R}_s - \mathbf{R}_r\|}{\|\mathbf{R}_r\|}\right) \times 100 \quad \text{where } \|\cdot\| \text{ denotes the Euclidean } (\mathrm{L}_2) \text{ norm.}$$

A PC value near 100% would indicate that the model is able to generate data that has a nearly identical correlation structure as the original data. A PC value near 0% indicates a complete failure to model the data. While determining precisely what constitutes a sufficiently large PC value is an area for future research, a rule of thumb is that is a value of PC > 85% suggests the model is adequately capturing the correlation structure of the original data. The percent consistency can be calculated in SIFT using **`est_checkMVARConsistency()`**.

3.6.3. Checking the stability and stationarity of the model

In section 3.1. we provided a condition for the stability of a VAR[p] process. Namely, an M-dimensional VAR[p] process is stable if all the eigenvalues of the (Mp x Mp) augmented coefficient matrix **A** have modulus less than 1. Thus, a useful stability index is the log of the largest eigenvalue λ_{max} of **A**:

$$\mathrm{SI} = \ln|\lambda_{\max}|$$

A VAR[p] process is stable if and only if SI < 0. The magnitude of the SI can be loosely interpreted as an estimate of the degree to which the process is stable. As mentioned in section 3.1. , a stable process is a stationary process. Thus it is sufficient to test for stability of the model to guarantee that the model is also stationary. If the model is not stable, additional tests such as the Augmented Dickey-Fuller test may be used to separately evaluate the stationarity of the data. However, since we are generally interested modeling *stable* processes, these additional stationarity tests are not implemented in SIFT. The stability index of a fitted model can be calculated in SIFT using **`est_checkMVARStability()`**.

3.6.4. Comparing parametric and nonparametric spectra and coherence

Another approach sometimes used to validate a fitted VAR model is to compare the spectra and/or pairwise coherence estimated from the parametric models with those derived from a robust nonparametric approach such as multitapers or wavelets. Using an equation similar to percent consistency, we can estimate the fraction of the nonparametric spectrum or coherence that is captured by our VAR model. Of course, here we assuming the nonparametric spectra are optimal estimates of the true spectra ("ground truth"), which may not be the case (interestingly, Burg (1967; 1975) demonstrated that, if the data is generated by an AR process and the true model order is known, AR spectral estimation is a maximum-entropy method which means that it represents an optimal spectral estimator). Nevertheless, if the nonparametric quantities are carefully computed, this can be a useful validation procedure. An upcoming release of SIFT will include routines for computing this spectral consistency index.

4. Granger Causality and Extensions

Granger causality (GC) is a method for inferring certain types of causal dependency between stochastic variables based on reduction of prediction error of a putative effect when past observations of a putative cause are used to predict the effect, in addition to past observations of the putative effect. The concept was first introduced by Norbert Wiener in 1956 and later reformulated and formalized by C.W. Granger in the context of bivariate linear stochastic autoregressive models (Weiner, 1956; Granger, 1969). The concept relies on two assumptions:

Granger Causality Axioms

1. **Causes must precede their effects in time**
2. **Information in a cause's past must improve the prediction of the effect above and beyond information contained in the collective past of all other measured variables (including the effect).**

Assumption (1) is intuitive from basic thermodynamical principles: the arrow of causation points in the same direction as the arrow of time – the past influences the future, but not the reverse. Assumption (2) is also intuitive: for a putative cause to truly be causal, removal of the cause should result in some change in the future of the putative effect – there should be some shared information between the past of the cause and the future of the effect which cannot be accounted for by knowledge of the past of the effect.

The theory and application of GC (and its extensions) to neural system identification has been elaborated in a number of other articles and texts (Kaminski, 1997; Eichler, 2006; Blinowska and Kaminski, 2006; Ding et al., 2006; Schlögl and Supp, 2006; Bressler and Seth, 2010). As such, here we will only briefly introduce the theory and focus primarily on multivariate extensions of the granger-causal concept, including the partial directed coherence (PDC) and direct directed transfer function (dDTF).

4.1. Time-Domain GC

Suppose we wish to test whether a measured EEG variable j Granger-causes another variable i conditioned on all other variables in the measured set. Let V represent the set of all measured variables (e.g., all available EEG sources/channels): V = {1, 2, … , M}. Our complete (zero-mean) VAR[p] model is specified as:

$$x_t^{(V)} = \sum_{k=1}^{p} A_k x_{t-k}^{(V)} + u_t$$

We fit the full model and obtain the mean-square prediction error when $x^{(i)}$ is predicted from past values of $x^{(V)}$ up to the specified model order:

$\mathrm{var}(x_t^{(i)} \mid x_{(\cdot)}^{(V)}) = \mathrm{var}(u_t^{(i)}) = \Sigma_{ii}$ where $x_{(\cdot)}^{(V)} = \{x_{t-k}^{(V)}, k \in \{1,\ldots,p\}\}$ denotes the past of $x^{(V)}$

Now, suppose we exclude j from the set of variables (denoted $V\backslash j$) and re-fit the model

$$x_t^{(V\backslash j)} = \sum_{k=1}^{p} \bar{A}_k x_{t-k}^{(V\backslash j)} + \bar{u}_t$$

and again obtain the mean-square prediction error for $x^{(i)}$.

$$\mathrm{var}(x_t^{(i)} \mid x_{(\cdot)}^{(V\backslash j)}) = \mathrm{var}(\bar{u}_t^{(i)}) = \bar{\Sigma}_{ii}$$

In general, $\Sigma_{ii} \geq \bar{\Sigma}_{ii}$ and $\Sigma_{ii} = \bar{\Sigma}_{ii}$ if and only if the best linear predictor of $x^{(i)}$ based on the full past $x^{(V)}$ does not depend on $x^{(j)}$. This leads us to the following definition for multivariate GC (Eichler, 2006):

DEFINITION 1

Let *I* and *J* be two disjoint subsets of *V*. Then $x^{(J)}$ Granger-causes $x^{(I)}$ conditioned on $x^{(V)}$ if and only if the following two equivalent conditions hold:

1. $\Sigma_{ii} \gg \bar{\Sigma}_{ii}$
2. $A_{k,ij} \gg 0$ **for some** $k \in \{1,\ldots,p\}$

Here ≫ means "significantly greater than." In other words, inferring conditional GC relationships in the time domain amounts to identifying non-zero elements of a VAR[*p*] coefficient matrix fit to all available variables.

Granger (1969) quantified DEFINITION 1 for strictly bivariate processes in the form of an F-ratio:

$$F_{ij} = \ln\left(\frac{\bar{\Sigma}_{ii}}{\Sigma_{ii}}\right) = \ln\left(\frac{\operatorname{var}(x_t^{(i)} \mid x_{(\cdot)}^{(i)})}{\operatorname{var}(x_t^{(i)} \mid x_{(\cdot)}^{(i)}, x_{(\cdot)}^{(j)})}\right) \qquad \text{(Eq 4.1)}$$

Here, F_{ij} denotes the GC from process *j* to process *i*. This quantity is always non-negative and increases away from zero proportionate to the degree to which the past of process *j* conditionally explains ("granger-causes") the future of process *i*.

4.2. Frequency-Domain GC

In the frequency domain a very similar definition holds for GC as in the time domain. If we obtain the Fourier-transform of our VAR[*p*] coefficient matrices *A(f)* as in section 3.3. , based on the time-domain definition of GC we can derive the following definition for GC in the frequency-domain (Eichler, 2006):

DEFINITION 2

Let *I* and *J* be two disjoint subsets of *V*. Then $x^{(J)}$ Granger-causes $x^{(I)}$ conditioned on $x^{(V)}$ if and only if the following condition holds:

$A_{ij}(f) \gg 0$ **for some frequency *f***

DEFINITION 2 suggests a simple method for testing multivariate (conditional) GC at a given frequency *f*: we simply test for non-zero coefficients of |*A(f)*|. This approach yields a class of GC estimators known as Partial Directed Coherence (PDC) measures (Baccalá and Sameshima, 2001).

A slightly different approach, due to Granger (1969) and later refined by Geweke (1982), provides an elegant interpretation of frequency-domain GC as a decomposition of the total spectral interdependence between two series (based on the bivariate spectral density matrix, and directly related to the coherence) into a sum of "instantaneous", "feedforward" and "feedback" causality terms. However, this interpretation was originally derived only for bivariate processes and, while this has been recently been extended to trivariate (and block-trivariate) processes (Chen et al., 2006; Wang et al., 2007), it has not yet been extended to the true multivariate case. An implementation of the Granger-Geweke formulation for bivariate processes is provided in SIFT as the "GGC" connectivity estimator. The interested reader should consult (Ding et al., 2006) for an excellent tutorial on the Granger-Geweke approach.

There is a direct relationship between bivariate time-domain and frequency-domain GC. If F_{ij} is the time-domain GC estimator ((Eq 4.1) and $W(f)_{ij}$ is the frequency-domain Granger-Geweke estimator, then the following equivalency holds:

$$F_{ij} = \int_0^{F_s/2} W(f)_{ij}\, df$$

It is unknown whether a similar equivalency exists for other multivariate GC estimators, such as the PDC and dDTF. However, in practice, integrating these estimators over a range of frequencies provides a simple way to obtain a general time-domain representation of the estimator.

4.3. A partial list of VAR-based spectral, coherence and GC estimators

Table 4 contains a list of the major spectral, coherence, and GC/information flow estimators currently implemented in SIFT. Each estimator can be derived from the quantities $S(f), A(f), H(f),$ and Σobtained in section 3.3. , with the exception of the renormalized PDC (rPDC). The rPDC requires estimating the [$(Mp)^2$ x $(Mp)^2$] inverse cross-covariance matrix of the VAR[*p*] process. SIFT achieves this using an efficient iterative algorithm proposed in (Barone, 1987) and based on the doubling algorithm of (Anderson and Moore, 1979). These estimators and more can be computing using the SIFT's functions **`pop_est_mvarConnectivity()`** or the low-level function **`est_mvtransfer()`**.

Table 4. A partial list of VAR-based spectral, coherence, and information flow / GC estimators implemented in SIFT.

	Estimator	Formula	Primary Reference and Notes
Spectral M.	Spectral Density Matrix	$S(f) = X(f)X(f)^*$ $= H(f)\Sigma H(f)^*$	(Brillinger, 2001) $S_{ii}(f)$ is the spectrum for variable *i*. $S_{ij}(f) = S_{ji}(f)^*$ is the cross-spectrum between variables *i* and *j*.
Coherence Measures	Coherency	$C_{ij}(f) = \frac{S_{ij}(f)}{\sqrt{S_{ii}(f)S_{jj}(f)}}$ $0 \le \lvert C_{ij}(f) \rvert^2 \le 1$	(Brillinger, 2001) Complex quantity. Frequency-domain analog of the cross-correlation. The magnitude-squared coherency is the *coherence* $Coh_{ij}(f) = \lvert C_{ij}(f) \rvert^2$. The phase of the coherency can be used to infer lag-lead relationships, but, as with cross-correlation, this should be treated with caution if the coherence is low, or if the system under observation may be open-loop.
	Imaginary Coherence (iCoh)	$iCoh_{ij}(f) = \mathrm{Im}(C_{ij}(f))$	(Nolte et al., 2004) The imaginary part of the coherency. This was proposed

			as a coupling measure invariant to linear instantaneous volume-conduction. $\text{iCoh}_{ij}(f) > 0$ only if the phase lag between *i* and *j* is non-zero, or equivalently, $0 < \text{angle}(C_{ij}(f)) < 2\pi$
	Partial Coherence (pCoh)	$P_{ij}(f) = \frac{\hat{S}_{ij}(f)}{\sqrt{\hat{S}_{ii}(f)\hat{S}_{jj}(f)}}$ $\hat{S}(f) = S(f)^{-1}$ $0 \le \lvert P_{ij}(f)\rvert^2 \le 1$	(Brillinger, 2001) The partial coherence between *i* and *j* is the remaining coherence which cannot explained by a linear combination of coherence between *i* and *j* and other measured variables. Thus, $P_{ij}(f)$ can regarded as the *conditional* coherence between *i* and *j* with respect to all other measured variables.
	Multiple Coherence (mCoh)	$G_i(f) = \sqrt{1 - \frac{\det(S(f))}{S_{ii}(f)\mathbf{M}_{ii}(f)}}$ $\mathbf{M}_{ii}(f)$ is the **minor** of $S(f)$ obtained by removing the ith row and column of $S(f)$ and returning the determinant.	(Brillinger, 2001) Univariate quantity which measures the total coherence of variable *i* with all other measured variables.
Partial Directed Coherence Measures	Normalized Partial Directed Coherence (PDC)	$\pi_{ij}(f) = \frac{A_{ij}(f)}{\sqrt{\sum_{k=1}^{M} \lvert A_{kj}(f)\rvert^2}}$ $0 \le \lvert \pi_{ij}(f)\rvert^2 \le 1$ $\sum_{j=1}^{M} \lvert \pi_{ij}(f)\rvert^2 = 1$	(Baccalá and Sameshima, 2001) Complex measure which can be interpreted as the conditional granger causality from *j* to *i* normalized by the total amount of causal outflow from *j*. Generally, the magnitude-squared PDC $\lvert \pi_{ij}(f)\rvert^2$ is used.
	Generalized PDC (GPDC)	$\bar{\pi}_{ij}(f) = \frac{\frac{1}{\Sigma_{ii}} A_{ij}(f)}{\sqrt{\sum_{k=1}^{M} \frac{1}{\Sigma_{ii}^2} \lvert A_{kj}(f)\rvert^2}}$ $0 \le \lvert \bar{\pi}_{ij}(f)\rvert^2 \le 1$ $\sum_{j=1}^{M} \lvert \bar{\pi}_{ij}(f)\rvert^2 = 1$	(Baccalá and Sameshima, 2007) Modification of the PDC to account for severe imbalances in the variance of the innovations. Theoretically provides more robust small-sample estimates. As with PDC, the squared-magnitude $\lvert \bar{\pi}_{ij}(f)\rvert^2$ is typically used

	Renormalized PDC (rPDC)	$\lambda_{ij}(f)=Q_{ij}(f)^{*}V_{ij}(f)^{-1}Q_{ij}(f)$ where $Q_{ij}(f)=\begin{pmatrix}\mathrm{Re}[A_{ij}(f)]\\ \mathrm{Im}[A_{ij}(f)]\end{pmatrix}$ and $V_{ij}(f)=\sum_{k,l=1}^{p}R_{jj}^{-1}(k,l)\Sigma_{ii}Z(2\pi f,k,l)$ $Z(\omega,k,l)$ $=\begin{pmatrix}\cos(\omega k)\cos(\omega l) & \cos(\omega k)\sin(\omega l)\\ \sin(\omega k)\cos(\omega l) & \sin(\omega k)\sin(\omega l)\end{pmatrix}$ ***R*** is the [(*Mp)*^{2} x *(Mp)*^{2}] covariance matrix of the VAR[*p*] process (Lütkepohl, 2006)	(Schelter et al., 2009) Modification of the PDC. Non-normalized PDC is renormalized by the inverse covariance matrix of the process to render a scale-free estimator (does not depend on the unit of measurement) and eliminate normalization by outflows and dependence of statistical significance on frequency. To our knowledge SIFT is the first publically available toolbox to implement this estimator.
Directed Transfer Function Measures	Normalized Directed Transfer Function (DTF)	$\gamma_{ij}(f)=\frac{H_{ij}(f)}{\sqrt{\sum_{k=1}^{M}\lvert H_{ik}(f)\rvert^{2}}}$ $0\le\lvert\gamma_{ij}(f)\rvert^{2}\le 1$ $\sum_{j=1}^{M}\lvert\gamma_{ij}(f)\rvert^{2}=1$	(Kaminski and Blinowska, 1991; Kaminski et al., 2001) Complex measure which can be interpreted as the total information flow from *j* to *i* normalized by the total amount of information inflow to *i.* Generally, the magnitude-squared DTF $\lvert\gamma_{ij}(f)\rvert^{2}$ is used and, in time-varying applications the DTF should not be normalized.
	Full-Frequency DTF (ffDTF)	$\eta_{ij}^{2}(f)=\frac{\lvert H_{ij}(f)\rvert^{2}}{\sum_{f}\sum_{k=1}^{M}\lvert H_{ik}(f)\rvert^{2}}$	(Korzeniewska, 2003) A different normalization of the DTF which eliminates the dependence of the denominator on frequency allowing more interpretable comparison of information flow at different frequencies.
	Direct DTF (dDTF)	$\delta_{ij}^{2}(f)=\eta_{ij}^{2}(f)P_{ij}^{2}(f)$	(Korzeniewska, 2003) The dDTF is the product of the ffDTF and the pCoh. Like the PDC, it can be interpreted as frequency-domain conditional GC.

Granger-Geweke	Granger-Geweke Causality (GGC)	$F_{ij}(f) = \frac{\left(\Sigma_{jj} - (\Sigma_{ij}^2 / \Sigma_{ii})\right) \left\| H_{ij}(f) \right\|^2}{S_{ii}(f)}$	(Geweke, 1982; Bressler et al., 2007) For bivariate models (M = 2), this is identical to Geweke's 1982 formulation. However, it is not yet clear how this extends to multivariate models (M > 2).

4.4. Time-Frequency GC

In section 3.4. we discussed using adaptive VAR models to model nonstationary time series. These methods allow us to obtain a sequence of time-varying VAR coefficient matrices. A time-frequency representation of the spectrum, coherence or information-flow/GC can thus easily be obtained by computing one or more of the estimators in Table 4 for each coefficient matrix. Figure 2 shows an example of a time-frequency image of dDTF information flow between two neural processes. Each column of the image corresponds to the dDTF "spectrum" at a given point in time.

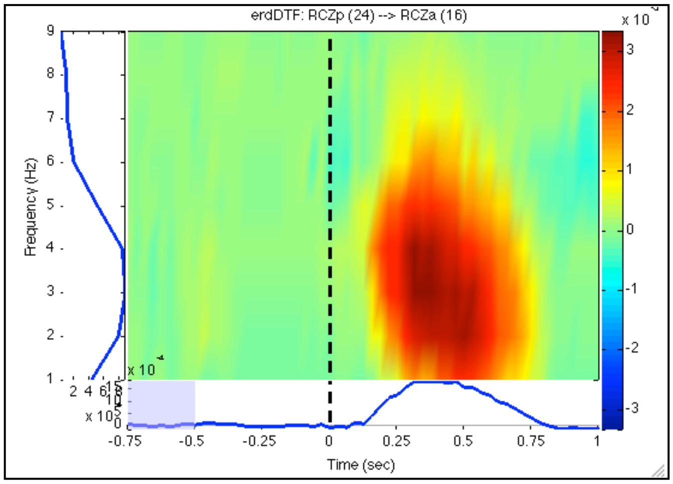


Figure 2. A time-frequency image showing the dDTF between two processes for a selected range of frequencies and times. Frequency is on the y-axis and Time on the x-axis. Red (blue) indicates more (less) information flow, relative to a baseline period (purple shaded region).

4.5. (Cross-) correlation does not imply (Granger-) causation

An important result of the definition of granger causality is that it provides a much more stringent criterion for causation (or information flow) than simply observing high correlation with some lag-lead relationship. A common approach for inferring information flow is to compute the cross-correlation (or cross-partial-correlation) between two variables for a range of time lags and determine whether there exists a peak in the correlation at some non-zero lag. From this we might infer that the leading variable "causes"

– or transmits information to – the lagged variable. However, using such an approach to infer causation, or even a direction of information flow, can be quite misleading for several reasons.

Firstly, the cross-correlation is a symmetric measure and is therefore unsuitable for identifying lag-lead relationships in systems with feedback (closed-loop systems) (Chatfield, 1989). It is currently understood that many neural systems exhibit feedback, albeit potentially on a large enough time scale that they system may appear locally open-loop.

Secondly, even if the system under observation is open-loop, a clear peak in the cross-correlation at some non-zero lag would satisfy Assumption 1 of GC (causes must precede effects in time) but not Assumption 2 (the past of a cause must share information with the future of the effect that cannot be explained by the past of all other measured variables, including the effect). In this regard it is fundamentally different than GC. As it turns out, the ability for GC to test Assumption 2 is what makes it such a powerful tool for causal inference, in contrast to simple correlative measures.

