# Workshop on Introductory Time Series Analysis

Mick Cooney
mickcooney@gmail.com

Sometime in 2016

## 1. Basic Concepts

Time series occur in almost any field of study that produces quantitative data. Whenever quantities are measured over time, those measurements form a time-series, or more formally, a *discrete-time stochastic process.*

One reasonably famous example of a time-series is count of airline passengers in the US, as seen in Figure 1. This is a fairly simple time-series, with measurements taken on a monthly basis over a number of years, with each datum consisting of a single number, i.e. this time-series is *univariate.*

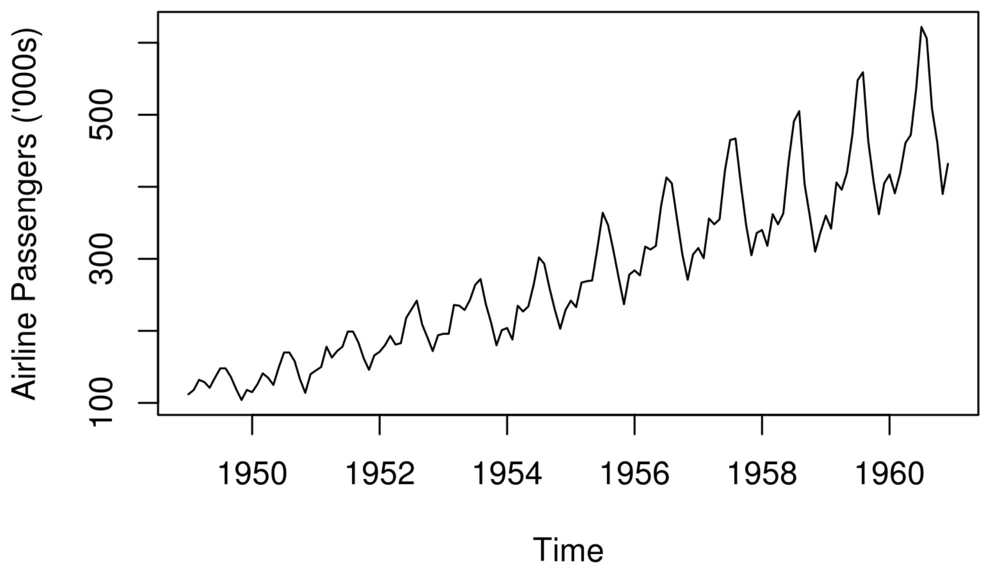

Figure 1: Example of a Time Series: Monthly Airline Passengers in the US

Before we begin trying to analyse data such as this, we need to first create a mathematical framework to work in. Fortunately, we do not need anything too complicated, and for a finite time-series of length $N$, we model the time series as a sequence of $N$ random variables, $X_i$, with $i = 1, 2, ..., N$.

Realise that each individual $X_i$ is a wholly separate random variable — analysing time series statistically is unusual as we only ever have a single measurement for each random variable from which we can do inference. In many cases we simplify this much further, but it is important to understand and appreciate that such simplifications are just that, and this is often the reason why time series can be very difficult to analyse.

Before we get to any of that though, and before we try to build any kind of models for the data, we always start with visualising the data. Often, a simple plot of the data helps use pick out aspects to analyse and incorporate into the models. For time series, one of the first things to do is the *time plot*, a simple plot of the data over time.

For the passenger data, a few aspects stand out that are very common in time series. It is apparent that the numbers increase over time, and this systematic change in the data is called the *trend.* Often, approximating the trend as a linear function of time is adequate for many data sets.

A repeating pattern in the data that occurs over the period of the data (in this case, each year), is called the *seasonal variation*, though a more general concept of 'season' is implied — it often will not coincide with the seasons of the calendar.

A slightly more generalised concept from the seasonality is that of *cycles*, repeating patterns in the data that do not correspond to the natural fixed periods of the model. None of these are apparent in the air passenger data, and accounting for them are beyond the scope of this introductory tutorial.

Finally, another important benefit of visualising the data is that it helps identify possible *outliers* and *erroneous* data.

