Red Lemurs F17 - S18 | **Neuro Data Design**
Dr. Joshua Vogelstein | Multivariate Parametric Time Series Models
TA: Eric Bridgeford | **Due:** Monday, October 23, 2017

1 Purpose

We wish to survey the applicability of common parametric time series models such as the Autoregressive Moving Average (ARMA) model to categorize the behavior of EEG signals of subjects during different states. Specifically, we would like to explore vector ARMA models used for multivariate time series, to capture two main effects within our data:

(a) The multivariate nature of the 128 channels on a given subject, all producing their own time series.

(b) The relationship between multiple subjects, assuming we used a small number (or one) time series to describe each subject.

In familiarizing ourselves with the mathematical models, we also look for software that will accomplish common functions such as Auto-correlation, Partial Auto-correlation, and Cross-correlation, as well as look at previous applications to EEG data. This outline is not meant to replace any reference text, but serve as a list of the main topics to look into when using these tools.

2 Summary of Univariate Time Series

2.1 ARMA Models

Given a time series $(X_1, X_2, ..., X_n)$, which we also write as (X_t), an ARMA model defines a dependence relationship between an observation X_t and the observations before it $(X_{t-1}, X_{t-1}, ..., X_1)$. Sepcifically, an ARMA$(p, q)$ model suggests the following relationship:

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + ... + \phi_p X_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + ... + \theta_q \epsilon_{t-q}$$

Where the $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$ represent the error terms at timestep i, and $\phi_1, ..., \phi_p, \theta_1, ..., \theta_q$ are real numbers. For example, an ARMA(2,1) process would look like:

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \epsilon_t + \theta_1 \epsilon_{t-1}$$

Here there are a few properties of the time series that are important for consideration. The first is **stationarity**. If the time series is stationary, then its statistical properties do not change over time (its conditional properties may change, however). We require the ϕ coefficients to be less than 1 for stationarity to hold. Additionally, we assume the time series to have zero mean, and if not can correct the series (subtract $\bar{X}$) to achieve that mean. Finally, the term Auto-regressive Integrated Moving Average (ARIMA) model is also used, which describe non-stationary time series whose mean seems to grow linearly, quadratically, or some higher order polynomial. We can use ARMA and ARIMA interchangably, because given an ARIMA(p, d, q) where d is the order of the mean function, we can "differentiate" the time series d times to achieve an ARMA(p, q) and perform the same analysis. (By differentiating or differencing, we define a new time series $Y_t = \frac{X_t - X_{t-1}}{t-(t-1)} = X_t - X_{t-1}$. This is the "derivative" of the time series).

2.2 The ACF and PACF Functions

The Auto-correlation (ACF) function $\rho(h)$ describes the correlation of the series' values h steps apart. Because the time series has zero mean, we can find the Auto-covariance function (ACVF), $\gamma(h)$ using the following process.

$$\begin{aligned}\gamma(h) &= \mathrm{Cov}[X_t, X_{t-h}] = \mathrm{E}[X_t X_{t-h}] - \mathrm{E}[X_t] \cdot \mathrm{E}[X_{t-h}] \\ \mathrm{E}[X_t] &= \mathrm{E}[X_{t-h}] = 0 \\ \gamma(h) &= \mathrm{E}[X_t X_{t-h}] \\ \rho(h) &= \frac{\gamma(h)}{\gamma(0)} = \mathrm{Corr}[X_t, X_{t-h}]\end{aligned}$$

By multiplying the ARMA expansion of a time series by X_{t-h}, and taking the expectation, we can get closed-form values for the ACF function. The PACF is taken by calculating the correlation of series values without the linear effects of the values in between. The table below summarizes how the sample plots of these functions for a given time series can be used to learn the p and q parameters for certain time series.

Table 1. ACF and PACF properties of ARMA models

	AR (p)	**MA (p)**	**ARMA(p,q)**
Autocorrelation function (ACF)	Infinite damped and or damped sine waves. Tails off.	Finite; cuts if after q lags	Infinite; damped exponentials and or damped sine waves. Tails off.
Partial autocorrelation function (PCF)	Finite; cuts of after p lags	Infinite; damped exponentials and or damped sine waves. Tails off	Infinite; damped exponentials and or damped sine waves. Tails off.

Note that these functions are symmetric, or $\gamma(h) = \gamma(-h)$ and $\rho(h) = \rho(-h)$.

2.3 Forecasting

We can also forecast these time series using the conditional expectation of a new observation on past data, or $\mathrm{E}[X_{n+1}|X_n, X_{n-1}, ..., X_1]$. Again using the ARMA(2,1) case, we have:

$$\begin{aligned}X_t &= \phi_1 X_{t-1} + \phi_2 X_{t-2} + \epsilon_t + \theta_1 \epsilon_{t-1} \\ \mathrm{E}[X_{n+1}|X_n, X_{n-1}, ..., X_1] &= \mathrm{E}[\phi_1 X_n + \phi_2 X_{n-1} + \epsilon_{n+1} + \theta_1 \epsilon_n | X_n, X_{n-1}, ..., X_1] \\ &= \phi_1 X_n + \phi_2 X_{n-1} + \theta_1 \epsilon_n\end{aligned}$$

This may not be of huge relevance to the EEG data, but is a common tool used for many different times of time series.