To illustrate: suppose we are observing two ants independently following a pheromone trail towards some tasty morsel. Ant 1 started the journey two minutes before Ant 2 and so he appears to be "leading" Ant 2. If we compute the cross-correlation between the two ants' trajectories for a range of time lags we would find a high correlation between their trajectories and, furthermore, we would find the correlation was peaked at a non-zero lag with Ant 1 leading Ant 2 by a lag of two minutes. But it would be foolish to say that Ant 1 was "causing" the behavior of Ant 2. In fact, not only is there no causal relationship whatsoever between the two, but there is not even any information being transmitted between the two ants. They are conditionally independent of each other, given their own past history and given the fact that each is independently following the pheromone trail (this is the "common (exogenous) cause" that synchronizes their behavior). If we were to intervene and remove Ant 2 (Ant 1), Ant 1 (Ant 2) would continue on his way, oblivious to the fact that his comrade is no longer in lock-step with him. Consequently, if we calculate the Granger-causality between the two trajectories we will find that the GC is zero in both directions: there is no information in the history of either ant that can help predict the future of the other ant above and beyond the information already contained in each ant's respective past.

Because the spectral coherence is simply the Fourier transform of the cross-correlation (and therefore the frequency-domain representation of the cross-correlation), the same limitations hold for coherence as for cross-correlation regarding inference of directionality of information flow or causation. Namely, using the phase of coherence to infer directionality of information flow in some frequency (as is often done in the neuroscience community) may be highly misleading if there is even moderate feedback in the system (or if the coherence is low). Coherence is not necessarily a measure of information flow, but rather correlation between two processes at a particular frequency (a useful analogy here, similar to the ants, is to consider two pendulums on opposite sides of the globe swinging in synchrony at the same frequency, with one pendulum started ¼ cycle before the other – their behavior is coherent, but is there information flow between them?). In contrast, frequency-domain extensions of Granger-causality condition on the past history of the processes and, assuming all relevant variables have been included in the model, can correctly distinguish between such spurious forms of information flow or causation and "true" information flow.

5. Statistics

When making inferences about information flow or causation in the neural systems, it is highly important to also produce confidence intervals and statistical significance thresholds for the estimator. The most common statistical tests used in neural system identification are listed in Table 5. Statistical routines in SIFT are designed to address one or more of these three tests and currently fall under two main categories: non-parametric surrogate statistics, and asymptotic analytic statistics.

Table 5. Common statistical tests. Here C(i,j) is the measured information flow from process *j* to process *i*. C_{null} is the expected measured information flow when there is no true information flow, C_{base} is the expected information flow in some baseline period.

Test	Null Hypothesis	What question are we addressing?	Applicable Methods
H_{null}	$C(i,j) \leq C_{null}(i,j)$	Is there significantly non-zero information flow from process *j*→*i* ?	Phase randomization Analytic tests
H_{base}	$C(i,j) = C_{base}(i,j)$	Is there a difference in information flow relative to the baseline?	Bootstrap resampling
H_{AB}	$C_A(i,j) = C_B(i,j)$	Is there a difference in information flow between experimental conditions/populations A and B?	Bootstrap resampling

5.1. Asymptotic analytic statistics

In recent years, asymptotic analytic tests for non-zero information flow (H_{null}) at a given frequency have been derived and validated for the PDC, rPDC, and DTF estimators (Schelter et al., 2005; Eichler, 2006b; Schelter et al., 2009). These tests have the advantage of requiring very little computational time, compared to surrogate statistics. However, these tests are also based on asymptotic properties of the VAR model, meaning they are *asymptotically* accurate and may suffer from inaccuracies when the number of samples is not very large or when assumptions are violated. Nonetheless, these tests can provide a useful way to quickly check for statistical significance, possibly following up with a more rigorous surrogate statistical test. These tests are implemented in SIFT's **`stat_analyticTests()`** function. To our knowledge, SIFT is the only publically available toolbox that implements these analytic tests.

5.2. Nonparametric surrogate statistics

Analytic statistics require knowledge of the probability distribution of the estimator in question. When the distribution of an estimator is unknown, nonparametric surrogate statistical methods may be used to test for non-zero values or to compare values between two conditions. Surrogate statistical tests utilize surrogate data (modified samples of the original data) to empirically estimate the expected probability distribution of either the estimator or a *null distribution* corresponding to the expected distribution of the estimator when a particular null hypothesis has been enforced. Two popular surrogate methods, implemented in SIFT, are **bootstrap resampling, and phase randomization.**

5.2.1. Bootstrap resampling

Bootstrap resampling (Efron and Tibshirani, 1994) approximates the true distribution of an estimator by randomly resampling with replacement from the original set of data and re-computing the estimator on the collection of resampled data. This is repeated many (e.g., 200-2000) times and the value of the estimator for each resample is stored. When the procedure terminates we have an empirical distribution of the estimator from which we can compute the expected value of the estimator, obtain confidence intervals around the expected value, and perform various statistical tests (t-tests, ANOVAs, etc) to compare different values of the estimator. It can be shown that, under certain conditions, as the number of resamples approaches infinity, the bootstrap distribution approaches the true distribution of the data. The convergence speed depends largely on the nature of the data, but a rule of thumb is that 250-1000 resamples is generally sufficient to produce a reasonable estimate of the distribution. In SIFT the bootstrap distribution of an estimator can be obtained using the **`stat_bootstrap()`** function. P-values for H_{null} and H_{base} significance tests, as well as confidence intervals on the estimators can be obtained via **`stat_bootSignificanceTests()`**

5.2.2. Phase Randomization

Phase randomization (Theiler, 1992) is a method for testing for non-zero information flow in a dynamical system. The concept is quite simple: we begin by constructing the expected probability distribution (the *null distribution*) of the estimator when the null hypothesis (no information flow) has been enforced in the data. An observed value of the estimator is then compared to the quantiles of the null distribution to obtain a significance level for rejection of the null hypothesis for that value. Specifically, to generate our null distribution we randomize only the phases of each time-series, preserving the amplitude distribution. We then re-compute our estimator. Repeating this procedure many times produces the desired null distribution. Phase randomization implemented efficiently in SIFT by applying a fast-fourier transform (FFT) to obtain the complex power spectrum, replacing the phases with those of a uniform random matrix, and finally applying the inverse FFT to obtain our surrogate data matrix. This procedure ensures that (a) the surrogate spectral matrix is hermitian and (b) the real part of the surrogate spectrum is identical to that of the original data. Since our frequency-domain information flow estimators depend critically on the relative phases of the multivariate time-series, any observed information flow in the surrogate dataset should be due to chance. Therefore, values of the estimator greater than, say, 95% of the values in the null distribution should be significant at 5% level ($p < 0.05$). In SIFT, the phase-randomized null distribution can be obtained using the **`stat_phaserand()`** function. P-values for significance can be obtained via **`stat_bootSignificanceTests()`**.

Importantly, the above tests (both analytic and surrogate) are only pointwise significance tests, and therefore, when jointly testing a collection of values (for example, obtaining p-values for a complete time-frequency image), we should expect some number of non-significant values to exceed the significance threshold. As such, it is important to correct for multiple comparisons using tests such as False Discovery Rate (FDR) (Benjamini and Hochberg, 1995). SIFT currently affords FDR statistical correction using EEGLAB's **`fdr()`** function with other statistical correction methods soon to be made available.

6. Using SIFT to analyze neural information flow dynamics

This section provides a demonstration of the use of SIFT to estimate and visualize source-domain information flow dynamics in an EEG dataset. To get the most of this tutorial you may want to download the toolbox and sample data and follow along with the step-by-step instructions. The toolbox is demonstrated through hands-on examples primarily using SIFT's Graphical User Interface (GUI). Theory boxes provide additional information and suggestions at some stages of the SIFT pipeline.

In order to make the most use of SIFT's functionality, it is important to first separate your data into sources – e.g. using EEGLAB's built-in Independent Component Analysis (ICA) routines. To make use of the advanced network visualization tools, these sources should also be localized in 3D space e.g. using dipole fitting (`pop_dipfit()`). Detailed information on performing an ICA decomposition and source localization can be found in the EEGLAB wiki. In this example we will be using two datasets from a single subject performing a two-back with feedback continuous performance task depicted in Figure 3 (Onton and Makeig, 2007). Here the subject is presented with a continuous stream of letters, separated by ~1500 ms, and instructed to press a button with the right thumb if the current letter matches the one presented twice earlier in the sequence and press with the left thumb if the letter is not a match. Correct and erroneous responses are followed by an auditory "beep" or "boop" sound. Data is collected using a 64-channel Biosemi system with a sampling rate of 256 Hz. The data is common-average re-referenced and zero-phase high-pass filtered at 0.1 Hz. The datasets we are analyzing are segregated into correct (RespCorr) and incorrect (RespWrong) responses, time-locked to the button press, and separated into maximally independent components using Infomax ICA (Bell and Sejnowski, 1995). These sources are localized using a single or dual-symmetric equivalent-current dipole model using a four-shell spherical head model co-registered to the subjects' electrode locations by warping the electrode locations to the model head sphere using tools from the EEGLAB dipfit plug-in.

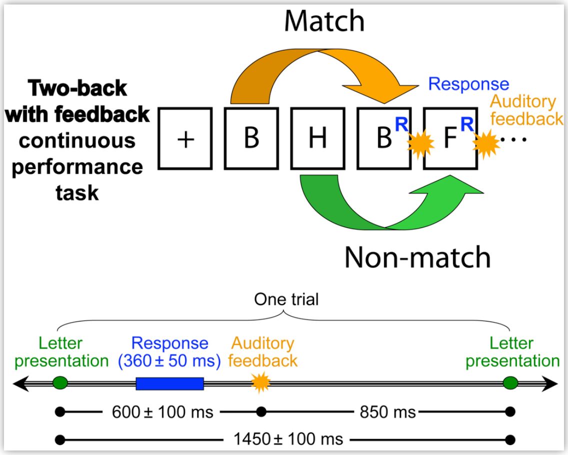


Figure 3. Two-back with feedback CPT (Onton and Makeig, 2007).

In this exercise we will be analyzing information flow between several of these anatomically-localized sources of brain activity during correct responses and error commission.

6.1. System Requirements

It is assumed that you have Matlab® 2008b or later, the Signal Processing Toolbox, Statistics Toolbox, EEGLAB and the SIFT toolbox. The latter two are downloadable from http://sccn.ucsd.edu/eeglab/ and http://sccn.ucsd.edu/sift. The SIFT toolbox must be in the Matlab path when starting EEGLAB.

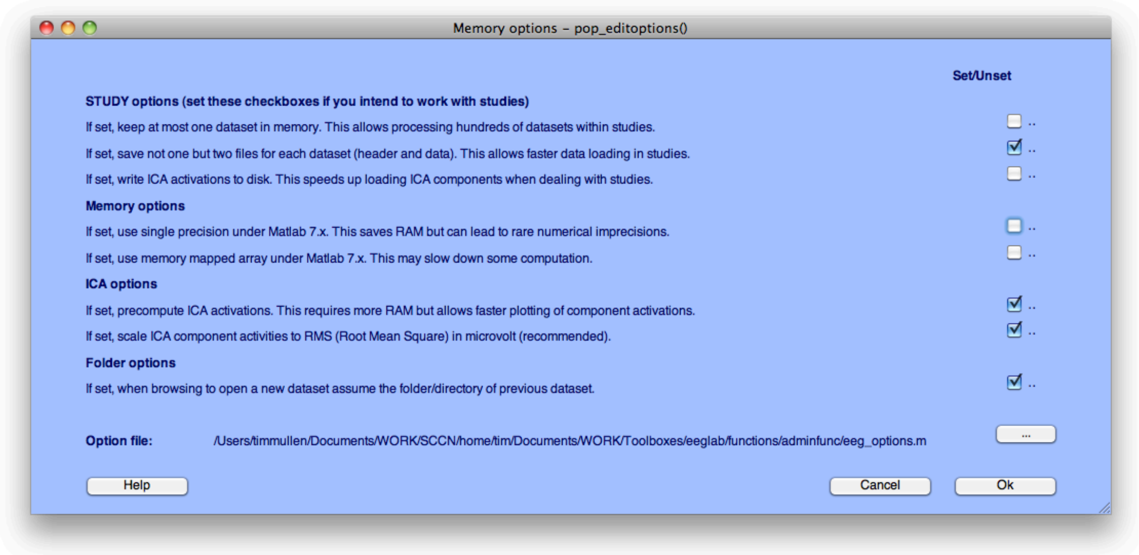
To use the ARFIT model-fitting algorithm and other (recommended) tools, you must download the free ARFIT package from http://www.gps.caltech.edu/~tapio/arfit/. After downloading and unzipping the ARFIT package, place the **`/arfit/`** folder in **`<SIFT-path>/external/`** where **`<SIFT-path>`** is the full path to the SIFT install directory.

6.2. Configuring EEGLAB

First start up EEGLAB from the MATLAB® command-prompt:

```
>> eeglab.
```

Before we load any data, we need to make sure we have the right memory options set. From the EEGLAB menu select **File→Memory and other options**. Make sure your memory options are set as in Figure 4 below.



Memory options – pop_editoptions()

Set/Unset

STUDY options (set these checkboxes if you intend to work with studies)

If set, keep at most one dataset in memory. This allows processing hundreds of datasets within studies. □ ..

If set, save not one but two files for each dataset (header and data). This allows faster data loading in studies. ☑ ..

If set, write ICA activations to disk. This speeds up loading ICA components when dealing with studies. □ ..

Memory options

If set, use single precision under Matlab 7.x. This saves RAM but can lead to rare numerical imprecisions. □ ..

If set, use memory mapped array under Matlab 7.x. This may slow down some computation. □ ..

ICA options

If set, precompute ICA activations. This requires more RAM but allows faster plotting of component activations. ☑ ..

If set, scale ICA component activities to RMS (Root Mean Square) in microvolt (recommended). ☑ ..

Folder options

If set, when browsing to open a new dataset assume the folder/directory of previous dataset. ☑ ..

Option file: /Users/timmullen/Documents/WORK/SCCN/home/tim/Documents/WORK/Toolboxes/eeglab/functions/adminfunc/eeg_options.m

Figure 4. Memory options for EEGLAB.

6.3. Loading the data

Now let's load our sample datasets in EEGLAB. We will first load the *RespWrong.set* file and then *RespCorr.set* located in the **`<SIFT-path>/Data/`** folder.

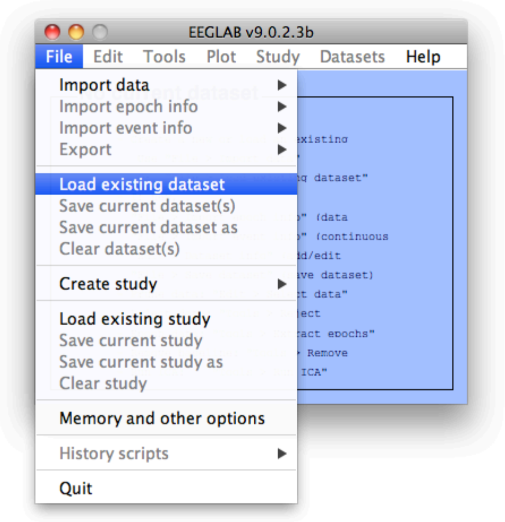
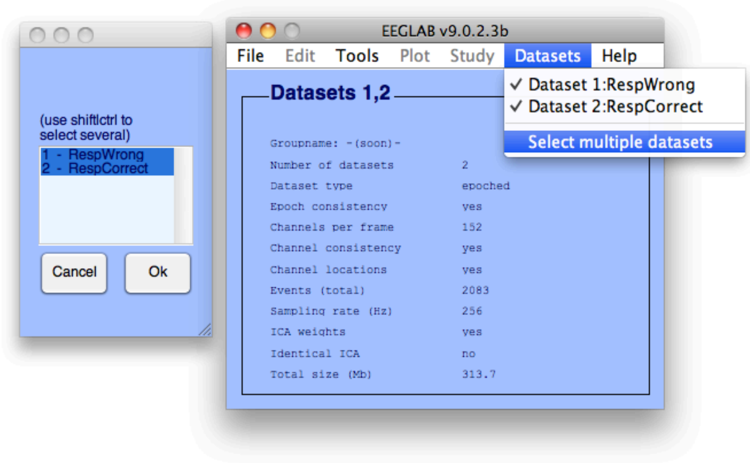


Figure 5. Load *RespWrong.set* then *RespCorr.set*

Now select both datasets from the **Datasets→Select multiple datasets** menu**.** This will enable SIFT to process these datasets in sequence, and visualize differences in connectivity between conditions.



Datasets 1,2

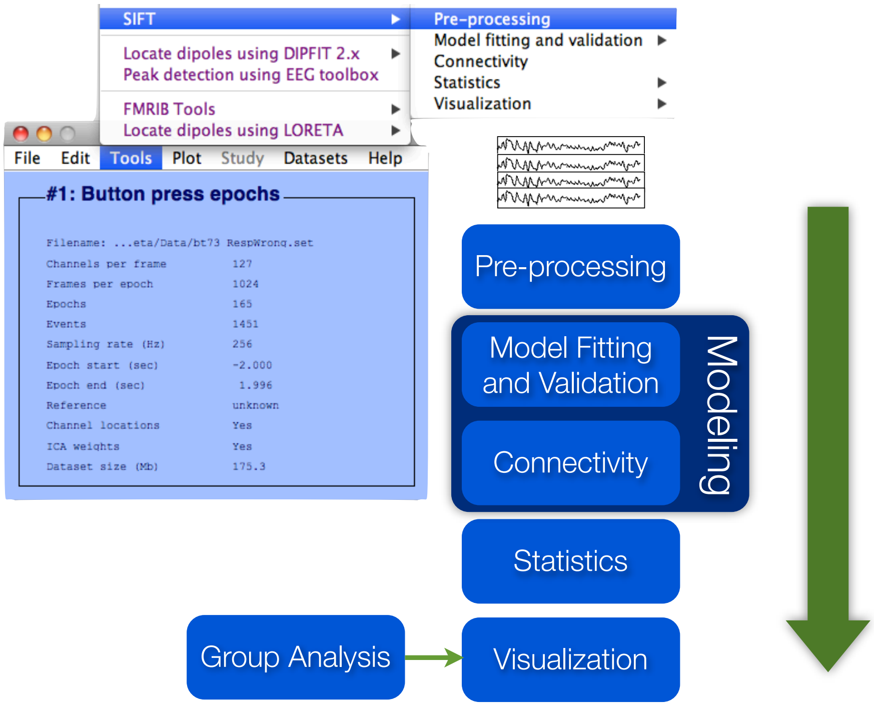
```
Groupname: -(soon)-
Number of datasets        2
Dataset type              epoched
Epoch consistency         yes
Channels per frame        152
Channel consistency       yes
Channel locations         yes
Events (total)            2083
Sampling rate (Hz)        256
ICA weights               yes
Identical ICA             no
Total size (Mb)           313.7
```

Figure 6. Select both datasets in EEGLAB

Note that the "ICA weights" field in the dataset description is set to "Yes" indicating we have ICA decompositions for both these datasets. Source separation (and localization, for advanced visualization) is currently a prerequisite for the GUI-based SIFT data-processing pipeline, although it is possible to apply low-level SIFT command-line routines to analyze connectivity using channel data. Future releases of SIFT will support a wider variety of data formats.

6.4. The SIFT analysis pipeline

Now that our data is properly loaded, we are ready to begin the SIFT data-processing pipeline. SIFT can be started from the **Tools→SIFT** menu. Figure 7 shows SIFT's data-processing pipeline. The sub-menu options correspond to SIFT's five main modules: Pre-Processing, Model Fitting and Validation, Connectivity Analysis, Statistics, and Visualization. Group Analysis is a sixth module which is applied after a cohort of datasets have been processed using Modules 1-4. This module is currently unavailable in SIFT 0.1a, but it is being integrated into EEGLAB's STUDY module and will be released in SIFT 1.0.