**Exercise 1.1** Load the air passengers data into your workspace and investigate the structure of the `ts` object using `str()`. How is a `ts` object different from a standard vector in R? Plot it using the default plot method.

**Exercise 1.2** Using the data supplied in the file `Maine.dat` and the function `read.table()`, load the Maine unemployment data into your workspace and repeat the tasks above.

**Exercise 1.3** Analyse the trend and seasonality for the air passenger data by using the `aggregate()` and `cycle()` functions. Create a boxplot for the data, segmenting the data by month.

**Exercise 1.4** Repeat the above analysis for Maine unemployment data.

**Exercise 1.5** Calculate the average monthly data for each of the above time series. Compare this to the actual monthly data and plot them together. What can we learn from this?

**Exercise 1.6** Using the `window()` function, calculate quantitative values for the above.

## 2. Multivariate Time Series

In many cases, we will also be dealing with time series that have multiple values at all, many or some of the points in time.

Often, these values will be related in some ways, and we will want to analyse those relationships also. In fact, one of the most efficient methods of prediction is to find *leading indicators* for the value or values you wish to predict — you can often use the current values of the leading indicators to make inference on future values of the related quantities.

The fact that this is one of the best methods in time series analysis says a lot about the difficulty of prediction (Yogi Berra, a US baseball player noted for his pithy statements, once said "Prediction is difficult, especially about the future").

**Exercise 2.1** Load in the multivariate data from the file `cbe.dat`. Investigate the object type and some sample data to get an idea of how it is structured. The R functions `head()` and `tail()` will be of use for this.

**Exercise 2.2** Create time series objects for this data using `ts()`, and plot them beside each other. `cbind()` is useful for creating all the plots together.

**Exercise 2.3** Merge the electricity usage data with the US airline passenger data using `ts.intersect` and investigate any possible similarities between the two time series.

**Exercise 2.4** Use the `cor()` function, investigate the correlation between the two time series. How plausible is a causal effect in this case?

## 3. Time Series Decomposition

Since many time series are dominated by trends or seasonal effects, and we can create fairly simple models based on these two components. The first of these, the *additive decompositional model*, is just the sum of these effects, with the residual component being treated as random:

$$x_t = m_t + s_t + z_t, \tag{1}$$

where, at any given time $t$, $x_t$ is the observed value, $m_t$ is trend, $s_t$ is the seasonal component, and $z_t$ is the error term.

It is worth noting that, in general, the error terms will be a correlated sequence of values, something we will account for and model later.

In other cases, we could have a situation where the seasonal effect increases as the trend increases, modeling the values as:

$$x_t = m_t s_t + z_t. \tag{2}$$

Other options also exist, such as modeling the log of the observed values, which does cause some non-trivial modeling issues, such as biasing any predicted values for the time series.

Various methods are used for estimating the trend, such as taking a *moving average* of the values, which is a common approach.

**Exercise 3.1** Using the `decompose()` function in R, look at the trend and the seasonal variation for the airline passenger data. The output of this function can be plotted directly, and visually check the output. Does the output match your intuition about what you observed?

**Exercise 3.2** Repeat this process for the CBE dataset.

**Exercise 3.3** Try a multiplicative model for all of the above. `decompose()` allows the selection of this via the '`type`' parameter. Is the multiplicative model better? In either case, explain why this might be.

**Exercise 3.4** Repeat the above, but use the `stl()` R function instead of `decompose()`. Compare the output of the two.

## 4. Autocorrelation

Assuming we can remove the trend and the seasonal variation, that still leaves the random component, $z_t$. Unfortunately, analysing this is usually highly non-trivial. As discussed, we model the random component as a sequence of random variables, but no further assumptions we made.

To simplify the analysis, we often make assumptions like *independent and identically distributed (i.i.d.)* random variables, but this will rarely work well. Most of the time, the $z_t$ are correlated.