3 Multivariate Generalization

3.1 Model Complexity

In the multivariate case, we have a similar time series, denoted $(X^{(t)})$, but with each observation in m-dimensions. That is:

$$X^{(t)} = \begin{bmatrix} X_1^{(t)} \\ X_2^{(t)} \\ \vdots \\ X_m^{(t)} \end{bmatrix}$$

We maintain the same ARMA(p, q) model, but with new dimensionality.

$$\begin{aligned} X^{(t)} &= \phi^{(1)} X^{(t-1)} + \phi^{(2)} X^{(t-2)} + ... + \phi^{(p)} X^{(t-p)} + \epsilon^{(t)} + \theta^{(1)} \epsilon^{(t-1)} + ... + \theta^{(q)} \epsilon^{(t-q)} \\ \epsilon^{(i)} &\in \mathbb{R}^m \\ \phi^{(i)}, \theta^{(i)} &\in \mathbb{R}^{m \times m} \end{aligned}$$

Evidently, the complexity of the model grows immensely, as for each of the $p+q$ linear dependencies, we have to learn a matrix instead of a coefficient. In the case of our EEG channels, we would learn a 128×128 matrix for every dependency, so it would be in our best interest to maintain small values of p and q in applying such a model, and/or simplifying our data with Principal Component Analysis (PCA) or a similar dimensionality-reduction method.

3.2 The CCF Function

Finally, we define the Cross-correlation function, or CCF. This function is NOT symmetric - it defines the correlation between two time series (X_t) and (Y_t) at a lag time h. It is important to specify which series is lagging behind. In our notation, we will subtract the lag time from the first series in the subscript. We will first define the Cross-covariance function and use it to define the CCF:

$$\begin{aligned} \gamma_{XY}(h) &= \mathrm{Cov}[X_{t-h}, Y_t] = \mathrm{E}[X_{t-h} Y_t] \\ \rho_{XY}(h) &= \frac{\gamma_{XY}(h)}{\sqrt{\gamma_{XX}(0)\gamma_{YY}(0)}} = \mathrm{Corr}[X_{t-h}, Y_t] \end{aligned}$$

We have two uses for this. In the case of a particular subject, we can define their Cross-correlation matrix $\Gamma_{\text{channels}}(h)$ with the entry (i, j) containing $\rho_{ij}(h) = \mathrm{Corr}[X_i^{(t-h)}, X_j^{(t)}]$, or the Cross-correlation of channels i and j at lag h. If we had one time series to describe one subject (the average series for example), we could create a Cross-correlation $\Gamma_{\text{subjects}}(h)$ matrix for the subjects in a particular group, with entry (i, j) as $\rho_{XY}(h) = \mathrm{Corr}[X_{t-h}, Y_t]$ being the Cross-correlation between the i-th subject $(X^{(t)})$ and the j-th subject $(Y^{(t)})$, at lag h.

4 Software for Parameter Estimation (and Other Functions)

Because of the highly-dependent nature of the data, parameter estimation algorithms are not extremely intuitive - most are recursive and take time to implement, especially in the multivariate case. Many packages exist for this purpose.

4.1 `R` Packages

1 `stats` is the base statistics package in `R`. It comes with general functionality for fitting ARMA models and plotting series along with their ACF and PACF functions. It contains wrappers for general functions that adjust to time series data.

2 `MTS` is the most used all-purpose multivariate time series package. It contains all the functionality discussed above.

3 More obscure functions and their corresponding implementations van be found here.

4.2 `Python` Packages

1 `nsim` has efficient parallel computing functions for computations on multiple time series.

2 `columnts` presents a neat data structure for storing single multidimensional time series and performing operations on them.

3 `statmodels` offers another general-purpose statistics API that includes functions specifically for time series.

5 Relevance to EEG Data

5.1 Cabrieto Paper:

Change point detection is crucial in analyzing many time series dataignoring them leads to incorrect conclusions and forecasts. In this context, detection of change points is necessary to understand how a cognitive process unfolds in response to a stimulus. Applies procedure (vector autoregressive - a stochastic process model used to capture the linear interdependencies among multiple time series) to electroencephalograms and demonstrate its potential impact in identifying change points in complex brain processes during a cognitive motor task.

5.2 Quantitative EEG Analysis Textbook

Multivariate measures are used to consider the interactions between the channels and how they correlate rather than looking at each individually. Bivariate measures (simple synchronization and lag synchronization) only look at correlation of two channels. Simple synchronization counts the number of times event pairs between two time series have occurred over a period time. Lag synchronization finds correlation between two time series that are identical except for that one of them is offset by some time lag. Other methods use PCA to reduce dimensionality in order to take all channels into account. Finding these correlations could lead to effective ways for predicting seizures.