#1: Button press epochs

```
Filename: ...eta/Data/bt73 RespWrong.set
Channels per frame          127
Frames per epoch            1024
Epochs                      165
Events                      1451
Sampling rate (Hz)          256
Epoch start (sec)           -2.000
Epoch end (sec)              1.996
Reference                   unknown
Channel locations           Yes
ICA weights                 Yes
Dataset size (Mb)           175.3
```

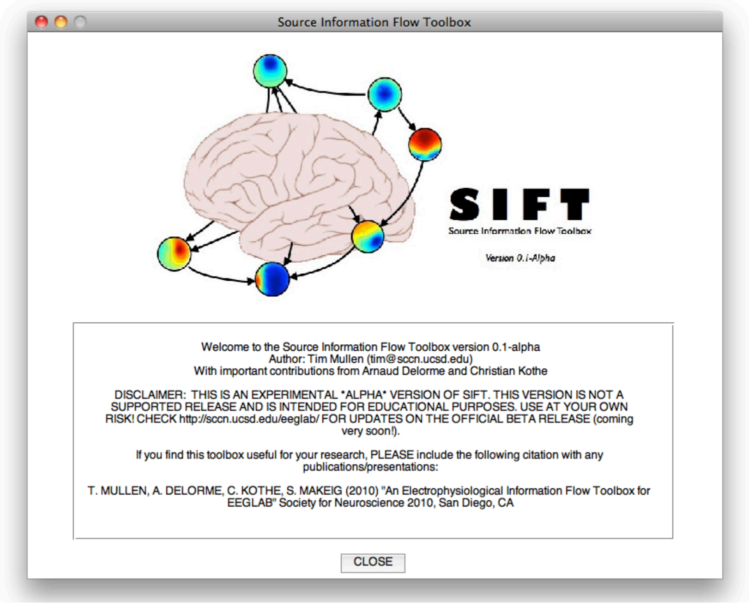
Figure 7. SIFT Data processing pipeline

6.5. Preprocessing

The first step in our pipeline is to pre-process the data. Select **SIFT→Pre-processing** to bring up the Preprocessing GUI. This can also be opened from the command-line using

```
>> EEG = pop_pre_prepData(EEG).
```

The first thing you will see is the splash screen (Figure 8). This lists acknowledgements, disclaimers, and, most importantly, the reference (Mullen et al, 2010) to cite in your presentations/publications if you use SIFT. This is very important for continuing maintenance/development of SIFT so **please don't forget to cite**!


Source Information Flow Toolbox
SIFT
Source Information Flow Toolbox
Version 0.1-Alpha

Welcome to the Source Information Flow Toolbox version 0.1-alpha
Author: Tim Mullen (tim@sccn.ucsd.edu)
With important contributions from Arnaud Delorme and Christian Kothe

DISCLAIMER: THIS IS AN EXPERIMENTAL *ALPHA* VERSION OF SIFT. THIS VERSION IS NOT A SUPPORTED RELEASE AND IS INTENDED FOR EDUCATIONAL PURPOSES. USE AT YOUR OWN RISK! CHECK http://sccn.ucsd.edu/eeglab/ FOR UPDATES ON THE OFFICIAL BETA RELEASE (coming very soon!).

If you find this toolbox useful for your research, PLEASE include the following citation with any publications/presentations:

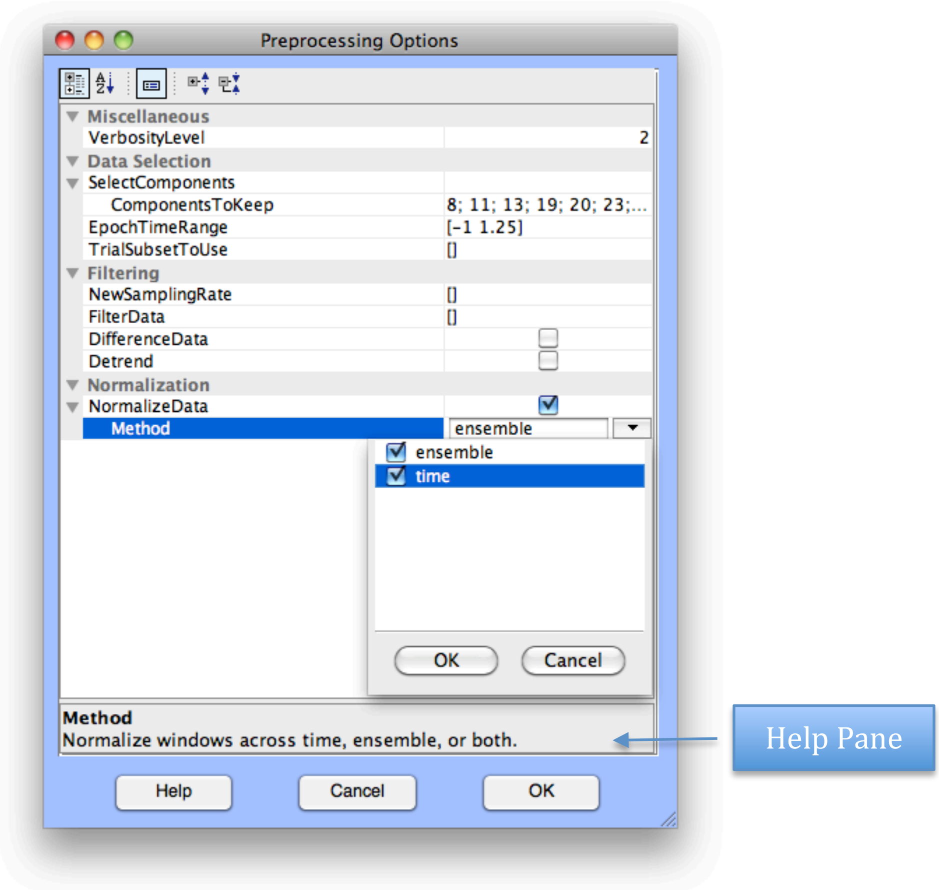
T. MULLEN, A. DELORME, C. KOTHE, S. MAKEIG (2010) "An Electrophysiological Information Flow Toolbox for EEGLAB" Society for Neuroscience 2010, San Diego, CA

Figure 8. SIFT splash screen.

Click **Close** to bring up the Preprocessing GUI. Figure 9 shows the GUI, set to the options we will be using for this example. For most GUIs, help text for each menu item appears in the **Help Pane** at the bottom of the GUI when the menu item is selected (for other GUIs help text appears in the tooltip text when the mouse is hovered over a menu item). The **VerbosityLevel** determines how much information SIFT will communicate to the user, via command-window or graphical output, throughout the remaining pipeline (0=no (or minimal) command-window output, 1=full command-window output, 2=command-window and graphical (progress-bars, etc) output). The **Data Selection** group contains options for selecting ICs and re-epoching the data. The **Filtering** group contains options for downsampling the data, filtering, differencing and linear detrending. **Normalization** (removal of mean and division by standard deviation) can be performed across the ensemble, across time, or both (first temporal then ensemble normalization). For our example, we will use the options shown in Figure 9 and in the table below:

Option	Value
VerbosityLevel	**2**
ComponentsToKeep	**8; 11; 13; 19; 20; 23; 38; 39;** (if typing component numbers manually, make sure to use semicolons to delimit numbers)
EpochTimeRange	**[-1 1.25]** (1 second before button press event to 1.25 seconds after event)
NormalizeData	**checked**
Method	**time; ensemble.**

Once these options have been input, click **OK.** Both datasets will be pre-processed using these settings and you will be prompted to create a new dataset(s) or overwrite the current datasets. Check "Overwrite" and click OK.



▼ Miscellaneous	
VerbosityLevel	2
▼ Data Selection	
▼ SelectComponents	
ComponentsToKeep	8; 11; 13; 19; 20; 23;...
EpochTimeRange	[-1 1.25]
TrialSubsetToUse	[]
▼ Filtering	
NewSamplingRate	[]
FilterData	[]
DifferenceData	□
Detrend	□
▼ Normalization	
▼ NormalizeData	☑
Method	ensemble

☑ ensemble
☑ time

OK Cancel

Method
Normalize windows across time, ensemble, or both.

Figure 9 Preprocessing GUI. Accessible through `pop_pre_prepData()`.

6.5.1. Theory: preprocessing

SIFT currently allows the user access to the following pre-processing options:

1. Component selection
2. Epoching
3. Filtering
4. Downsampling
5. Differencing
6. Detrending
7. Normalization

Many of these preprocessing steps can also be performed from EEGLAB prior to starting SIFT (see the EEGLAB Wiki). Pre-processing can be performed from the command-line using SIFT's **`pre_prepData()`** function

6.5.1.1. Component Selection

Ideally, one should fit a multivariate causal model to *all* available variables. This helps reduce the chances of false positives in connectivity (e.g., spurious causation) due to exogenous causal influences from excluded variables – i.e. "non-brain" components (Schelter et al., 2006; Pearl, 2009). However, increasing the number of variables also increases the number of parameters which we must estimate using a VAR model. For example, if we wish to fit a VAR model of order *p*, increasing the number of variables from *M* to *M*+1 requires us to estimate $(2M+1)p$ additional parameters. This in turn increases the minimal amount of data we need to adequately fit our model (see the discussion on Window Length in section 6.6.1.). Thus, when limited data is available it is often necessary to fit our models to a subset of relevant variables.

Variables can be selected using several approaches. One approach is to estimate the partial coherence between all variables using a non-parametric method (e.g., FFTs, or wavelets) and then only keep those variables that show significant partial coherence with at least one other variable. If one is working with ICA components, another (possibly complementary) approach is to select only a subset of ICs that are clearly related to brain activity. This can be performed manually (Onton and Makeig, 2009) or with the assistance of automation tools such as ADJUST (Mognon et al., 2010). The validity of this approach relies on the (rather strong) assumption that ICA has effectively removed all non-brain activity from brain components, such that there is no shared information between variables in the preserved set and those excluded from the set. See (Fitzgibbon et al., 2007; Onton and Makeig, 2009) for discussions on the topic. Both approaches can be performed using standard EEGLAB routines, as documented in the EEGLAB Wiki. In practice, one should apply a combination of these two approaches, selecting the largest subset of partially-coherent ICs which will afford an adequate model fit given the amount of data available, while giving highest priority to those ICs which likely arise from brain sources and which demonstrate significant partial coherence with one or more other "brain" ICs.

Sparse VAR estimation methods generally obviate the need to select variables by imposing additional sparsity constraints on the model coefficients. Although we may have a large number of variables, and therefore a large number of coefficients to estimate, we assume that only a small subset of the coefficients are non-zero at any given time, effectively decreasing the amount of data required to obtain unbiased coefficients estimates (Valdés-Sosa et al., 2005; Schelter et al., 2006). We are currently implementing this approach and sparse VAR modeling will be incorporated into SIFT in an upcoming release.

6.5.1.2. Epoching

This simply allows the user to analyze a subset of the original epoch. When using a sliding-window AMVAR modeling approach with a window length of *W* seconds, if one wishes to analyze time-varying connectivity from T1 to T2 seconds, he should choose his epoch length to be T1-0.5*W* to T2+0.5*W* seconds. This is because the connectivity estimate at time *t* will correspond to the connectivity over the *W*-second window *centered at time t.* Thus the earliest timepoint for which we will have a connectivity estimate is T1+0.5*W,* where T1 denotes the start of the epoch.

6.5.1.3. *Filtering*

Filtering can be a useful pre-processing step if the data contains low-frequency drift or pronounced artifacts in one or more frequency bands. Removal of drift (trend) can dramatically improve data stationarity, and thereby the quality of a VAR model fit. Since the relative phases of each time series are a key element in information flow modeling, it is critical to apply a zero-phase (acausal) filter, which introduces no phase distortion. Filtering is performed using EEGLAB's **`eegfilt()`** function. This in turn calls **`filtfilt()`** from the Matlab Signal Processing Toolbox which implements a forward-reverse (zero-phase) FIR filter.

6.5.1.4. *Downsampling*

Downsampling can be an important step when fitting parametric autoregressive models. Time series with high sampling rates generally require large model orders to appropriately capture the process dynamics, particularly if interactions occur over a relatively long time lag. As in the discussion above regarding variable selection, increasing the model order increases the number of model coefficients that must be estimated which can lead to biased estimates when limited data is available. Generally speaking, spectral and causal estimates derived from models of high order exhibit increased variability relative to those with low model order, which can complicate interpretation unless appropriate statistics are applied (Schelter et al., 2005a). In SIFT, downsampling is implemented using EEGLAB's **`pop_resample()`** function which employs a zero-phase antialiasing filter prior to downsampling. The use of a zero-phase antialiasing filter is critical for the same reasons described above for band-pass filtering.

6.5.1.5. *Differencing*

Differencing is a standard operation for improving stationarity prior to fitting time-domain parametric VAR models (Chatfield, 1989). A first-order difference filter for input *x* is given by $y_t = x_t - x_{t-1} = \nabla x_t$. This operation can be applied repeatedly to obtain an n^th^ order difference filter: $y_t = \nabla^n x_t = \nabla^{n-1} x_t - \nabla^{n-1} x_{t-1}$. Orders larger than two are rarely needed to ensure stationarity. An important point to note is that differencing is a high-pass filter and may complicate frequency-domain interpretations of connectivity (Seth, 2010). Differencing is implemented in **`pre_diffData()`**.

Recently published reports have examined the effect of different forms of downsampling, differencing, and filtering on granger-causal measures and demonstrate that in some cases, these operations may produce spurious connectivity estimates (Florin et al., 2010; Seth, 2010). In general, if the sampling rate is not excessively high (> 500 Hz) and there are not large frequency-specific artifacts in the data, it is advisable to avoid downsampling or filtering. Differencing should also be treated with great caution if one wishes to examine frequency-domain connectivity. In general, one should maintain caution when applying any transformation to their data – either to remove artifacts or improve stationarity – due to the fact that may not be well-understood how these operations affect connectivity estimates, particularly under less-than-ideal, real-world conditions. When possible, a safer alternative to stationarity-improving transformations is to instead use adaptive algorithms that either fit a model to locally-stationary windows of data (e.g., sliding-window AMVAR), estimate model coefficients from spectral density matrices (e.g., minimum-phase factorization) or utilize a state-space representation (e.g., Kalman Filter). SIFT allows the user to select from several such adaptive algorithms and also provides methods for rigorously testing the stationarity and quality of the fitted model.

6.5.1.6. Detrending

When only linear trend/drift is present in the data, an alternative to high-pass filtering is to linearly detrend the time-series using a least-squares fit. This is a phase-preserving operation. Detrending and centering (subtraction of the temporal mean) is implemented in SIFT's **`pre_detrend()`**.

6.5.1.7. Normalization

SIFT allows you to perform two kinds of normalization, *ensemble normalization* and *temporal normalization*. In section 3.4.1. we noted that ensemble normalization (pointwise subtraction of an ensemble mean and division by ensemble standard deviation) is an important preprocessing step when using multi-trial sliding-window AMVAR.

In contrast, when using short windows, temporal normalization (subtraction of mean of each window and division by standard deviation) is not a good choice since the small-sample estimate of the mean and variance within each small window will be highly biased (inaccurate). As such, SIFT only allows you to perform temporal normalization across the *whole trial* prior to segmentation. This should be performed prior to ensemble normalization and ensures that all variables have equal weight (variance) across the trial. This is important since the units of many of our derived VAR-based causal measures, like any regression method, are not scale-free and depend highly on the units and variance of the signals. Thus, severely unbalanced variances amongst the variables will likely lead to model misspecification (e.g. the variables with the highest variance may be incorrectly indicated as causal sources). It is worthwhile to note that scale-free measures such as renormalized PDC (rPDC) are theoretically immune to this problem. Nonetheless, temporal normalization, when possible and reasonable, is usually a good idea.

6.6. Model Fitting and Validation

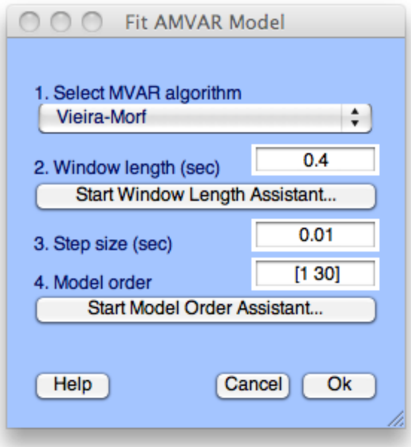
Once the data has been pre-processed, we can proceed to **Model Fitting and Validation.** SIFT 0.1a currently supports parametric VAR modeling using the Vieira-Morf (Marple, 1987), or ARFIT algorithm (Schneider and Neumaier, 2001). ARFIT must be downloaded separately and installed in <SIFT directory>/external/arfit/). SIFT currently supports time-varying parametric VAR modeling either through the use of sliding-window adaptive VAR modeling (**`est_fitMVAR()`**) or recursive-least-squares Kalman filtering (**`est_fitMVARKalman()`**).

To model the data using a sliding-window AMVAR, select **SIFT→Model Fitting and Validation→Fit AMVAR Model**. This can also be started from the command-line:

```
>> EEG = pop_est_fitMVAR(EEG);
```

You should now see a GUI similar to that displayed in Figure 12. Here we can select the MVAR algorithm, choose the sliding window length and step size, and specify the model order. ARFIT uses a modified least-squares approach while Vieira-Morf uses a multichannel geometric-mean non-least-squares lattice approach to solve for the model coefficients. ARFIT includes an additional term for the process mean, whereas Vieira-Morf assumes a zero-mean process. The current implementation of ARFIT is faster than that of the Vieira-Morf algorithm, although Vieira-Morf returns slightly better coefficient estimates. For a detailed performance comparison between these and other estimators, see (Schlögl, 2006). In this example we will use the Vieira-Morf algorithm. We will also use a **window length** of 0.4 sec (400 ms) with a **step size** of 0.1 sec (10 ms). This is approximately 1 cycle of a 2.5 Hz oscillation and should allow us to adequately model information flow from the lowest frequency band of interest (delta) up to our Nyquist frequency of 128 Hz. Consult the theory section (6.6.1.) below for more details on selecting an appropriate window length. Your GUI should appear as shown in Figure 10 with options set as in the table below:

Option	Value
Select MVAR Algorithm	**Vieira-Morf**
Window length (sec)	**0.4**
Step Size (sec)	**0.01**
Model Order	**[1 30]**



Fit AMVAR Model

1. Select MVAR algorithm: Vieira-Morf

2. Window length (sec): 0.4

3. Step size (sec): 0.01

4. Model order: [1 30]

Start Window Length Assistant...
Start Model Order Assistant...
Help
Cancel
Ok

Figure 10. AMVAR model fitting GUI generated by `pop_est_fitMVAR()`. We have selected a window length of 0.4 sec, step size of 0.01 sec and specified a model order range (for order selection) of 1 to 30.

6.6.1. Theory: selecting a window length

There are several important considerations that can aid us in choosing an appropriate window length. An appropriate window length often requires a compromise between one or more of the following considerations:

1. Local Stationarity
2. Temporal Smoothing
3. Sufficient amount of data
4. Process Dynamics and Neurophysiology

6.6.1.1. Local Stationarity

In section 3.4. we discussed issues surrounding non-stationary EEG data and introduced the concept of using a sliding window to fit VAR models to *locally stationary* data. It is thus important that our window be small enough to ensure local stationary. As we shall see in the section 6.6.4. validating our model and plotting the stability index for a chosen window size can give us a sense as to whether our window is sufficiently small to ensure local stationarity.

6.6.1.2. Temporal Smoothing

A sliding-window analysis is inherently a temporal smoothing operation. Larger windows result in model coefficients being aggregated over increasingly longer segments of data and therefore results in increased temporal smoothing. If there are rapid transient changes in process dynamics, choosing a too-large window may obscure the fine temporal structure of information flow. When multiple trials are available, a useful heuristic proposed by (Ding et al., 2000b) for obtaining a rough upper limit on the window length is to plot the bivariate correlation for a few pairs of variables, across all trials, beginning with a 1-point window and successively averaging correlation within trials and across the window for increasingly larger windows. An illustration from (Ding et al., 2000b) is reproduced in **Figure 11**. Note that with the 1-point window there are large fluctuations in correlation structure (covariance non-stationarity). As we increase the window length, we get increased temporal smoothing. In this case, a reasonable window length might be 10 points, since it reduces local covariance non-stationarity (local fluctuations in cross-correlation) while still preserving some of the temporal dynamics of interest (namely the changes in correlation structure). Of course, this suggested window length is completely application-specific; one should select a window tailored to their specific data.

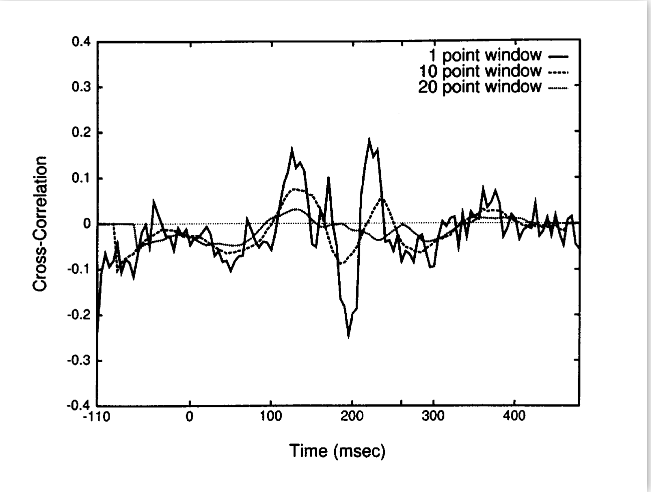


Figure 11. Cross-correlation between two intracranial EEG time-series, averaged over increasing window lengths (1 point, 10 points, 20 points), and plotted as a function of time. Figure reproduced from (Ding et al., 2000b).