The *expected value* or *expectation* of a random variable $x$, denoted $E(x)$, is the mean value of $x$ in the population. So, for a continuous $x$, we have

$$\mu = E(x) = \int p(x)\, x\, dx. \tag{3}$$

and the *variance*, $\sigma^2$, is the expectation of the squared deviations,

$$\sigma^2 = E[(x-\mu)^2], \tag{4}$$

For bivariate data, each datapoint can be represented as $(x, y)$ and we can generalise this concept to the *covariance*, $\gamma(x, y)$,

$$\gamma(x, y) = E[(x-\mu_x)(y-\mu_y)]. \tag{5}$$

Correlation, $\rho$, is the standardised covariance, dividing the covariance by the standard deviation of the two variables,

$$\rho(x, y) = \frac{\gamma(x, y)}{\sigma_x \sigma_y}. \tag{6}$$

The mean function of a time series model is

$$\mu(t) = E(x_t), \tag{7}$$

with the expectation in this case being across the *ensemble* of possible time series that might have been produced by this model. Of course, in many cases, we only have one realisation of the model, and so, without any further assumption, estimate the mean to be the measured value.

If the mean function is constant, we say that the time-series is *stationary in the mean*, and the estimate of the population mean is just the sample mean,

$$\mu = \sum_{t=1}^{n} x_t. \tag{8}$$

The variance function of a time-series model that is stationary in the mean is given by

$$\sigma^2(t) = E[(x_t - \mu)^2], \tag{9}$$

and if we make the further assumption that the time-series is also stationary in the variance, then the population variance is just the sample variance

$$\mathrm{Var}(x) = \frac{\sum (x_t - \mu)^2}{n-1} \tag{10}$$

Autocorrelation, often referred to as *serial correlation*, is the correlation between the random variables at different time intervals. We can define the *autocovariance function* and the *autocorrelation function* as functions of the *lag*, $k$, as

$$\begin{aligned}\gamma_k &= E[(x_t - \mu)(x_{t+k} - \mu)], & (11)\\ \rho_k &= \frac{\gamma_k}{\sigma^2}. & (12)\end{aligned}$$

Be default, the `acf()` function plots the *correlogram*, which is a plot of the sample autocorrelation at $r_k$ against the lag $k$.

**Exercise 4.1** Using the function `acf()`, calculate the autocorrelations for all the time series we have looked at. Look at the structure of the output, and use the help system to see what options are provided.

**Exercise 4.2** Check the output of `acf()` against manual calculations of the correlations at various timesteps. Do the numbers match?
**HINT:** The `cor()` function and some vector indexing will be helpful here.

**Exercise 4.3** Plot the output of the `acf()` for the different time series. Think about what these plots are telling you. Do do these plots help the modelling process, if so, how?

**Exercise 4.4** Decompose the air passenger data and look at the appropriate correlogram. What does this plot tell you? How does it differ from the previous correlogram you looked at?

**Exercise 4.5** How can we use all that we have learned so far to assess the efficacy of the decompositional approach for time series?

## 5. Basic Forecasting

As mentioned earlier, an efficient way to forecast a variable is to find a related variable whose value leads it by one or more timesteps. The closer the relationship and the longer the lead time, the better it becomes.

The trick, of course, is to find a leading variable.

Multivariate series has a temporal equivalent to correlation and covariance, known as the *cross-covariance function (ccvf)* and the *cross-correlation function (ccf)*,

$$\gamma_k(x,y) = E[(x_{t+k}-\mu_x)(y_t-\mu_y)], \tag{13}$$

$$\rho_k(x,y) = \frac{\gamma_k(x,y)}{\sigma_x\sigma_y}. \tag{14}$$

Note that the above functions are not symmetric, as the lag is always on the first variable, $x$.

**Exercise 5.1** Load the building approvals and activity data from the `ApprovActiv.dat` file. The data is quarterly and starts in 1996. Determine which is the leading variable and investigate the relationship between the two.

**Exercise 5.2** Binding the time-series using `ts.union()`, find the cross-correlations for the building data. Is the relationship symmetric, and why?

**Exercise 5.3** Examine the cross-correlations of the random element of the decomposed time-series for the building data, and compare this to the original cross-correlations.

Our main objective in forecasting is to estimate the value of a future quantity, $x_{n+k}$, given past values $x_1, x_2, ..., x_n$. We assume no seasonal or trend effects, or any such effects have been removed from the data. We assume that the underlying mean of the data is $\mu_t$, and that this value changes from timestep to timestep, but this change is random.

Our model can be expressed as

$$x_t = \mu_t + w_t, \tag{15}$$

where $\mu_t$ is the non-stationary mean of the process at time $t$ and $w_t$ are independent random variates with mean 0 and standard deviation $\sigma$. We let $a_t$ be our estimate of $\mu_t$, and can define the *exponentially-weighted moving average (EWMA)*, $a_t$ to be

$$a_t = \alpha x_t + (1-\alpha)a_{t-1}, \quad 0 \le \alpha \le 1. \tag{16}$$

The value of $\alpha$ controls the amount of smoothing, as is referred to as the *smoothing parameter.*

**Exercise 5.4** Load the data in the `motororg.dat` file. This is a count of complaints received on a monthly basis by a motoring organisation from 1996 to 1999. Create an appropriate time series from this data. Plot the data, checking it for trends or seasonality.

**Exercise 5.5** Using the function `HoltWinters()`, with the additional parameters set to zero, create the EWMA of the data, allowing the function itself to choose the optimal value of $\alpha$. Investigate and visualise the output, comparing it to the original time series.

**Exercise 5.6** Specifying values of $\alpha$ of 0.2 and 0.9, create new versions of the EWMA and compare them with previous fits of the EWMA.

The Holt-Winters method generalises this concept, allowing for trends and seasonal effects. The equations that govern this model for seasonal period, $p$, are given by

$$\begin{aligned} a_t &= \alpha(x_t - s_{t-p}) + (1-\alpha)(a_{t-1} - b_{t-1}), \\ b_t &= \beta(a_t - a_{t-1}) + (1-\beta)b_{t-1}, \\ s_t &= \gamma(x_t - a_t) + (1-\gamma)s_{t-p}, \end{aligned} \tag{17}$$

where $a_t$, $b_t$, $s_t$ are the estimated level, slope and seasonal effect at time $t$, and $\alpha$, $\beta$ and $\gamma$ are the smoothing parameters.

**Exercise 5.7** Fit the Holt-Winters parameters to the air passenger data and check the fit. Visualise the raw time-series against the fitted data.

**Exercise 5.8** Predict data ahead for four years and visualise this data. How reliable are these forecasts do you think?

## 6. Stochastic Methods and Regression

A time series $w_t$ is *discrete white noise* if the $w_t$ are i.i.d with a mean of zero. Thus, they all have the same variance $\sigma^2$ and $\mathrm{Cor}(w_i, w_j) = 0$ for $i \neq j$. In addition, if the $w_j \sim N(0, \sigma^2)$ then it is said to be *Gaussian white noise.*

A time series $x_t$ is a *random walk* if

$$x_t = x_{t-1} + w_t, \tag{18}$$

where $w_t$ is a white-noise series.

**Exercise 6.1** Generate a white noise series using `rnorm()`, with an initial value, $w_0 = 1.$ and length 100. Plot the output, and investigate its correlogram.

**Exercise 6.2** Generate a random walk time series with initial value $x_0 = 1$ and length 100. Plot its output and investigate its correlogram.

**Exercise 6.3** Think about how you might create a random walk with an underlying drift?

The time series $x_t$ is a *auto-regressive process of order p*, AR($p$), if,

$$x_t = \sum_{i=1}^{p} \alpha_i x_{t-i} + w_t, \tag{19}$$

where $w_t$ is a white-noise process and the $\alpha_p \neq 0$ for an order-$p$ process.

**Exercise 6.4** Generate data for an AR(1) model with $\alpha = 0.5$ and initial value $x_1 = 1$. Plot the data and investigate its correlogram. Does this time series appear to be stationary? The R function `arima.sim()` can be used for this.

**Exercise 6.5** Generate data for an AR(2) model with $\alpha_1 = 1$ and $\alpha_2 = -0.25$ and initial value $x_1 = 1$, $x_2 = 1$. Plot the data and investigate its correlogram. Does this time series appear to be stationary?