6.6.1.3. Sufficient amount of data

In section 3.4.1. we noted that a minimum of M^2p data points are required to fit an M-dimensional VAR[p] model. We also stated that, in practice we would like to have 10 times that many data points to ensure an unbiased fit. This leads us to the rule-of-thumb equation

$$\boldsymbol{M^2p \leq \left(\frac{1}{10}\right) NW}$$

or, equivalently,

$$\boldsymbol{W \geq 10\left(\frac{M^2p}{N}\right)}$$

where W is the window length in points and N is the total number of trials available. SIFT performs checks on parameters (**`est_checkMVARParams()`**) and will let you know if the selected window length is sub-optimal (as well as recommend a better window length).

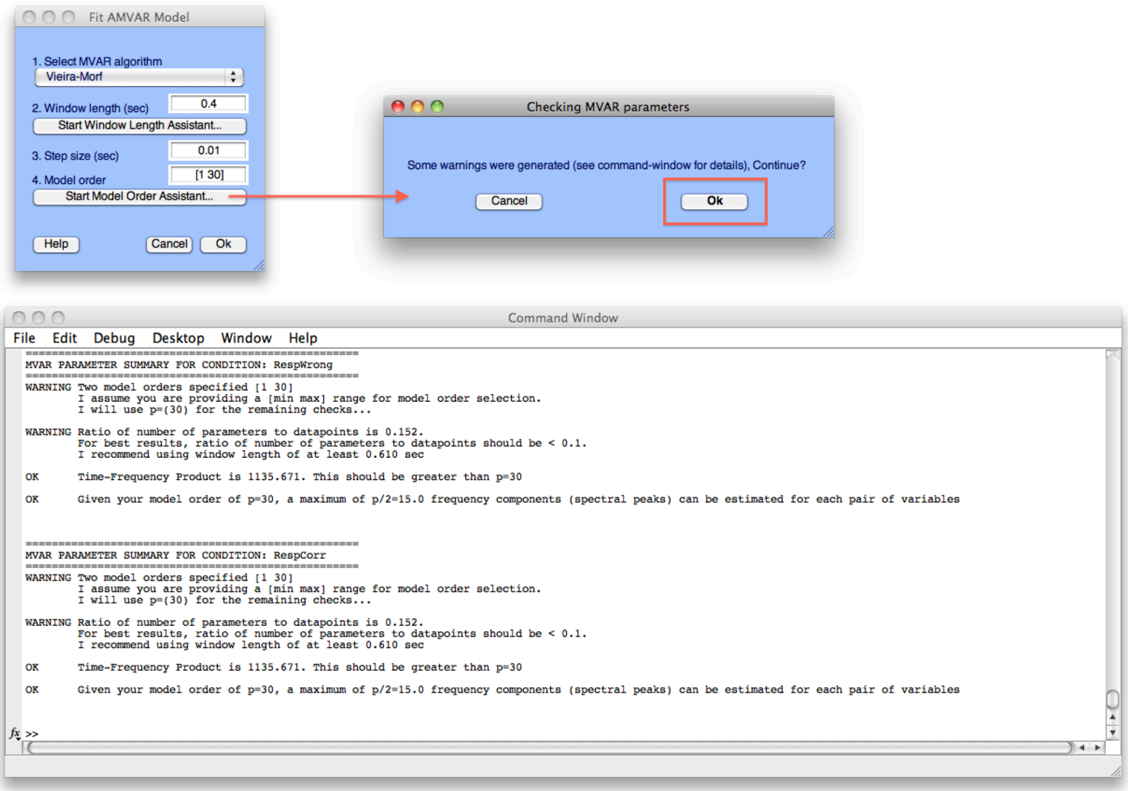
6.6.1.4. Process dynamics and neurophysiology

In section 3.5. we discussed how, when selecting an appropriate model order, one should take into account the temporal dynamics and neurophysiology of the data. The same principles hold for window length selection. Although, with a large number of trials, we could theoretically fit our model using a window length as short at *p+1* sample points long, we must consider that all interactions being modeled must occur within the selected window. In general, if we expect a maximum interaction lag of τ seconds between any two brain processes, we should make sure to select a window length of $W \geq \tau$.

Futhermore, if we are interested in frequency-domain quantities, we should consider the time-frequency uncertainty principle, which states that every increase in temporal resolution leads to a concomitant decrease in frequency resolution. In general, a useful rule-of-thumb is to ensure that the window is long enough to span approximately one cycle of the lowest frequency of interest.

6.6.2. Selecting the model order

Now that we have chosen our VAR algorithm, window length and step size, we can proceed to model order selection. Click **Start Model Order Assistant.** You should see a command-window pop up indicating that some warnings were generated (Figure 12). The Matlab command-window shows the results of a sanity check that evaluates the ratio of parameters to datapoints, calculates the number of estimable frequencies, checks the time-frequency product, and performs other relevant checks. Information is displayed for each condition, along with suggestions on optimal parameters to use, if your parameter selections are sub-optimal. Here, we are being warned that the ratio of free parameters to datapoints is greater than 0.1, which may be a cause for concern. This is because our upper model order of 30 is quite large. Let's go ahead and ignore this error (we are likely to use a much lower model order when we fit our final model). Click **OK** to continue to the model order selection GUI.



1. Select MVAR algorithm: Vieira-Morf

2. Window length (sec): 0.4

3. Step size (sec): 0.01

4. Model order: [1 30]

Some warnings were generated (see command-window for details), Continue?

```
==================================================
MVAR PARAMETER SUMMARY FOR CONDITION: RespWrong
==================================================
WARNING Two model orders specified [1 30]
        I assume you are providing a [min max] range for model order selection.
        I will use p=(30) for the remaining checks...

WARNING Ratio of number of parameters to datapoints is 0.152.
        For best results, ratio of number of parameters to datapoints should be < 0.1.
        I recommend using window length of at least 0.610 sec

OK      Time-Frequency Product is 1135.671. This should be greater than p=30

OK      Given your model order of p=30, a maximum of p/2=15.0 frequency components (spectral peaks) can be estimated for each pair of variables


==================================================
MVAR PARAMETER SUMMARY FOR CONDITION: RespCorr
==================================================
WARNING Two model orders specified [1 30]
        I assume you are providing a [min max] range for model order selection.
        I will use p=(30) for the remaining checks...

WARNING Ratio of number of parameters to datapoints is 0.152.
        For best results, ratio of number of parameters to datapoints should be < 0.1.
        I recommend using window length of at least 0.610 sec

OK      Time-Frequency Product is 1135.671. This should be greater than p=30

OK      Given your model order of p=30, a maximum of p/2=15.0 frequency components (spectral peaks) can be estimated for each pair of variables

fx >>
```

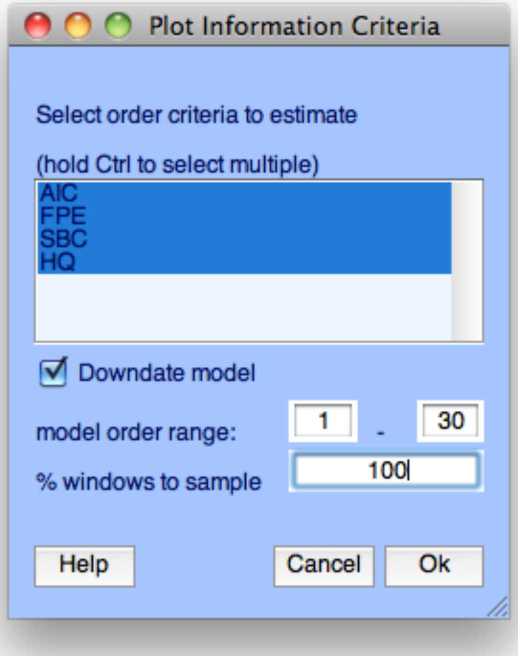
Figure 12. Sanity check on the model parameters. Command-window shows the results of a sanity check performed on the specified model parameters (this is always performed prior to model fitting).

The model order selection GUI is shown in Figure 13. Here we can choose to calculate one or more of the information criteria listed in section 3.5. over a range of model orders (p_{min} to p_{max}). If more than one window is available, the information criteria are calculated for each window, and histograms will be generated showing the distribution of optimal model orders for all windows. If we have a large number of windows and are pressed for time, we can specify a random percentage of windows for which to calculate the criteria (**% windows to sample)**. Bear in mind, however, that increasing the number of windows used will result in a better estimate of the empirical distribution of the information criteria across windows. Additionally, for a fast, approximate estimate of the information criteria, we can choose to ***downdate*** the model (Neumaier and Schneider, 2001). Rather than fitting $p_{max} - p_{min}$ VAR models of increasing order, this fits a single VAR[p_{max}] model, and downdates the noise covariance matrix to obtain approximate estimates of the information criteria for each model order $p \in \{p_{min}, \dots, p_{max}\}$.

For this example, set the parameters as shown in Figure 13 and in the table below:

Option	Value
Order criteria	**select all** (hold down Ctrl (Win/Linux) or Command (Mac) and click to select multiple criteria)
Downdate model	**checked**
Model order range	**1 - 30**
% windows to sample	**100**

Click **OK** to continue.



Select order criteria to estimate

(hold Ctrl to select multiple)

AIC
FPE
SBC
HQ

☑ Downdate model

model order range: 1 - 30

% windows to sample 100

Figure 13. The Model Order Selection GUI generated by `pop_est_selModelOrder()`

A progress bar should now popup indicating our progress for each condition (*RespCorr, RespWrong*). When this is complete, you should see the resulting figures shown in Figure 14. On left is the result of model order selection for *RespWrong* and on the right is the result for *RespCorr*. The top panel shows the information criteria (averaged across windows) plotted versus model. The vertical lines indicates the average optimal model order (model order that minimizes the information criterion) for a each criterion. The lower array of histograms show the distribution of optimal model orders for all windows, for each criterion. Note that, as mentioned in section 3.5. for many windows the AIC and FPE criteria do not appear to exhibit a clear minimum across the specified model range. In contrast, SBC shows a clear minimum peaking around *p*=5 (which is likely too low given that we will only be able to estimate 2.5 spectral peaks for each pair of variables) while HQ shows a clear minimum around *p*=11 and *p*=13 for *RespWrong* and *RespCorr*, respectively. Note, however that in both *RespWrong* and *RespCorr*, the upper limit on the model order selection criteria is approximately *p*=15. If we click on the top panel of *RespCorr*, we get an expanded view of the panel (Figure 15a). Likewise clicking on the histogram for HQ pops up an expanded view of the histogram (Figure 15b). Note that although the minimum for HQ criterion (turquoise) is at *p*= 13, the upper limit of the "elbow" for the HQ criterion is around *p=15* or *p=16*. It also appear that AIC/FPE begin to "flatten out" after *p=15.* From this we might conclude that a suitable, safe model order for all windows and conditions is be *p*=15.

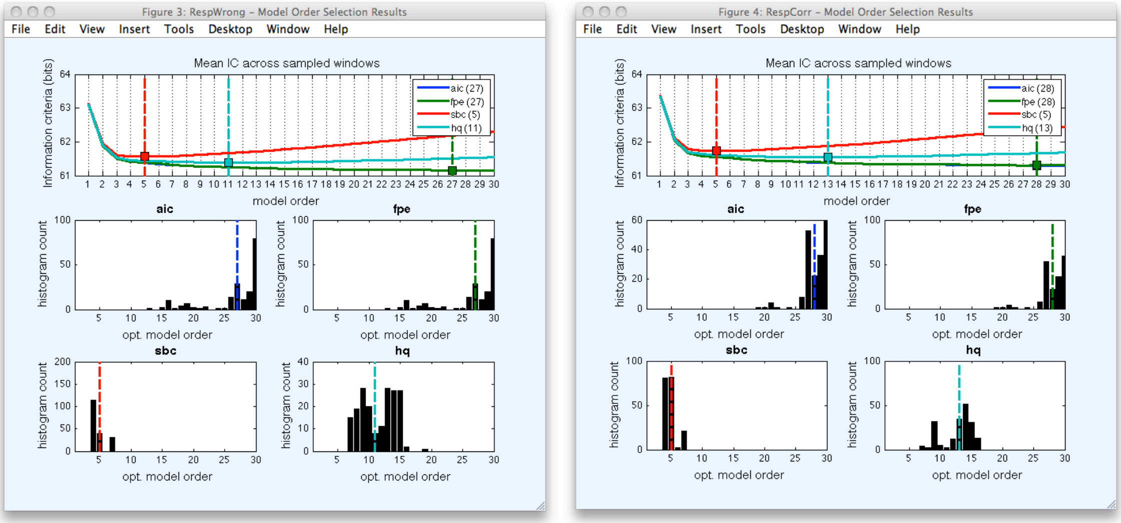


Figure 3: RespWrong – Model Order Selection Results
File Edit View Insert Tools Desktop Window Help
Mean IC across sampled windows
Information criteria (bits)
model order
aic (27)
fpe (27)
sbc (5)
hq (11)
aic
fpe
sbc
hq
histogram count
opt. model order
Figure 4: RespCorr – Model Order Selection Results
File Edit View Insert Tools Desktop Window Help
Mean IC across sampled windows
Information criteria (bits)
model order
aic (28)
fpe (28)
sbc (5)
hq (13)
aic
fpe
sbc
hq
histogram count
opt. model order

Figure 14. Results of model order selection for *RespWrong* (left) and *RespCorr* (right). The top panel plots the information criteria versus model order and marks the average optimal model order for the selected criteria. If multiple windows of data are available, the bottom panels show histograms of optimal model orders across the selected data windows.

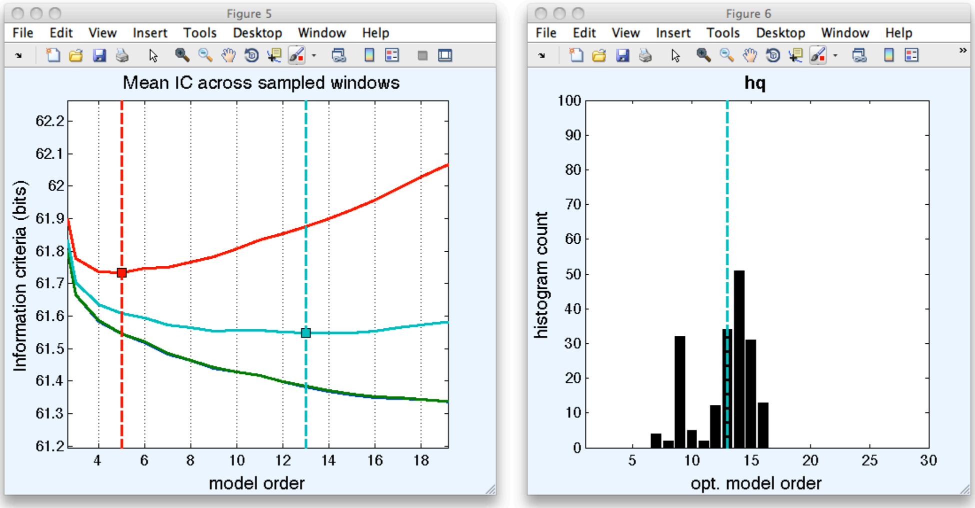
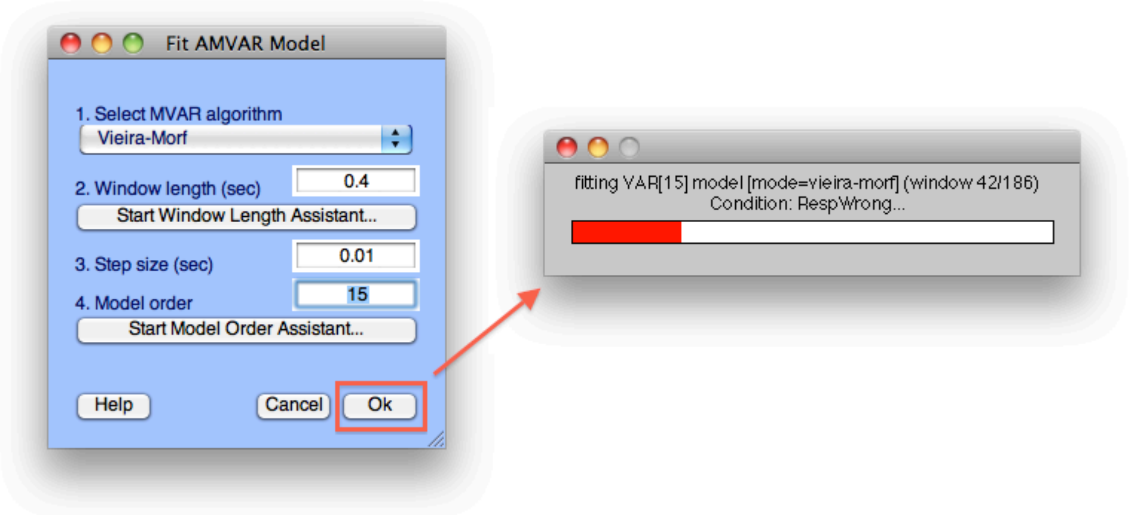


Figure 15. Left: Close-up view of SBC (red), HQ (turquoise), FPE (green), AIC (blue) information criteria plotted versus model order. Note that FPE and AIC plots are almost identical. Right: Distribution over all windows of optimal model order using HQ information criterion. Vertical markers denote distribution mean. Note the distribution is somewhat bimodal with one peak around 9 and another around 14.

6.6.3. Fitting the final model

Returning to our model order selection GUI, we now set the model order option to **15** as per the previous discussion. The final set of parameters should appear as in Figure 16 and the table below:

Option	Value
Select MVAR algorithm	**Vieira-Morf**
Window length (sec)	**0.4** (400 ms)
Step size (sec)	**0.01** (10 ms)
Model order	**15**



Fit AMVAR Model
Start Window Length Assistant...
Start Model Order Assistant...
Help
Cancel
Ok

1. Select MVAR algorithm
Vieira-Morf

2. Window length (sec)
0.4

3. Step size (sec)
0.01

4. Model order
15

fitting VAR[15] model [mode=vieira-morf] (window 42/186)
Condition: RespWrong...

Figure 16. Our final set of selected parameters for model fitting. Note that we have selected the Vieira-Morf *algorithm*, a *window length* of 0.4 seconds with a *step size* of 0.01 sec, and a *model order* of 15. Upon clicking OK, a progress bar will show us the status of the model-fitting algorithm.

Click **OK** to continue. Our sanity check should proceed and generate no warnings or errors indicating we chosen a valid set of model parameters. For each condition, the VAR[15] model will not be fit for each of the 186 windows. We should now see a progress bar indicating the model fitting progress for each condition. Depending on the speed of your computer, this may take a while, so you might want to get a coffee or do a little yoga or something. If you have little computer memory or processor speed issues, you can *increase the step size to 0.03 sec*. This will greatly reduce the computation time demands while still producing similar results as in the remainder of this exercise.

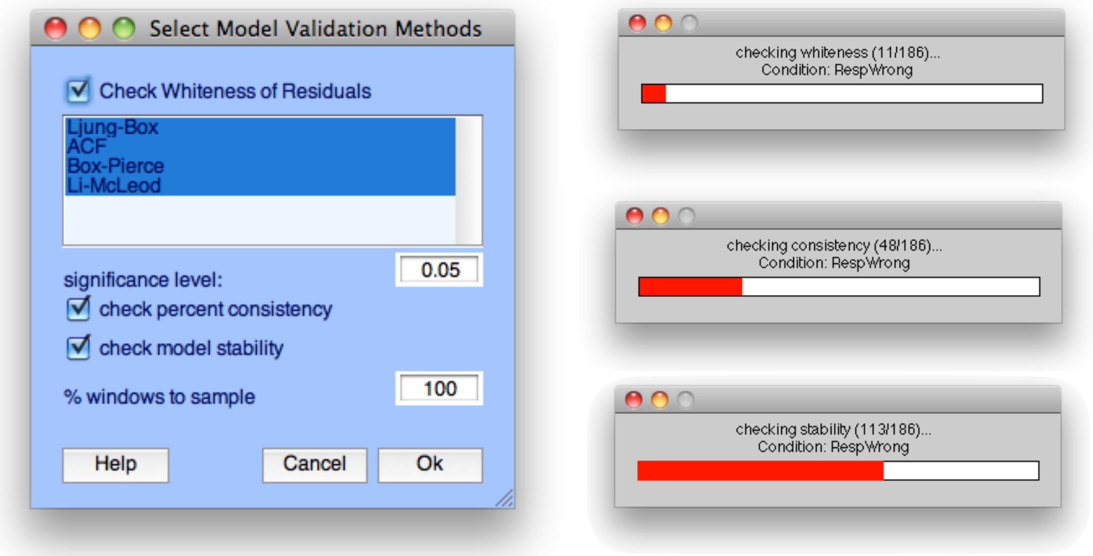
6.6.4. Validating the fitted model

After you are refreshed from that Yoga session and the model has been fit for each condition, we will need to validate our fitted model. Select **SIFT→Model Fitting and Validation → Validate Model** from the EEGLAG menu. This can also be achieved from the command-line using:

```
>> pop_est_validateMVAR(EEG);
```

You should now be presented with the GUI shown in Figure 17 (left). Here we have the option to check the whiteness of the residuals, percent consistency, and model stability for each (or a random subset) of our windows. As we discussed in section 3.6. , residual whiteness tests include Portmanteau and autocorrelation tests for correlation in the residuals of the model (which could indicate a poor model fit). Here we have the option to choose from the Ljung-Box, Box-Pierce, and Li-McLeod multivariate portmanteau tests and a simple autocorrelation function test. Percent consistency denotes the fraction of the correlation structure of the original data that is captured by the model, while model stability performs an eigenvalue test for stability/stationarity of the process. The options for this GUI should be set as shown in Figure 17 and the table below:

Option	Value
Check Whiteness of Residuals	**checked** and **select all**
significance level	**0.05**
Check percent consistency	**checked**
Check model stability	**checked**
% windows to sample	**100**



Select Model Validation Methods

☑ Check Whiteness of Residuals

Ljung-Box
ACF
Box-Pierce
Li-McLeod

significance level: 0.05

☑ check percent consistency

☑ check model stability

% windows to sample 100

checking whiteness (11/186)...
Condition: RespWrong

checking consistency (48/186)...
Condition: RespWrong

checking stability (113/186)...
Condition: RespWrong

Figure 17. Model Validation GUI generated by `pop_est_validateMVAR()`. Here we can choose to check the whiteness of residuals, percent consistency, and model stability for all (or some random subset) of windows. In this example, we have chosen a significance threshold of p<0.05 for our whiteness tests.

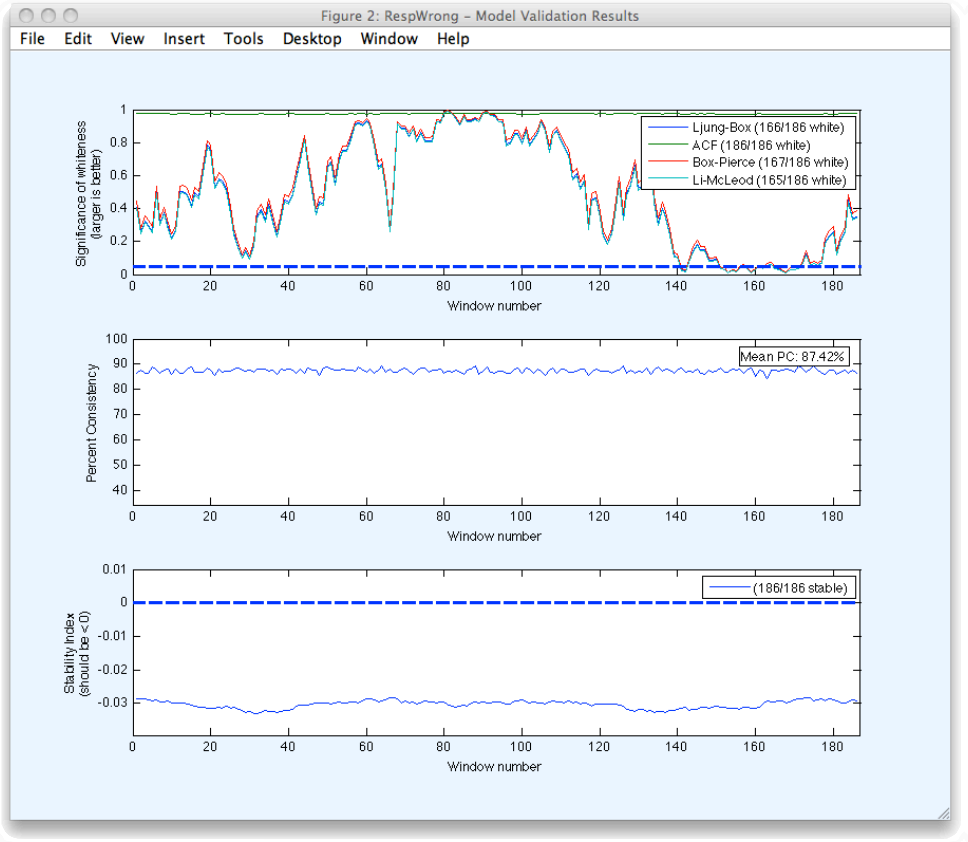
Click **OK** to continue. You should now see a sequence of progress bars for each condition as shown in Figure 17 (right). This may take a while, so go ahead and have another coffee or (preferably) finish that Yoga session.

Once model the model validation routines have completed, you should see the results shown for each condition as in Figure 18. The top panel of each figure shows the results of the whiteness tests as a function of window index (sorted in order of temporal precedence). For the Portmanteau tests, we have plotted the p-value for acceptance of the null hypothesis of correlated residuals (namely 1-*p* where *p* is the p-value for rejection of the null hypothesis). Values greater than 0.05 (blue dashed line) indicates the residuals are white at the p<0.05 level. For the ACF test (green) we have plotted the probability of an observed ACF coefficient to be within the expected interval for white residuals. Values greater than 0.95 indicates the residuals are white at the p<0.05 level. The fraction of windows that pass the whiteness test are noted in the legend. Note that the ACF tests classifies all windows as having white residuals, while the Portmanteau tests (which are much more conservative) indicate that the majority of windows are white. The fact that a range of windows near the end of the epoch marginally fail the portmanteau tests may indicate that we may want to use a slightly larger model order (e.g., *p=16*) or perhaps a smaller window size (to improve local stationarity).

The middle panel shows the percent consistency plotted versus increasing window index. Note that the PC is reliably high (μ≈87%) suggesting a reasonable model fit.

The lower panel shows the stability index for each window. Values above or near 0 indicate an unstable (and possibly nonstationary) model. In this case, we might try some additional preprocessing or shorten the window length to improve local stationarity of the data. In our example, the stability index is reliably low indicating a stable/stationary model.

The validation checks all indicate a reasonably fit model (although there may be room for improvement of the fit). Assuming we are comfortable with this we can now proceed to spectral/connectivity estimation and visualization.



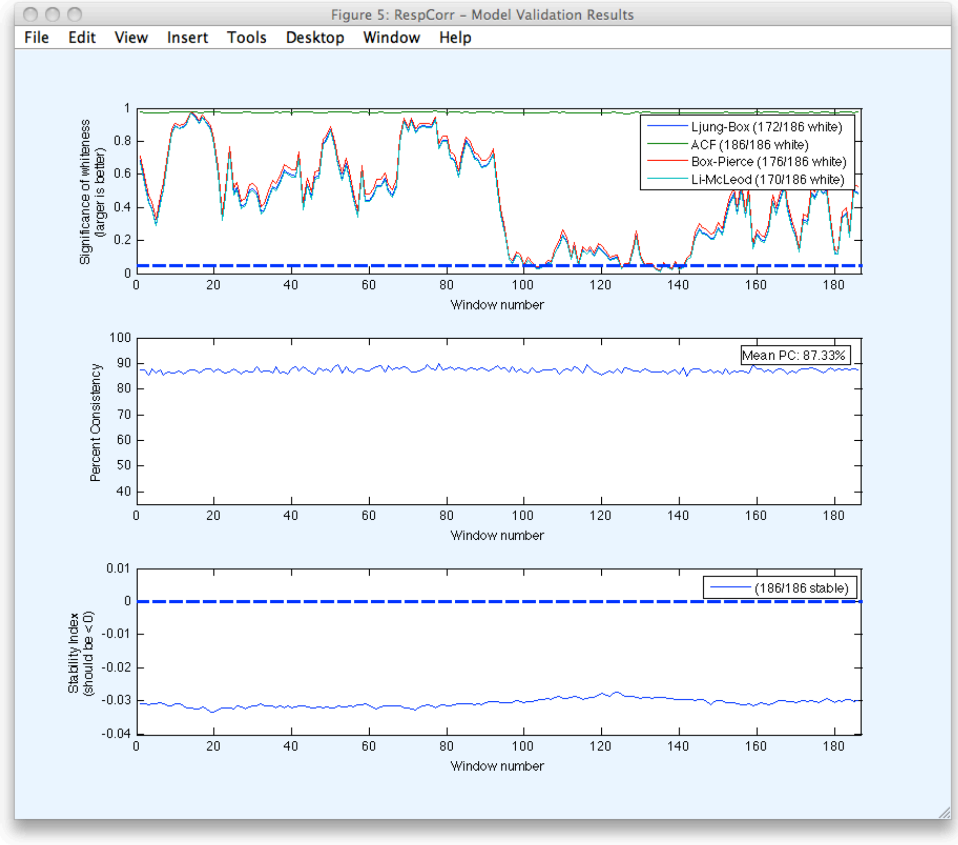


Figure 18. Results of model validation for RespWrong (top) and RespCorr (bottom) conditions. For each condition a validation statistic is plotted versus window number (sorted in order of temporal precedence). If only one window is available, bar plots are generated instead. The top panel shows the significance level for rejection of the hypothesis of correlated residuals. For Portmanteau tests (LMP, Box-Pierce, Ljung-Box), a value greater than 0.05 (dashed blue line) means we can reject the null hypothesis at the $p \leq 0.05$ level (the residuals are white). For the ACF test (green line), a value greater than 0.95 indicates white residuals at the $p \leq 0.05$ level. The middle panel shows the percent consistency. The bottom panel shows the stability index.

6.7. Connectivity Estimation

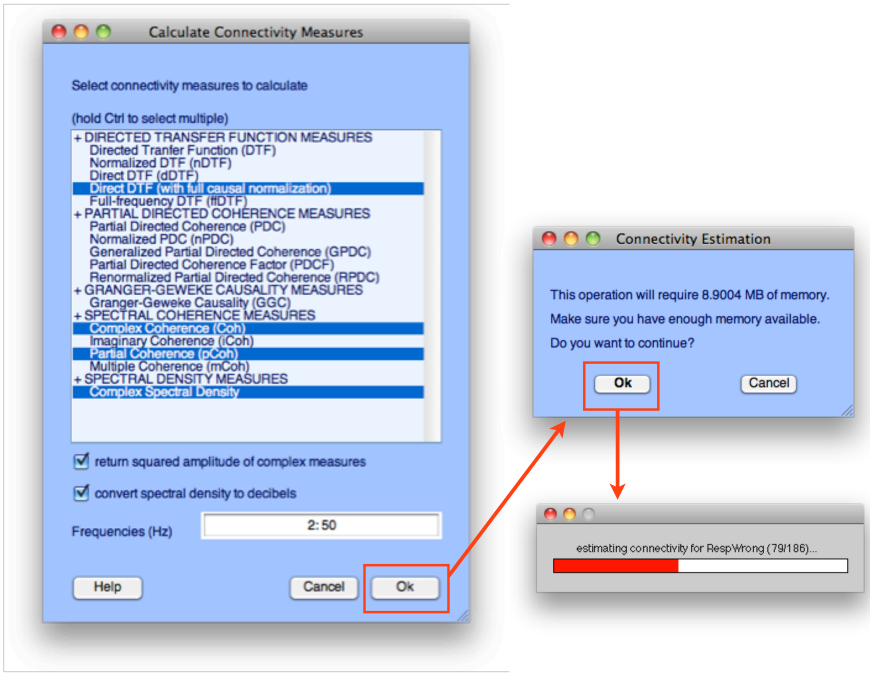
Now that our model has been fit, we'd like to calculate some frequency-domain measures, such as the spectrum, coherence, and granger-causality. Bring up the Connectivity Estimation GUI by selecting **SIFT→Connectivity**. You can start this from the command-line for a single dataset EEG using:

```
>> EEG.CAT.Conn = pop_est_mvarConnectivity(EEG);
```

You should now see the GUI shown in Figure 19 (left). Here we can compute all the measures (and more) listed in Table 4 in section 4.3. We can specify a list of frequencies at which to compute the measure(s) and we can do some simple conversions of complex measures and spectral densities.

For our example, let's compute the direct DTF (with full causal normalization, denoted dDTF08), the complex coherence, the partial coherence, and the complex spectral density over the frequency range 2-50 Hz (with 1 Hz resolution). Your options should be set as in Figure 19 and the table below:

Option	Value
Select connectivity measures	**Direct DTF (with full causal norm.)** **Complex Coherence (Coh)** **Partial Coherence (pCoh)** **Complex Spectral Density (S)**
return squared amplitude of complex measures	**checked**
convert spectral density to decibels	**checked**
Frequencies (Hz)	**2:50**



Calculate Connectivity Measures
Select connectivity measures to calculate
(hold Ctrl to select multiple)
+ DIRECTED TRANSFER FUNCTION MEASURES
Directed Tranfer Function (DTF)
Normalized DTF (nDTF)
Direct DTF (dDTF)
Direct DTF (with full causal normalization)
Full-frequency DTF (ffDTF)
+ PARTIAL DIRECTED COHERENCE MEASURES
Partial Directed Coherence (PDC)
Normalized PDC (nPDC)
Generalized Partial Directed Coherence (GPDC)
Partial Directed Coherence Factor (PDCF)
Renormalized Partial Directed Coherence (RPDC)
+ GRANGER-GEWEKE CAUSALITY MEASURES
Granger-Geweke Causality (GGC)
+ SPECTRAL COHERENCE MEASURES
Complex Coherence (Coh)
Imaginary Coherence (iCoh)
Partial Coherence (pCoh)
Multiple Coherence (mCoh)
+ SPECTRAL DENSITY MEASURES
Complex Spectral Density
Help
Cancel
Ok
Connectivity Estimation
Ok
Cancel

☑ return squared amplitude of complex measures

☑ convert spectral density to decibels

Frequencies (Hz) 2:50

This operation will require 8.9004 MB of memory.
Make sure you have enough memory available.
Do you want to continue?

estimating connectivity for RespWrong (79/186)...

Figure 19. Connectivity estimation GUI generated by `pop_est_mvarConnectivity()`. Here we have chosen to estimate the Direct DTF (with full causal normalization; dDTF08), the Complex Coherence (Coh), the Partial Coherence (pCoh), and the Spectral Density (S).

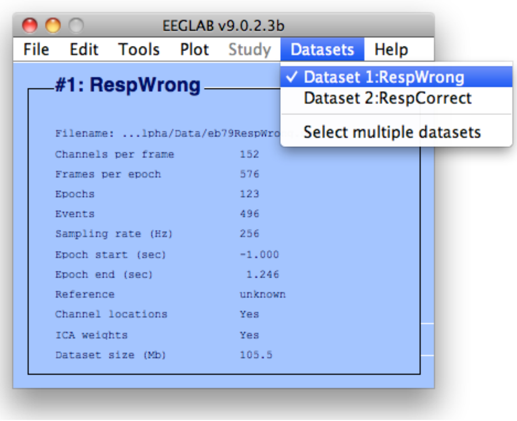
While selecting additional measures only marginally increases the computational time, doubling the number of measures will generally double the memory demands. Click **OK** to continue. You should now get a prompt notifying you of how much memory will be required (for each condition). If you have enough memory to continue, click **OK**. A progress bar will appear showing the status of the connectivity estimation for each condition.

6.8. Statistics

Once a model has been fit, and connectivity estimates computed, we often wish to compute statistics for the dataset. As discussed in section 5. This can be achieved in SIFT using several approaches, including asymptotic analytic tests (for PDC, RPDC, and DTF measures) and surrogate statistics (bootstrapping, phase randomization). The statistics module in SIFT is currently undergoing major revisions to improve its compatibility with EEGLAB's statistics routines and the STUDY module in EEGLAB. Thus, support for statistics in SIFT has been withdrawn pending the beta release.

6.9. Visualization

Once we've computed our connectivity estimates, and potentially computed some statistics, we will want to visualize the results. SIFT currently provides three main visualization programs for exploring results for a single dataset or a cohort of datasets: an Interactive Time-Frequency Grid, an Interactive BrainMovie3D, and Causal Projection (to be released in SIFT 1.0beta). For the next several sections, let's start by visualizing the results for each condition separately. Let's begin by selecting only the *RespWrong* dataset as shown in Figure 20.



#1: RespWrong

```
Filename: ...lpha/Data/eb79RespWro
Channels per frame          152
Frames per epoch            576
Epochs                      123
Events                      496
Sampling rate (Hz)          256
Epoch start (sec)           -1.000
Epoch end (sec)              1.246
Reference                   unknown
Channel locations           Yes
ICA weights                 Yes
Dataset size (Mb)           105.5
```

Figure 20. Select only the *RespWrong* dataset to continue.

6.9.1. Interactive Time-Frequency Grid

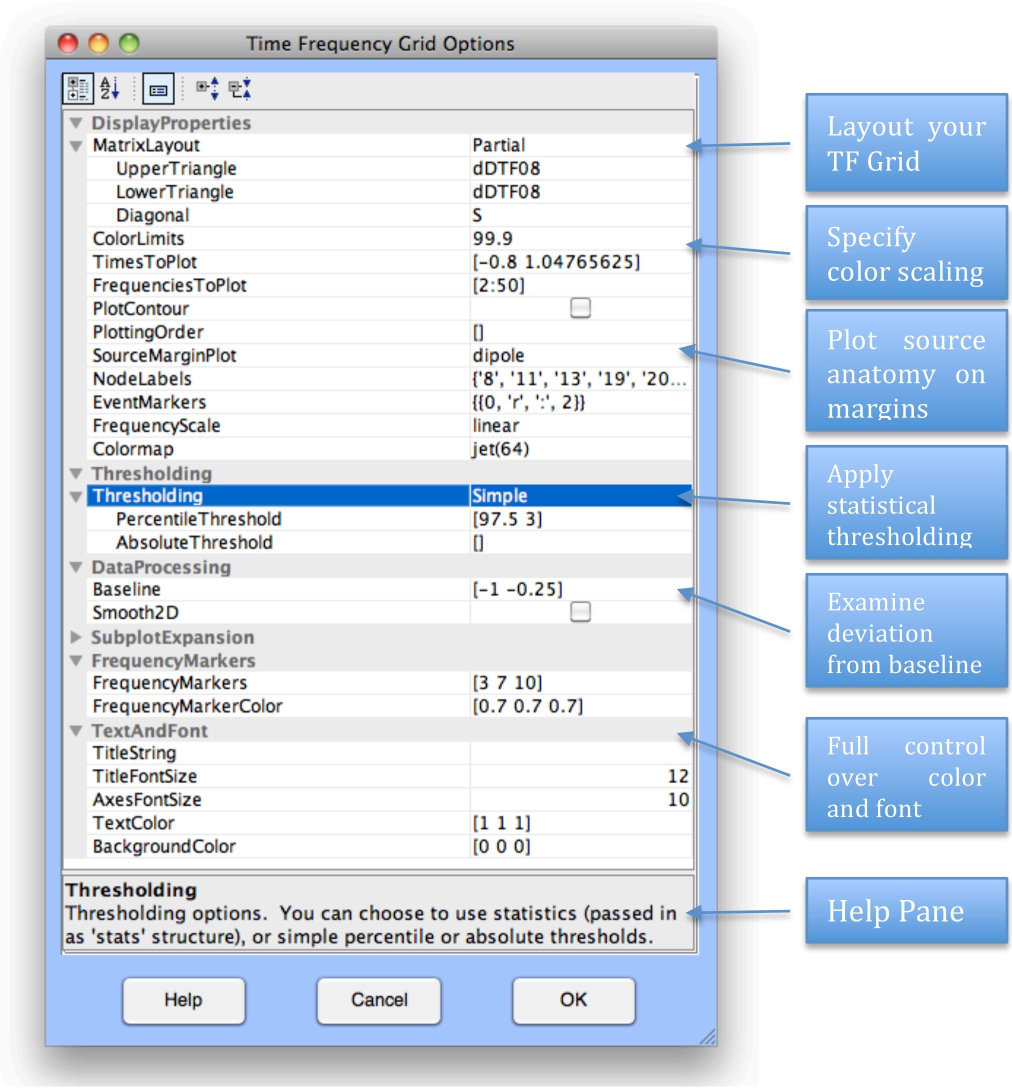
Bring up the Interactive Time-Frequency Grid Options GUI by selecting **SIFT→Visualization→ Time-Frequency Grid**. This can also be achieved from the command-line:

```
>> pop_vis_TimeFrequencyGrid(EEG);
```

This should generate the GUI seen in Figure 21. This GUI has substantially more options than we've previously seen, and we will only briefly introduce them here. Help text for each option can be obtained by expanding the Help Pane at the bottom of the PropertyGrid. The first step to creating a Time-Frequency Grid is to design the grid layout. We can plot time-frequency images of different VAR-based measures on the upper triangle, lower triangle, or diagonal of the grid. This is achieved by setting the **MatrixLayout** property to ***Partial*** and selecting the measures to plot on the various grid components. Next we should decide which **FrequenciesToPlot**. Usually we want to visualize a subset of all frequencies, to make interesting details more salient. We can also control how the color map is saturated, using *a priori* color limits or adaptive ones based on percentiles of the data. Picking a good color scaling is important for visual inspection of the data. If source localization has been performed we can set **SourceMarginPlot** to ***dipole*** to plot the anatomical locations of the sources on the column and row margins. If source locations are not available, but ICA has been performed, we can set this property to ***topoplot*** to instead plot the scalp maps of the ICs on the margins. We can provide a **Baseline** window (in seconds) for computing event-related measures. We can also perform statistical **Thresholding** or use simple percentile or absolute thresholds to establish significance. If the threshold is constant, contours can be plotted around significant regions by enabling **PlotContour**. Finally, we can customize a wide variety of display options, including placement of event and frequency markers, labels and title strings, font colors and sizes, and more.