**Exercise 6.6** Generate data for an AR(2) model with $\alpha_1 = 0.5$ and $\alpha_2 = 0.5$ and initial value $x_1 = 1$, $x_2 = 1$.

**Exercise 6.7** Fit the time-series you generated above to an autoregressive model using the function `ar()`. How do the fitted parameters match the values you used?

A moving average (MA) process of order $q$ is a linear combination of the current white noise term and the $q$ most recent past white noise terms,

$$x_t = w_t + \sum_{i=1}^{q} \beta_i w_{t-i} \tag{20}$$

where $w_t$ is a white-noise process with mean 0 and variance $\sigma^2$.

**Exercise 6.8** Generate data for an MA(1) model with $\beta = 0.5$ and initial value $x_1 = 1$. Plot the data and investigate its correlogram. Does this time series appear to be stationary?

**Exercise 6.9** Generate data for an MA(2) model with $\beta_1 = 1$ and $\beta_2 = -0.25$ and initial values $x_1 = 1$, $x_2 = 2$. Plot the data and investigate its correlogram. Does this time series appear to be stationary?

**Exercise 6.10** Generate data for an MA(2) model with $\beta_1 = 0.5$ and $\beta_2 = 0.5$ and initial values $x_1 = 1$, $x_2 = 1$.

**Exercise 6.11** Fit the time-series you generated above to a moving-average model using the function `arima()`. How do the fitted parameters match the values you used?

**Exercise 6.12** Compare the AR and MA models to one another.

## 7. ARMA and ARIMA Models

Now suppose we combine the ideas of autoregressive and moving average models together. A time series follows an *autoregressive moving average (ARMA)* process of order $(p, q)$ when

$$x_t = \sum_{i=1}^{p} \alpha_i x_{t-i} + w_t + \sum_{j=1}^{q} \beta_j w_{t-j} \tag{21}$$

where $w_t$ is white noise.

Both AR($p$) and MA($q$) models are special cases of ARMA($p, q$) (with $q = 0$ and $p = 0$ respectively), and ARMA models are usually preferred due to *parameter parsimony* — when fitting data, the ARMA model is usually more parameter efficient, requiring few parameters.

**Exercise 7.1** Using the R function `arima.sim()`, create an ARMA(1,1) time-series of length 1000 with $\alpha = -0.6$, and $\beta = 0.5$. Plot this time series and check its ACF. Is this correct?

**Exercise 7.2** Using `arima()`, fit your generated time series to an ARMA$(1, 1)$ model and compare the fitted output to the values you have set.

**Exercise 7.3** Repeat the above exercises, but for an ARMA$(2, 2)$ model with $\alpha_1 = 0.2$, $\alpha_2 = -0.5$, $\beta_1 = -0.1$ and $\beta_2 = 0.3$.

**Exercise 7.4** Investigate the effect of the parameters $p$ and $q$ by generating the various combinations of the ARMA$(p, q)$ models but using the same set of innovations in each case.
**HINT:** `arima.sim()` has a parameter `innov = ...` that allows you to pass in a set of innovations into the ARMA process.

**Exercise 7.5** Load in the GBP/NZD currency pair data from the file `pound_nz.dat`, and create a time-series from this. The data is quarterly, and starts in Q1 1991.

**Exercise 7.6** Fit the data GBP/NZD to an MA(1), an AR(1) and an ARMA$(1, 1)$ process. Which of the above models does the best job at fitting the data?

It may be becoming quickly apparent that the choice of parameter count is non-trivial, and some kind of systematic approach would be desirable.

One such method for choosing the optimal number of parameters is to fit multiple options and choose the best one, using a metric known as the *Akaike Information Criterion (AIC)*. The AIC is defined to be

$$\text{AIC} = -2 \times \text{loglikelihood of fit} + 2 \times \text{parameter count}, \tag{22}$$

so that it balances the better fitting of parameters while penalising using too many parameters to fit.

**Exercise 7.7** Use the R function `AIC` to calculate the AIC of the three models above. Which one is the best? Why is this question a trap?

**Exercise 7.8** Using all of the various techniques in this workshop, try to model the electricity time series data from the CBE dataset.