For this example, make sure your options are set as in Figure 21 and the table below:

Option	Value	Description
MatrixLayout	**Partial**	
UpperTriangle	**dDTF08**	Put the dDTF08 on the upper triangle
LowerTriangle	**dDTF08**	Put the dDTF08 on the lower triangle
Diagonal	**S**	Put the power spectra on the diagonal
ColorLimits	**99.9**	Saturation level for colormaps. Providing a single number (as shown here) indicates that we'd like the colormap to saturate at the 99.9th percentile of all measured values
FrequenciesToPlot	**2:50**	
Thresholding	**Simple**	If statistics are available, we can use them, otherwise we get a rough sense of significance by applying simple percentile thresholding
PercentileThreshold	**[97.5, 3]**	For each frequency (dimension 3), plot only values larger (in absolute value) than 97.5% of all other measured values at that frequency
Baseline	**[-1, -0.25]**	Subtract the average connectivity in the pre-event baseline window (1 sec to ¼ sec prior to button-press event) from each measured value. This produces an event-related measure
FrequencyMarkers	**[3, 7, 10]**	This will place horizontal markers at these frequencies (Hz).
FrequencyMarkerColor	**[0.7, 0.7, 0.7]**	Here we can determine the [R G B] color(s) of the frequency markers.



Time Frequency Grid Options

▼ DisplayProperties	
▼ MatrixLayout	Partial
UpperTriangle	dDTF08
LowerTriangle	dDTF08
Diagonal	S
ColorLimits	99.9
TimesToPlot	[-0.8 1.04765625]
FrequenciesToPlot	[2:50]
PlotContour	□
PlottingOrder	[]
SourceMarginPlot	dipole
NodeLabels	{'8', '11', '13', '19', '20...
EventMarkers	{{0, 'r', ':', 2}}
FrequencyScale	linear
Colormap	jet(64)
▼ Thresholding	
▼ Thresholding	Simple
PercentileThreshold	[97.5 3]
AbsoluteThreshold	[]
▼ DataProcessing	
Baseline	[-1 -0.25]
Smooth2D	□
▶ SubplotExpansion	
▼ FrequencyMarkers	
FrequencyMarkers	[3 7 10]
FrequencyMarkerColor	[0.7 0.7 0.7]
▼ TextAndFont	
TitleString	
TitleFontSize	12
AxesFontSize	10
TextColor	[1 1 1]
BackgroundColor	[0 0 0]

Thresholding
Thresholding options. You can choose to use statistics (passed in as 'stats' structure), or simple percentile or absolute thresholds.

Layout your TF Grid

Specify color scaling

Plot source anatomy on margins

Apply statistical thresholding

Examine deviation from baseline

Full control over color and font

Help Pane

Figure 21. Interactive Time-Frequency Grid option GUI generated by `pop_vis_TimeFrequencyGrid()`. Almost every aspect of the grid is customizable, and only the most commonly-used options are represented in the GUI.

Click **OK** to continue and generate the Time-Frequency Grid. After a few seconds, you should see a figure similar to Figure 22. Here we have plotted an array of time-frequency images, where frequency is on the y-axis and time on the x-axis. On the upper and lower triangle of the grid (above and below the red-bordered diagonal) we have the dDTF (conditional GC) between each pair of sources. Information flows from columns to rows. Thus the time-frequency (TF) image at (row,col) = (3,1) shows information flow at different times and frequencies from the source above column 1 (IC8) to the source on the left of row 3 (IC13). Note that we have vertical red lines indicating events of interest (here the time of the button-press event) and horizontal gray lines denoting our frequencies of interest **(FrequencyMarkers).** On the diagonal we have plotted the event-related spectral perturbation (ERSP). Because we provided a baseline, each pixel shows the information flow or spectrum relative to the baseline window. Red denotes more information flow than in the baseline, while blue denotes less. The anatomical dipole locations for each source are rendered on the margins. Clicking on this will expand an interactive 3D MRI plot (dipplot).

Clicking on any time-frequency image generates a figure with more detail regarding the interaction between the respective sources.

Note the large bursts of information flow and spectral power in the theta (3-7 Hz) and delta (2-3 Hz) bands around, and just after, the erroneous button press. This suggests some kind of transient network synchronization occurring around the time when the button press is made in error. As a side note, observe that, although spectral power often increases with information flow/granger-causality, it does not *necessarily* do so. Consider IC 38 (7th row and column). It shows very little change in ERSP (cell (7,7)) around the button press, but appears to exhibit large changes in information flow with ICs 11 and 8. As a rule, spectral power modulation and phase synchronization/information flow can occur independently of each other – one does not imply the other. Merely observing two regions concomitantly increase their spectral amplitude does not necessarily suggest that they are communicating. Conversely, observing a lack of event-related spectral power modulation in some putative component of a brain network does not mean it is not critically participating in the network.

To explore one of these interactions further, let's go ahead and click on cell (3,1), which corresponds to IC8→IC13.

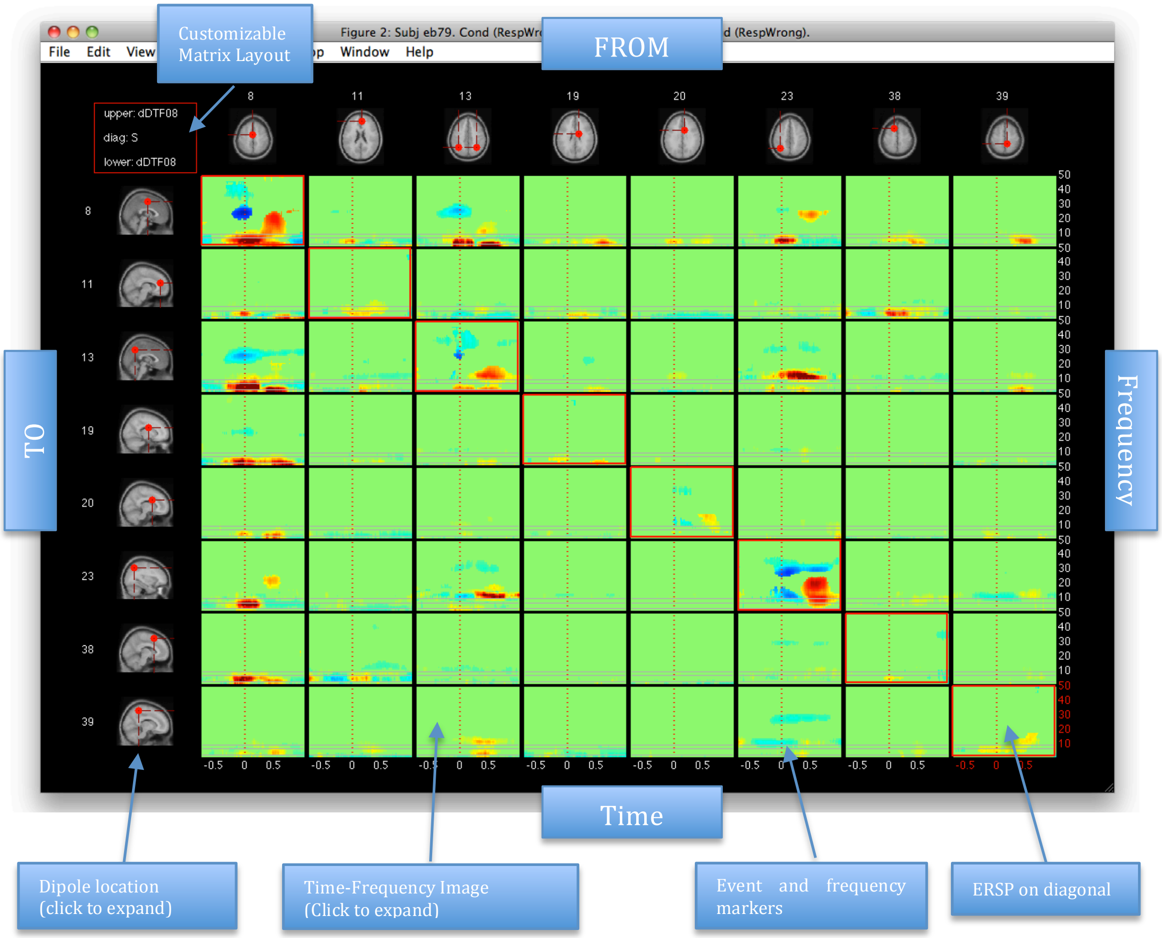


Figure 2: Subj eb79. Cond (RespWr
d (RespWrong).
File
Edit
View
p
Window
Help
Customizable Matrix Layout
FROM
upper: dDTF08
diag: S
lower: dDTF08
8
11
13
19
20
23
38
39
TO
Frequency
50
40
30
20
10
-0.5
0
0.5
Time
Dipole location (click to expand)
Time-Frequency Image (Click to expand)
Event and frequency markers
ERSP on diagonal

Figure 22. Interactive Time-Frequency Grid.

Clicking on cell (3,1) should generate an image similar to that in Figure 23a. Here we can explore the interaction between these two processes in greater detail. On the top panel of we have the dDTF flow from IC8→IC13 and on the bottom panel we have the feedback flow (IC13→IC8). On the left marginals we have rendered the column-envelope of the matrix (maximum and minimum dDTF across time), while on the bottom marginal we have the row-envelope (maximum and minimum of the dDTF across frequency). The envelopes of the two-sided thresholds for statistical significance (using the percentile threshold) are plotted as green-black lines on the marginals. Values between the threshold lines are considered non-significant and masked in the time-frequency plot. The purple shaded region on the time-marginal indicates our baseline window. Every part of the image is expandable. Clicking on the marginals generates images (b) and (c), while clicking on the source anatomical images generates images (d) and (e).

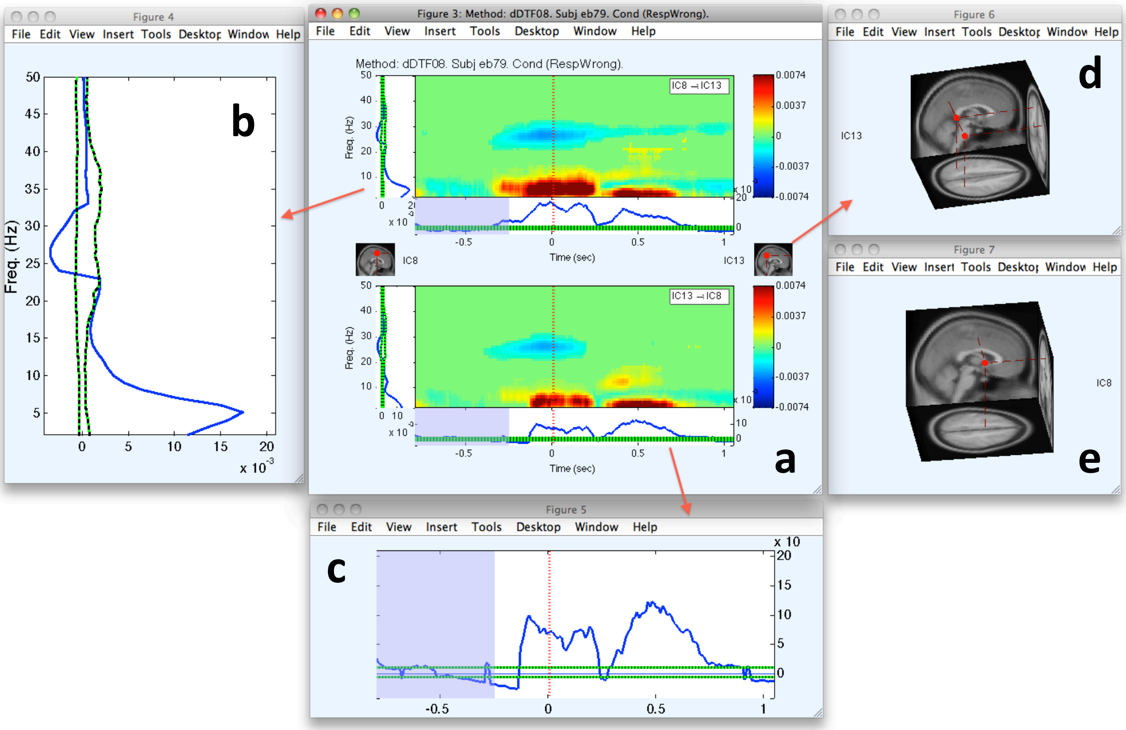


Figure 4
File Edit View Insert Tools Desktop Window Help
b
Freq. (Hz)
50
45
40
35
30
25
20
15
10
5
0
5
10
15
20
x 10^-3
Figure 3: Method: dDTF08. Subj eb79. Cond (RespWrong).
File Edit View Insert Tools Desktop Window Help
Method: dDTF08. Subj eb79. Cond (RespWrong).
IC8 → IC13
IC13 → IC8
Freq. (Hz)
Time (sec)
-0.5
0
0.5
1
0.0074
0.0037
0
-0.0037
-0.0074
IC8
IC13
a
Figure 6
File Edit View Insert Tools Desktop Window Help
d
IC13
Figure 7
File Edit View Insert Tools Desktop Window Help
IC8
e
Figure 5
File Edit View Insert Tools Desktop Window Help
c
x 10
20
15
10
5
0
-0.5
0
0.5
1

Figure 23. Expansion of the Time-Frequency Grid cell (3,1) corresponding to IC8→IC13. As with the Time-Frequency Grid, each element of the Time-Frequency Cell (a, center) is also interactively expandable. On the top panel of (a) we have the dDTF flow from IC8→IC13 and on the bottom panel we have the feedback flow (IC13→IC8). The envelopes of the time-frequency matrix are plotted on the marginal (b, c). Here, two-sided thresholds for statistical significance are plotted as green-black lines on the marginal. The purple shaded region denotes the baseline interval [-1 -0.25] sec. Clicking on the source anatomical images will generate interactive dipole plots of the sources (d, e).

By examining this time-frequency image we can see that there is a significant flow of information from IC8 to IC13 in the theta-band around the time of the erroneous button-press. There is also slightly delayed, and damped feedback from IC13→IC8. This emphasizes the points made in section 4.5. regarding the importance of using an asymmetric measure that can separate feedforward and feedback influences in closed-loop systems. Note that the early information flow is highly peaked around 5 Hz theta-band, while we see later information flow around 250-600 ms shifting to the delta-band (2-3 Hz). This is precisely in line with several observations regarding the functional role in error processing (the so-called Error-Related Negativity (ERN) seen reported in ERP literature) and electrophysiology of the cortical area to which IC8 is localized (Anterior Cingulate Cortex (ACC)) (Holroyd and Coles, 2002; Yordanova et al., 2004; Luu et al., 2004; Roger et al., 2010).

Returning to our Time-Frequency Grid in Figure 22, and turning our attention to the first column, we note that IC8 (ACC) appears to be exerting a disproportionate amount of theta-band causal influence on the rest of the network around the time of the erroneous button press. IC8 appears to be some kind of hub, synchronizing and communicating with multiple other brain areas when the error is being committed. In order to examine the full network behavior in more detail, let's generate a 3D BrainMovie.

6.9.2. Interactive Causal BrainMovie3D

The Interactive Causal BrainMovie3D (Mullen and Delorme et al, 2010; Delorme, 2005) is a way of visualizing brain network activity across time, frequency and anatomical location in the form of an anatomically localized directed graph. Directed graphs (graphical models) are powerful constructs for representing causal network structure (Pearl, 2000; Eichler, 2006a). Graph-theoretic measures are being increasing used to study brain network organization (Bullmore and Sporns, 2009). The BrainMovie3D provides a way to interactively explore multiple dimensions of source-domain network dynamics and graph structure in an intuitive and aesthetically pleasing manner.

To begin, let's bring up the BrainMovie3D GUI by selecting **SIFT→Visualization→Causal BrainMovie3D**. The command-line analogue is

>> pop_vis_causalBrainMovie3D(EEG);

You should now be presented with a control panel similar to that shown in Figure 24. This GUI has the most options of any thus far, and we will, again, only explore a small subset of the options for this example. The Help Pane (and some adventurous exploration) should allow the user to deduce the function of many of the remaining options.

One of the interesting features of the BrainMovie is the ability to modulate the color and size of nodes based on graph-theoretic measures such as inflow/outflow, indegree/outdegree, causal flow, causal density, asymmetry ratio, and other such quantities (Seth, 2005; Bullmore and Sporns, 2009). This is achieved through the **NodeColorMapping,** and **NodeSizeMapping** properties.

Measure	Description
Outflow	Sum connectivity strengths over outgoing edges
Inflow	Sum connectivity strength over incoming edges
Causal Flow	*Outflow - Inflow*
Outdegree	Number of significant outgoing edges
Indegree	Number of significant incoming edges
Causal Degree	*Outdegree-Indegree*

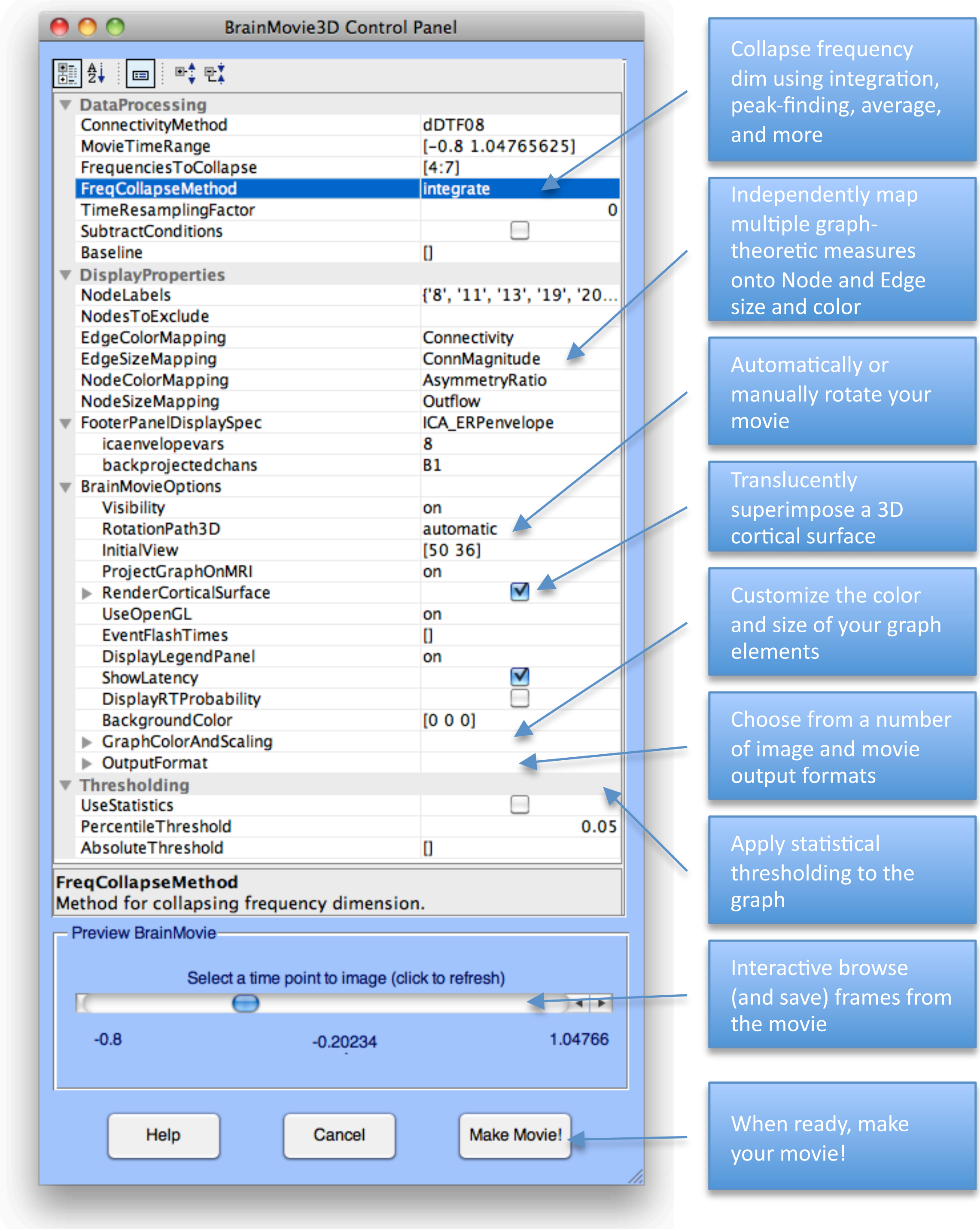
Asymmetry Ratio	$AR = \frac{Inflow - Outflow}{Inflow + Outflow}$ $-1 \leq AR \leq 1$ AR = -1 indicates all connectivity related to that node is inflowing (a causal sink) AR = +1 indicates all connectivity related to that node is outflowing (a causal source) AR = 0 indicates either balanced flow or no significant flow

Let's begin by starting with the default options and setting the remaining options as shown in Figure 24 and the table below:

Option	Value	Description
ConnectivityMethod	**dDTF08**	Which connectivity measure to use
FrequenciesToCollapse	**4:7**	Collapse frequencies across the theta range
FreqCollapseMethod	**Integrate**	Which method to use to collapse frequencies. *Integrate*: integrate over the selected frequencies *Mean*: take the mean over frequencies *Max*: take the maximum *Peak*: return the peak value over frequencies (a monotonically increasing or decreasing sequence does not have a peak)
EdgeColorMapping	**Connectivity**	The color of the edges will be mapped to connectivity strength (amount of information flow along that edge). Red = high connectivity, Green = low connectivity.
EdgeSizeMapping	**ConnMagnitude**	The size of edges of the graph (connecting "arrows") will be mapped to connectivity magnitude (absolute value of connectivity strength, useful if there are negative values as with event-related (baselined) or between-condition analysis)
NodeColorMapping	**AsymmetryRatio**	The color of a node (source) will be mapped to the asymmetry ratio of connectivity for that source. Red = causal source, Blue = causal sink. Green = balanced flow
NodeSizeMapping	**Outflow**	The size of a node will be mapped to

		the amount of information outflow from the source
FooterPanelDisplaySpec	**ICA_ERPenvelope**	This configures the footer panel at the bottom of the brainmovie. Here we've chosen to display the ERP envelope of some backprojected components
icaenvelopevars	**8**	Backproject the ERP of IC 8 (ACC)
backprojectedchans	**B1;**	… and compute the envelope only for channel B1 (FCz)
RotationPath3d	**automatic**	This creates an automatic rotation of the brainmovie when we create the final movie
ProjectGraphOnMRI	**on**	This projects the 3D directed graph onto the 2D anatomical slices
Thresholding		If statistics are available, we can use them, otherwise we get a rough sense of significance by applying simple percentile thresholding
PercentileThreshold	**0.05**	We will only render the top 5% of all connections across all time

A useful feature of the Control Panel is that we can **Preview** frames from the brainmovie before committing to render the movie. You can also save these preview frames allowing an easy way to create a network image for any desired time point. Now that we have configured our options, go ahead and click on the scrollbar in the **Preview BrainMovie** panel. It may take a second or two for the brainmovie to render, so be patient and don't click multiple times in rapid succession. If you have graphics problems, try setting the **UseOpenGL** option to ***off***. If you move the slider to approximately -0.2 seconds (200 ms before the button press) you should see a figure similar to Figure 25. We are looking at a 3D rendering of the brain of this subject derived from MRI images. To be precise, here we have coregistered (warped) this subject's electrode montage to the 4-shell spherical head model corresponding to the Montreal Neurological Institute (MNI) average brain. This accounts for the low-resolution of the MRIs (and much of the error in the dipole localization). If individual MRIs are available for the subject, an individualized head model can be constructed. The outline of the cerebral spinal fluid (CSF) is rendered translucently (**RenderCorticalSurface** option) to show us the outline of the cortical surface. As described in the section above, node and edge color and size are modulated by one or more network or graph-theoretic measures. Since we have mapped outflow to NodeColor and AsymmetryRatio to NodeSize we can immediately see that IC8 (big red ball in center) is a causal source hub here, driving many other brain areas in the theta frequency band. Note the backprojected ERP from IC8 at the bottom of the screen shows a sharp negativity around 40 ms followed by a late positive complex at around 350-400 ms. This is the well-known ERN potential known to be associated with error-processing. Try scrolling to the time point corresponding to the negative peak of the ERN (40 ms) and see what happens to the network (particularly IC 8). Try rotating the graph to examine it from different angles. Try scrolling through various stages of the epoch and exploring different mappings for node and edge color and size.



BrainMovie3D Control Panel

DataProcessing	
ConnectivityMethod	dDTF08
MovieTimeRange	[-0.8 1.04765625]
FrequenciesToCollapse	[4:7]
FreqCollapseMethod	integrate
TimeResamplingFactor	0
SubtractConditions	□
Baseline	[]
DisplayProperties	
NodeLabels	{'8', '11', '13', '19', '20…
NodesToExclude	
EdgeColorMapping	Connectivity
EdgeSizeMapping	ConnMagnitude
NodeColorMapping	AsymmetryRatio
NodeSizeMapping	Outflow
FooterPanelDisplaySpec	ICA_ERPenvelope
icaenvelopevars	8
backprojectedchans	B1
BrainMovieOptions	
Visibility	on
RotationPath3D	automatic
InitialView	[50 36]
ProjectGraphOnMRI	on
RenderCorticalSurface	☑
UseOpenGL	on
EventFlashTimes	[]
DisplayLegendPanel	on
ShowLatency	☑
DisplayRTProbability	□
BackgroundColor	[0 0 0]
GraphColorAndScaling	
OutputFormat	
Thresholding	
UseStatistics	□
PercentileThreshold	0.05
AbsoluteThreshold	[]

FreqCollapseMethod
Method for collapsing frequency dimension.

Preview BrainMovie

Select a time point to image (click to refresh)

Collapse frequency dim using integration, peak-finding, average, and more

Independently map multiple graph-theoretic measures onto Node and Edge size and color

Automatically or manually rotate your movie

Translucently superimpose a 3D cortical surface

Customize the color and size of your graph elements

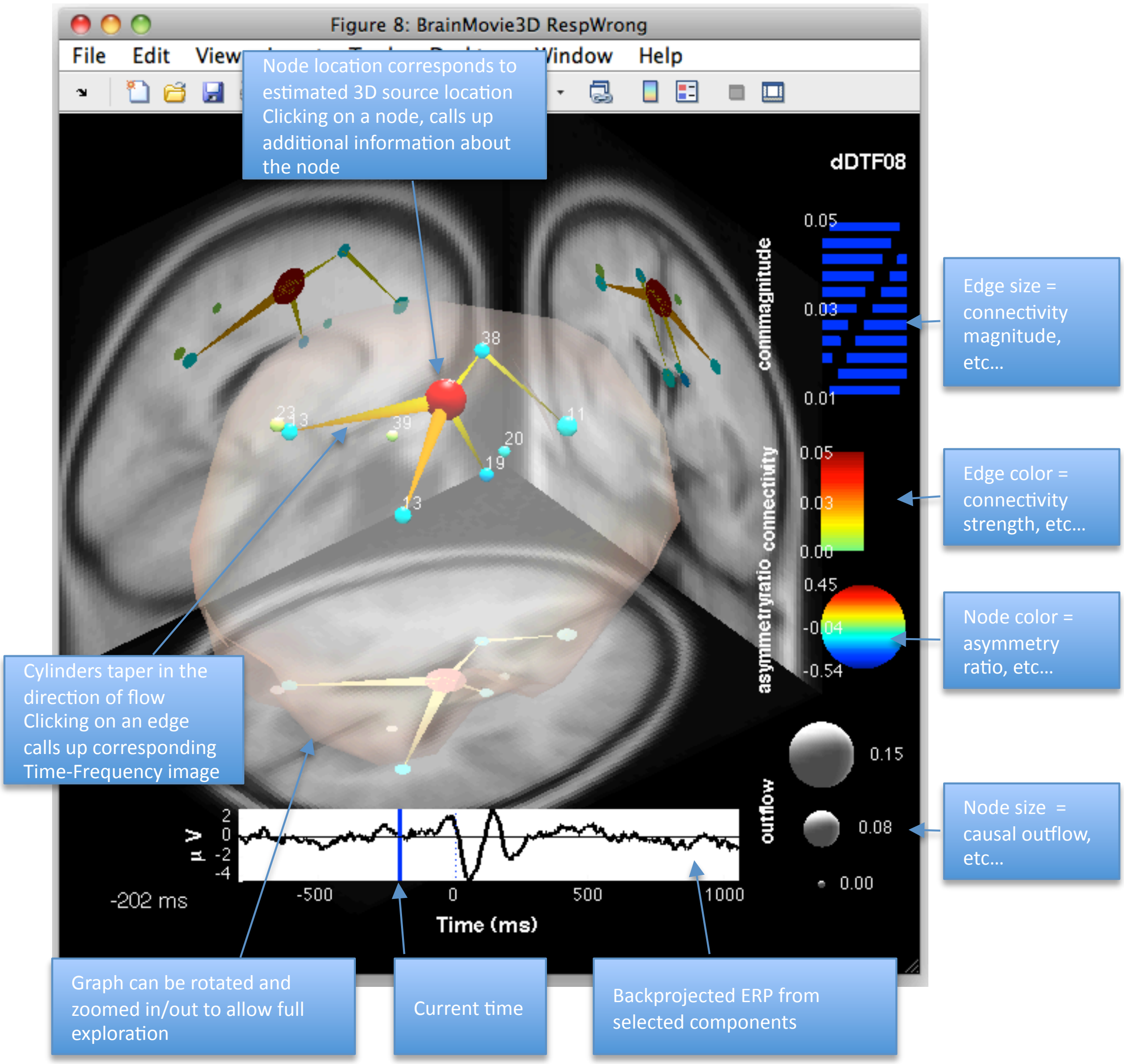
Choose from a number of image and movie output formats

Apply statistical thresholding to the graph

Interactive browse (and save) frames from the movie

When ready, make your movie!

Figure 24. The Interactive BrainMovie3D Control Panel.



Node location corresponds to estimated 3D source location Clicking on a node, calls up additional information about the node

Edge size = connectivity magnitude, etc…

Edge color = connectivity strength, etc…

Node color = asymmetry ratio, etc…

Node size = causal outflow, etc…

Cylinders taper in the direction of flow Clicking on an edge calls up corresponding Time-Frequency image

Graph can be rotated and zoomed in/out to allow full exploration

Current time

Backprojected ERP from selected components

Figure 25. A frame of the interactive BrainMovie3D at -0.2 seconds (-200 ms) relative to the event.

When you are ready, specify an output folder and format under **OutputFormat-→ImageOutputDirectory** and click **Make Movie!** All frames of the movie will now be rendered and saved to disk. This may take a while so you might want to pull out that Yoga mat again (you can also choose a narrower **MovieTimeRange** if you don't want to wait around). If you selected **BrainMovieOption→Visibility = *On*** then you should see each frame rendered on your display. Setting visibility to off will replace the on-screen rendering with a progress bar, which should speed up the movie-making process.

Now that we've made our movie, let's take a look at some of our frames. Figure 26 shows three of these frames, corresponding to the start (-523 ms), middle (40 ms), and end (606 ms) of our button-press task. Note that at the start of the epoch, the network is initially quiescent, with some weak communication between sources in or near anterior rostral ACC (RCZa; IC 11) and supplementary motor area (SMA/preSMA; IC 38).

Moving to the time just following the button-press event (center frame) we see that now IC8, located in posterior ACC (RCZp/CCZ), has become a central causal hub, exerting significant influence on several areas of the network, but particularly posterior parietal cortex (IC13) and RCZa. There is some bidirectional flow, but the flux is largely outward from IC8, as indicated by the red hue of the node (indicating large positive asymmetry ratio). Note that this corresponds precisely to the negative peak of the ERN. However, we *are not* modeling dependencies in the event-locked ERN itself (which is an ERP and subtracted during ensemble normalization) but rather in the ongoing oscillations *underlying* the ERN complex.

Moving to the end of the epoch, around 606 ms, we see that the network has almost returned to its initial decoupled state, and examining the last frames of the movie will reveal the complete decoupling of IC8 from the rest of the network. This panel seems to implicate RCZp/CCZ as some sort of causal hub in a cortical network for error-processing. As noted in section 6.9.1. this is entirely consistent with the theoretical (and partly experimentally verified) role of RCZp/CCZ in error processing.

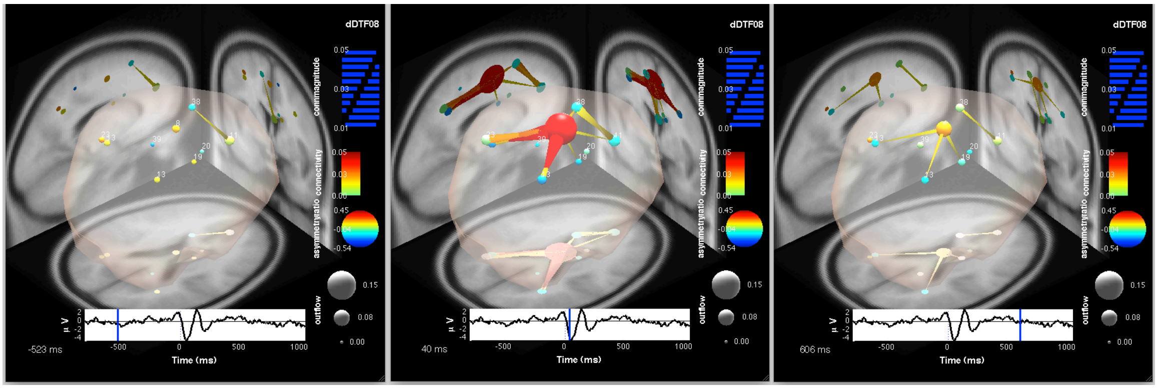


Figure 26. Three frames of a causal BrainMovie3D showing transient theta information flow during error commission. The frames are correspond to -523 ms (left), 40 ms (center), and 606 ms (right) relative to the button press (0 ms).

It is left as an exercise to the reader to do the following:

1. Try creating brainmovies for other frequency bands (e.g., delta). What is different between the evolution of the delta-band cortical network and the theta-band network? You can even have brainmovie find the peaks over some frequency range (e.g., 2-9 Hz) and map the peak frequency onto edge color to color-code different frequency-specific sub-networks (Hint: examine the FreqCollapseMethod and EdgeColorMapping properties).
2. Select both *RespWrong* and *RespCorr* datasets and create TimeFrequencyGrid images and BrainMovies for the between-condition differences (if more than one datasets is selected TimeFrequencyGrid and BrainMovie3D automatically assume you want to examine the between-condition difference. Is there more theta-band information flow from RCZp during error commission than during correct button presses? What about the delta band?

6.9.3. Causal Projection

When we are interested in visualizing the anatomical distribution of univariate measures such as ERSP and multiple coherence, as well as graph-theoretic measures such as causal flow and asymmetry ratio, an alternative to creating a brainmovie is create a Causal Projection (CP) image (Mullen et al, 2010a). CP is based on the dipole density concept proposed by Delorme and Makeig in 2003. The basic assumption behind CP is that each dipole located at coordinate c = [x y z] has some spatial localization uncertainty which we will (naively) approximate by a three-dimensional Gaussian distribution with mean $\mu=c$ and covariance Σ. Furthermore, each dipole has some functional measure (spectral modulation, information outflow, etc) associated with its source process. We can then compute the probability of observing a dipole at a given voxel, weighted by the amplitude of the specified measure to obtain an estimate of the likelihood of observing a source in a particular brain location, and that source having a significant value for the measure of interest.

Informally, the CP at a given voxel is a weighted sum of gaussian-projected distances to all (neighboring) dipoles, each weighted by the amplitude of the specified measure (causal outflow, ERSP, etc) for the source associated with that dipole.

Formally, if w_i is the measure associated with the ith source, c_i is the coordinate of the ith dipole location and Σ_i is the covariance matrix of the corresponding Gaussian (reflecting the uncertainty in localization and/or inter-subject variability) then for a voxel with 3-dimensional coordinate v we have that:

$$CP(v) = \frac{\sum_{i=1}^{M} w_i \, g(v, c_i, \Sigma_i)}{Z}$$

where Z is an optional normalization term to ensure a total probability mass of 1 over the brain volume and

$$g(v, \mu, \Sigma) = \frac{1}{(2\pi)^{3/2} |\Sigma|^{1/2}} \exp\left(-\tfrac{1}{2}(v-\mu)'\Sigma^{-1}(v-\mu)\right)$$

is the multivariate gaussian p.d.f. evaluated at v .

This measure can easily be generalized to multi-subject datasets by simply combining all dipoles, from all subjects in the same MNI volumetric space. Figure 27 shows the result of applying Causal Projection to a cohort of 26 subjects performing the two-back task described earlier. Here we have projected the delta-theta band outflows and inflows from each source that exhibited a significant difference between *RespWrong* and *RespCorr* conditions. Note the causal source hub in RCZp and sinks in RCZa and PPC following erroneous button presses. We can simultaneously plot two measures (e.g., outflow and inflow) or two conditions using gamma-corrected color mapping (lower panel). In this manner red might represent increases in the first measure (outflow), and green represents increases in second measure (inflow) with the sum of the two color (yellow) representing no difference between measures (balanced flow) and transparency representing non-significance (no flow).

Causal projection is currently being revamped for inclusion in SIFT 0.1b **(pop_vis_causalProjection())**.

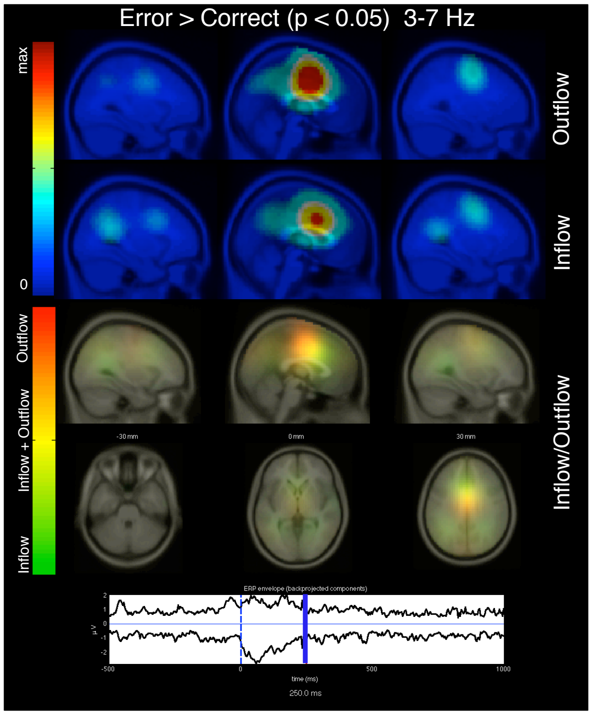


Figure 27. A frame (250 ms following button press) of a Causal Projection movie computed across a cohort of 26 subjects performing the two-back CP task. Here we have projected the delta-theta outflows and inflows from each source that exhibited a significant difference between *RespWrong* and *RespCorr* conditions. Note the causal source hub in RCZp and sinks in RCZa and PPC.

6.10. Group Analysis

Cognitive experiments are typically carried out across a cohort of participants and it is useful to be able to characterize difference in brain network activity within and between groups of individuals for different conditions. The Group Analysis module in SIFT, currently under development, will afford several routines for assessing group-level connectivity networks with confidence intervals. While such analysis is relatively simple when performing analyses on scalp channels, it become more complicated when estimating connectivity in the source domain between dipolar IC processes. This is primarily due to the fact that it is often difficult to equate IC sources between participants. While we typically utilize clustering techniques to help equate dipolar sources across participants, in some cases a subset of participants will still not exhibit one or more sources approximately observed in all other participants. If one does not take into account these *missing variables*, one may risk obtaining biased estimates of average connectivity across the subject population. This missing variable problem is well-known in statistics, and several approaches have been proposed for dealing with this. Currently, group analysis in the source domain is implemented using two methods, disjoint clustering, which does not take into account the missing variable problem but may still be useful for a general analysis, particularly if there is high agreement across the cohort of datasets in terms of source location, and a Bayesian mixture model approach which provides more robust statistics across datasets. A brief description of these methods is provided below, and a more detailed description will be included with the beta release of SIFT.

6.10.1. Disjoint Clustering

This approach adopts a 3-stage process:

1. Identify K ROI's (clusters) by affinity clustering of sources across subject population using EEGLAB's Measure-Product clustering.

2. Average all incoming and outgoing individually statistically significant connections between each pair of ROIs to create a [K X K [x freq x time]] group connectivity matrix.

3. Visualize the results using any of SIFTs visualization routines. This method suffers from low statistical power when subjects do not have high agreement in terms of source locations (missing variable problem).

6.10.2. Bayesian Mixture Model

A more robust approach (in development with Wes Thompson and to be released in SIFT 1.0b) uses smoothing splines and Monte-Carlo methods for joint estimation of posterior probability (with confidence intervals) of cluster centroid location and between-cluster connectivity. This method takes into account the "missing variable" problem inherent to the disjoint clustering approach and provides robust group connectivity statistics.

7. Conclusions and Future Work

In this paper we have introduced a new, open-source (Matlab-based) toolbox for electrophysiological information flow analysis which functions as a plugin for the EEGLAB environment. We sought to outline the theoretical basis for vector autoregressive (VAR) model fitting of electrophysiological data as well as some VAR-based measures for multivariate granger-causality and spectral analysis in the time and frequency domain. We then demonstrated the applicability of these approaches through simulations and, using the SIFT toolbox, the analysis of EEG data from an individual performing an error-inducing cognitive task.

Although the current release of SIFT is ***alpha*** (and therefore lacking important several features which are currently in development, such as group analysis and integrated statistics), SIFT 0.1-beta, scheduled for released in January, will contain these features and more.

8. Acknowledgements

I would like to first express my deep appreciation to Arnaud Delorme and Christian Kothe who have helped in the development of this toolbox. Dr. Delorme is the principal developer of EEGLAB and has helped substantially with integrating the toolbox into the EEGLAB environment as well as with modifications of his brainmovie3d.m and dipoledensity.m functions on which the causal brainmovie and causal projection functions have been based. Mr. Kothe contributed to many conversations regarding the structure of the toolbox and contributed the function input/output specification and PropertyGrid code, which is used in some of the graphical user interfaces.

I would also like to thank Scott Makeig for his constant encouragement and intellectual contributions in developing this toolbox. Additionally, I'd like to thank my undergraduate/postgraduate advisor Dr. Robert Knight (UC Berkeley) who supported my development of the ECViz toolbox on which the concept of SIFT was based.

Thanks also goes to Virginia De Sa and to Doug Nitz for serving as my project committee members and for their constant patience throughout the development of this project.

Finally, a big thanks goes to Nima Bigdely Shamlo, Julie Onton, Thorsten Zander and others from SCCN and elsewhere who have contributed ideas, datasets, visualization code, and other useful items to this project.

The author (Tim Mullen) is generously supported by a San Diego Fellowship, a Glushko Fellowship (Dept. of Cognitive Science), and endowments from the Swartz Foundation (Old Field, NY).

Parametric model-fitting in SIFT makes use of modified routines from Alois Schloegl's open-source TSA+NaN package and, if downloaded separately, Tapio Schneider and Arnold Neumaier's ARfit package.

9. Appendix I: SIFT 0.1a Function Reference

The table below is a partial reference for SIFT 0.1a functions. Not all functions are documented in this list.

	Function Name	Description
GUI functions	pop_pre_prepData	generate GUI for data preprocessing
	pop_est_fitMVAR	generate GUI for VAR/AMVAR model fitting
	pop_est_selModelOrder	generate GUI for VAR model order selection
	pop_est_validateMVAR	generate GUI for VAR model validation
	pop_est_mvarConnectivity	generate GUI for computing connectivity measures
	pop_vis_TimeFreqGrid	generate GUI for Interactive Time-Frequency Grid
	pop_vis_causalBrainMovie3D	generate GUI for Interactive Causal BrainMovie3D
Preprocessing	pre_detrend	linearly detrend or center an ensemble of data
	pre_diffData	apply a difference filter to an ensemble of data
	pre_normData	apply temporal or ensemble normalization to an ensemble of data
	pre_prepData	preprocess an ensemble of data (calls other subfunctions)
	pre_selectComps	select a set of independent components from the data
Modeling	est_calcInvCovMat	compute inverse covariance matrix of a VAR process
	est_calcInvCovMatFourier	compute frequency-domain transformation of the inverse covariance matrix of a VAR process
	est_calcInvCovMatFourierPDC	same as above, but a specific version used for analytic PDC significance thresholds
	est_checkMVARConsistency	check the percent consistency of a fitted VAR model
	est_checkMVARParams	perform a sanity check on a set of specified MVAR parameters – return recommendations on optimal parameters, if relevant.
	est_checkMVARStability	check the stability/stationarity of fitted VAR model
	est_checkMVARWhiteness	check the whiteness of the residuals of a fitted VAR model
	est_eigenmode	return the eigenmodes of a VAR process (requires ARFIT package)
	est_fitMVAR	fit a VAR[p] model to the data using one of several algorithms (Vieira-Morf, ARFIT, MLS, etc). Optionally can use a sliding window to perform segmentation-based adaptive MVAR analysis. Calls modified routines from Alois Schloegl's open-source TSA package or from the ARFIT package.
	est_fitMVARKalman	fit a VAR[p] model to continuous data using a Kalman

		filter. Adapts code from Alois Schloegl’s open-source TSA package
	est_MVARConnectivity	compute spectral density, coherence, and connectivity measures from a fitted VAR model
	est_mvarResiduals	return the residuals of a fitted VAR model
	est_mvtransfer	compute frequency-domain quantities from a VAR model (spectrum, coherence, granger-causality, etc)
	est_selModelOrder	evaluate and return model order selection criteria (AIC, SBC, FPE, HQ) for a range of model orders
Statistics	stat_bootSignificanceTests	perform bootstrap significance tests on connectivity structure
	stat_analyticSignificanceTests	perform asymptotic analysis significance tests on connectivity structure
	stat_phaserand	return a distribution satisfying the null hypothesis of no connectivity using a phase-randomization approach (Theiler, 1997)
	stat_bootstrap	return a bootstrapped distribution of a spectral/connectivity estimator
	stat_prctileTest	perform one- or two-sided percentile tests to compare an observed value with the quantiles of a (null) distribution.
Visualization	vis_TimeFreqGrid	low-level function to create an interactive Time-Frequency Grid
	vis_TimeFreqCell	low-level function to render an expanded (detailed) version of a single cell of the Time-Frequency Grid
	vis_causalBrainMovie3D	low-level function to generate a causal BrainMovie3D
	vis_causalProjection	in development – low-level function to generate a Causal Projection image or movie
Simulations	sim_genVARModelFromEq	generate an arsim()-compatible VAR specification from a text-based equation
Helpers	hlp_*	A large number of helper functions (to be documented later)

10. References

Anderson BDO, Moore JB (1979) Optimal Filtering. Englewood Cliff, NJ: Prentice-Hall.

Arranz MA Portmanteau Test Statistics in Time Series. packages.tol-project.org Available at: http://packages.tol-project.org/docs/ndmtest.pdf.

Astolfi L, Cincotti F, Mattia D, Marciani MG, Baccala L a, Vico Fallani F de, Salinari S, Ursino M, Zavaglia M, Ding L, Edgar JC, Miller G a, He B, Babiloni F (2007) Comparison of different cortical connectivity estimators for high-resolution EEG recordings. Human brain mapping 28:143-57

Baccalá LA, Sameshima K (2001) Partial directed coherence: a new concept in neural structure determination. Biological cybernetics 84:463-74

Baccalá LA, Sameshima K (2007) Generalized partial directed coherence In Digital Signal Processing, 2007 15th International Conference on IEEE, p. 163–166.

Barone P (1987) A method for generating independent realizations of a multivariate normal stationary and invertible ARMA(p,q) process. J. Statist. Comput. Simulat. 8:273-83

Bell AJ, Sejnowski TJ (1995) An information-maximization approach to blind separation and blind deconvolution. Neural computation 7:1129–1159

Benjamini Y, Hochberg Y (1995) Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the Royal Statistical Society. Series B (Methodological) 57:289–300

Blinowska K, Kaminski M (2006) Multivariate Signal Analysis by Parametric Models In B. Schelter, M. Winterhalder, & J. Timmer, eds. Handbook of Time Series Analysis Wiley, Wienheim.

Bressler SL, Richter CG, Chen Y, Ding M (2007) Cortical functional network organization from autoregressive modeling of local field potential oscillations. Statistics in medicine 26:3875–3885

Bressler SL, Seth AK (2010) Wiener-Granger Causality: A well established methodology. NeuroImage

Brillinger DR (2001) Time series: data analysis and theory. SIAM.

Bullmore E, Sporns O (2009) Complex brain networks: graph theoretical analysis of structural and functional systems. Nature reviews. Neuroscience 10:186-98

Burg JP (1967) Maximum entropy spectral analysis In 37th Ann. Int. Meet., Soc. Explor.Geophys. Oklahoma City, OK, USA.

Burg JP (1975) Maximum entropy spectral analysis. Stanford, CA, USA: Stanford University Press.

Buzsaki G (2006) Rhythms of the Brain. Oxford University Press, USA.

Chatfield C (1989) The Analysis of Time Series: An Introduction 4th ed. Chapman & Hall.

Chen Y, Bressler SL, Ding M (2006) Frequency decomposition of conditional Granger causality and application to multivariate neural field potential data. Journal of neuroscience methods 150:228-37

Deshpande G, LaConte S, James GA, Peltier S, Hu X (2009)(a) Multivariate Granger causality analysis of fMRI data. Human brain mapping 30:1361-73

Deshpande G, Sathian K, Hu X (2009)(b) Effect of hemodynamic variability on Granger causality analysis of fMRI. NeuroImage

Dhamala M, Rangarajan G, Ding M (2008) Analyzing information flow in brain networks with nonparametric Granger causality. NeuroImage 41:354-62

Ding MZ, Bressler SL, Yang WM, Liang HL (2000)(a) Short-window spectral analysis of cortical event-related potentials by adaptive multivariate autoregressive modeling: Data preprocessing, model validation, and variability assessment. Biological Cybernetics 83:35-45

Ding M, Bressler SL, Yang W, Liang H (2000)(b) Short-window spectral analysis of cortical event-related potentials by adaptive multivariate autoregressive modeling: data preprocessing, model validation, and variability assessment. Biol. Cybern. 83:35-45

Ding M, Chen Y, Bressler SL (2006) Granger causality: Basic theory and application to neuroscience In B. Schelter, M. Winterhalder, & J. Timmer, eds. Handbook of Time Series Analysis Wiley, Wienheim.

Efron B, Tibshirani RJ (1994) An Introduction to the Bootstrap: Monographs on Statistics & Applied Probability. Chapman and Hall/CRC.

Eichler M (2006)(a) Graphical Modeling of Dynamic Relationships in Multivariate Time Series In B. Schelter, M. Winterhalder, & J. Timmer, eds. Handbook of Time Series Analysis Wiley, Wienheim.

Eichler M (2006)(b) On the evaluation of information flow in multivariate systems by the directed transfer function. Biological cybernetics 94:469-82

Fitzgibbon SP, Powers DMW, Pope KJ, Clark CR (2007) Removal of EEG noise and artifact using blind source separation. Journal of clinical neurophysiology : official publication of the American Electroencephalographic Society 24:232-43

Florian G, Pfurtscheller G (1995) Dynamic spectral analysis of event-related EEG data. Electroencephalography and clinical neurophysiology 95:393–396

Florin E, Gross J, Pfeifer J, Fink GR, Timmermann L (2010) The effect of filtering on Granger causality based multivariate causality measures. NeuroImage 50:577-88

Geweke J (1982) Measurement of linear dependence and feedback between multiple time series. Journal of the American Statistical Association 77:304–313

Granger CWJ (1969) Investigating causal relations by econometric models and cross-spectral methods. Econometrica: Journal of the Econometric Society 37:424-438

Haufe S, Tomioka R, Nolte G (2010) Modeling sparse connectivity between underlying brain sources for EEG/MEG. Biomedical Engineering:1-10

Holroyd CB, Coles MGH (2002) The Neural Basis of Human Error Processing: Reinforcement Learning, Dopamine, and the Error-Related Negativity. Psychological Review 109:679 - 709

Hui HB, Leahy RM (2006) Linearly Constrained MEG Beamformers for MVAR Modeling of Cortical Interactions In 3rd IEEE International Symposium on Biomedical Imaging: Macro to Nano IEEE, p. 237-240.

Jansen BH, Bourne JR, Ward JW (1981) Autoregressive estimation of short segment spectra for computerized EEG analysis. IEEE transactions on bio-medical engineering 28:630-8

Kaminski M (1997) Multichannel Data Analysis in Biomedical Research In Understanding Complex Systems, Handbook of Brain Connectivity series Berlin/Heidelberg: Springer, p. 327-355.

Kaminski M, Ding M, Truccolo WA, Bressler SL (2001) Evaluating causal relations in neural systems: Granger causality, directed transfer function and statistical assessment of significance. Biological Cybernetics 85:145–157

Kaminski M, Blinowska K (1991) A New Method of the description of the information flow in the brain structures. Biological Cybernetics 65:203–210

Kenet T, Arieli A, Tsodyks M, Grinvald A (2005) Are Single Cortical Neurons Soloists or Are They Obedient Members of a Huge Orchestra? In J. L. van Hemmen & T. J. Sejnowski, eds. 23 Problems in Systems Neuroscience Oxford University Press, USA.

Korzeniewska A (2003) Determination of information flow direction among brain structures by a modified directed transfer function (dDTF) method. Journal of Neuroscience Methods 125:195-207

Korzeniewska A, Crainiceanu M, Kus R, Franaszczuk P, Crone N (2008) Dynamics of Event-Related Causality in Brain Electrical Activity. Human brain mapping 29:1170–1192

Luu P, Tucker DM, Makeig S (2004) Frontal midline theta and the error-related negativity: neurophysiological mechanisms of action regulation. Clinical neurophysiology : official journal of the International Federation of Clinical Neurophysiology 115:1821-35

Lütkepohl H (2006) New Introduction to Multiple Time Series Analysis. Berlin, Germany: Springer.

Marple S (1987) Digital Spectral Analysis with Applications. Englewood Cliff, NJ: Prentice Hall.

Michel CM, Murray MM, Lantz G, Gonzalez S, Spinelli L, Grave De Peralta R (2004) EEG source imaging. Clinical Neurophysiology 115:2195–2222

Mognon A, Jovicich J, Bruzzone L, Buiatti M (2010) ADJUST: An automatic EEG artifact detector based on the joint use of spatial and temporal features. Psychophysiology:1-12

Mullen, T., Delorme, A., Kothe, C., Makeig, S (2010) An Electrophysiological Information Flow Toolbox for EEGLAB. Society for Neuroscience 2010, San Diego, CA

Neumaier A, Schneider T (2001) Estimation of parameters and eigenmodes of multivariate autoregressive models. ACM Transactions on Mathematical Software (TOMS) 27:27–57

Nolte G, Bai O, Wheaton L, Mari Z, Vorbach S, Hallett M (2004) Identifying true brain interaction from EEG data using the imaginary part of coherency. Clinical neurophysiology : official journal of the International Federation of Clinical Neurophysiology 115:2292-307

Onton J, Makeig S (2009) High-frequency Broadband Modulations of Electroencephalographic Spectra. Frontiers in human neuroscience 3:61

Pearl J (2000) Causality: Models, Reasoning, and Inference. Cambridge University Press.

Pearl J (2009) Causality: Models, Reasoning and Inference 2nd ed. New York, New York, USA: Cambridge University Press.

Pereda E, Quiroga RQ, Bhattacharya J (2005) Nonlinear multivariate analysis of neurophysiological signals. Progress in neurobiology 77:1-37

Roger C, Bénar CG, Vidal F, Hasbroucq T, Burle B (2010) Rostral Cingulate Zone and correct response monitoring: ICA and source localization evidences for the unicity of correct- and error-negativities. NeuroImage 51:391-403

Schelter B, Winterhalder M, Eichler M, Peifer M, Hellwig B, Guschlbauer B, Lucking CH, Dahlhaus R, Timmer J (2005)(a) Testing for directed influences among neural signals using partial directed coherence. J. Neurosci. Methods 152:210-219

Schelter B, Winterhalder M, Timmer J eds. (2006) Handbook of Time Series Analysis: Recent Theoretical Developments and Applications 1st ed. Wiley.

Schelter B, Timmer J, Eichler M (2009) Assessing the strength of directed influences among neural signals using renormalized partial directed coherence. Journal of neuroscience methods 179:121-30

Schelter B, Winterhalder M, Eichler M, Peifer M, Hellwig B, Guschlbauer B, Lücking CH, Dahlhaus R, Timmer J (2005)(b) Testing for directed influences among neural signals using partial directed coherence. Journal of neuroscience methods 152:210-9

Schlögl A (2000) The electroencephalogram and the adaptive autoregressive model: theory and applications. Doctoral Thesis.

Schlögl A (2006) A comparison of multivariate autoregressive estimators. Signal processing 86:2426-2429

Schlögl A, Supp G (2006) Analyzing event-related EEG data with multivariate autoregressive parameters. Progress in brain research 159:135–147

Schneider T, Neumaier A (2001) Algorithm 808: ARfit---a matlab package for the estimation of parameters and eigenmodes of multivariate autoregressive models. ACM Transactions on Mathematical Software 27:58-65

Seth AK (2005) Causal connectivity of evolved neural networks during behavior. Network (Bristol, England) 16:35-54

Seth AK (2010) A MATLAB toolbox for Granger causal connectivity analysis. Journal of neuroscience methods 186:262-73

Sommerlade L, Henschel K, Wohlmuth J, Jachan M, Amtage F, Hellwig B, Lucking CH, Timmer J, Schelter B (2009) Time-variant estimation of directed influences during Parkinsonian tremor. Journal of Physiology-Paris

Supp GG, Schlögl A, Trujillo-Barreto N, Muller MM, Gruber T (2007) Directed Cortical Information Flow during Human Object Recognition: Analyzing Induced EEG Gamma-Band Responses in Brain's Source Space. PLoS One 2:684

Theiler J (1992) Testing for nonlinearity in time series: the method of surrogate data. Physica D: Nonlinear Phenomena 58:77-94

Valdés-Sosa P a, Sánchez-Bornot JM, Lage-Castellanos A, Vega-Hernández M, Bosch-Bayard J, Melie-García L, Canales-Rodríguez E (2005) Estimating brain functional connectivity with sparse multivariate autoregression. Philosophical transactions of the Royal Society of London. Series B, Biological sciences 360:969-81

Wang X, Chen Y, Bressler SL, Ding M (2007) Granger Causality Between Multiple Interdependent Neurobiological Time Series: Blockwise Versus Pairwise Methods. International Journal of Neural Systems 17:71

Wang X, Chen Y, Ding M (2008) Estimating Granger causality after stimulus onset: a cautionary note. NeuroImage 41:767-76

Weiner N (1956) The Theory of Prediction In E. F. Beckenbach, ed. Modern Mathematics for Engineers New York, New York, USA: McGraw-Hill.

Yordanova J, Falkenstein M, Hohnsbein J, Kolev V (2004) Parallel systems of error processing in the brain. NeuroImage 22:590-602

Zetterberg LH (1969) Estimation of Parameters for a Linear Difference Equation with Application to EEG Analysis. Mathematical Biosciences 5:227–275