Time Series Analysis

R Seminar 3/18/2013
a little bit about time series analysis

Time series are ubiquitous in science and in daily life. The signature of an earthquake wave, the population dynamics of anchovies, or a year of air temperatures—all of these data sets represent repeated measures across time. Analyzing these data sets harbors a unique challenge: measures taken at times close to one another will be more similar than those taken at distant times. This "temporal autocorrelation" means that time series violate assumptions of conventional statistics, which require data points to be independent and identically distributed (iid).

For the purposes of our R package "ASTSA," a time series is a dataset comprised of measures taken at regular time intervals, and represent a theoretical continuous distribution through time.[1] The development of the analysis of time series emerged from several fields, most notably economics and physics, so much of the terminology used when describing and analyzing them is laden with the jargon from those fields.[2]

In this session, I have developed a question-based worksheet based on several examples taken from "Time Series Analysis and Its Application with R Examples."[3] I've separated this exercise into three parts:

1. What does my time series look like?
2. Do I see a trend, or is that just white noise?
3. Is this time series related to that time series?

These are the three questions I chose to address today—but there is really SO MUCH you can do with time series data, and some real fancy stuff you can do with non-linear patterns in data, analysis of seasonality and frequency, and so on.

This session is meant to be more of an exploration of time series and time series analysis, and less of an exercise in R programming—though I do hope you learn or solidify some basic programming skills along the way.

1. What does my time series look like?

Plotting time series is easy with R and astsa.

Let's consider the dataset available with your ASTSA package: global temperature deviations over time. We're going to be working with it a lot.

```
require(astsa)
data(gtemp)
plot(gtemp, type="o", ylab="Global Temperature Deviation")
```

Well that's neat. But does R count this dataset as a true "time series" object (abbreviated by "ts")?

---

[1] The "regular time intervals" part is important—samples taken at uneven time intervals count as a "survey," and the analysis of them as a true time series doesn't fly.
[2] And is therefore also terrifying.
[3] http://www.stat.pitt.edu/stoffer/tsa3/

How about we just ask R:

`is.ts(gtemp)`

Ok, that's settled. But what does that mean? Let's ask R. Type in `?ts` to take a look at the features of time series objects.

As you'll see, ts() is a function just like data.frame(), that can be used to create or coerce data into time series form. Unlike regular data.frame objects, time series have

1. a start and end point in time
2. a frequency—number of observations in a given unit of time. The default is 1, but you might want to change that if you have, say, monthly data (frequency = 12) or seasonal data (frequency = 4).
3. some other stuff that's not so important today.

You could make a time series out of a regular data frame if you want. For example, I wanted to make this long-term dataset (1990-2009) of thaw depth from Alaska into a time series object:

	juldate	year	meanthaw	sdthaw	sethaw	nthaw	minthaw	maxthaw	cvthaw
1	1990183	1990	18.2	4.0	0.3	182	10.0	41.0	21.7
2	1990220	1990	36.2	9.2	0.7	192	11.0	61.0	25.5
3	1991183	1991	17.0	5.4	0.4	198	10.0	34.0	31.9
4	1991220	1991	28.4	6.5	0.5	196	17.0	58.0	23.0
5	1992184	1992	18.5	5.9	0.4	192	10.0	47.5	31.8
6	1992221	1992	40.3	11.5	0.8	192	18.0	84.0	28.5
7	1993183	1993	24.2	8.8	0.6	191	10.0	57.0	36.4
8	1993220	1993	46.1	11.0	0.8	190	28.0	100.0	23.8

Unfortunately, the sampling dates are not evenly spaced. Oh well, let's roll with it. All I had to do was:

```
thaw<-read.csv('thaw.csv', header=TRUE) # from my own data
tsthaw<-ts(thaw, frequency=2)#because there are two observations per year
```

And voila! Ready for time series analysis, sort of.

You might be looking at your `gtemp` graph and thinking, man, that's harsh. I wish I could smooth it out a bit, and make the trends easier to see. Well hey! You can do just that. Type in `?filter` to your console to find out how to apply a moving average to the time series to smooth it out.

In this case, try this code to obtain a three-point moving average. It averages forward AND backwards (2 sides) and repeats the filter 1/3 time for each set of three points (rep…).

```
filtergtemp<-filter(gtemp, sides=2, rep(1, 3)/3)
plot(filtergtemp)
```

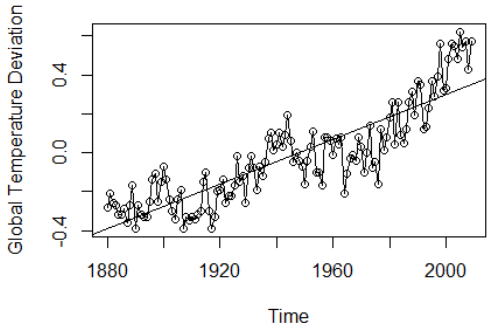
What code would you use to obtain a ten-point moving average? See `?rep` for hints and try it. When might you only want a one-sided moving average?

2. Do I see a trend, or is this just white noise?

You might be noticing something with this gtemp dataset. Temperature deviations go up and down over time, but they seem to be going more upwards than downwards. Let's test that, since we all know that climate change is a conspiracy imposed upon us by tree-hugging liberals with goatees and this is probably an anomaly.

First, you can do the usual thing by doing a standard linear regression on your gtemp plot.

```
summary(fit <- lm(gtemp~time(gtemp)))  # regress gtemp on time
abline(fit)  # add regression line to the plot
```



As we can see from the graph and the regression output, there's clearly a *trend* at work here. Sometimes the trend masks more interesting patterns in your dataset, and it's worth *de-trending* your dataset and examining the residuals. We can plot the residuals like so:

```
fit = lm(gtemp~time(gtemp), na.action=NULL) # regress gtemp on time
par(mfrow=c(2,1)) #prepare the graphics area, a two-high by one-wide setup
plot(resid(fit), type="o", main="detrended")#plot the leftovers from the regression
```

Now, we can see the random-seeming ups and downs of global temperature deviations apart from the upward trend. That wasn't so helpful. Let's try something else. Sometimes, instead of *de-trending*, you might want to *difference* your data. Differencing is basically the process whereby you subtract the value at time *t-1* from the value at time *t*[4]. Differencing allows you to examine the degree of stochasticity of your data. Let's plot the differenced data next to the detrended data and look at it:

```
plot(diff(gtemp), type="o", main="first difference")
```

The differenced data plot shows a pretty bouncy dataset. It seems fair to conclude that global temperature behaves like a "random walk" with some definite upward "drift." How much upward drift? Well, we could just take the mean of the differences to find out:

One last thing we can check for is the autocorrelation of the time series. Remember, nearer points in time are by default more related than distant things, messing up the iid clause of some statistical

[4] Technically, this is called the "first difference."

analyses. If we *got rid of* that autocorrelation, we could perform all sorts of statistical analyses on our data. We can check for temporal autocorrelation with the acf() function in ASTSA.

```
dev.new()#switch to the next graphic
par(mfrow=c(3,1))    # plot ACFs
acf(gtemp, 48, main="gtemp")
acf(resid(fit), 48, main="detrended")
acf(diff(gtemp), 48, main="first difference")
```

Now we have lag (time between points) on the x-axis, and correlation from positive to negative values on the y axis (indicating positive and negative correlation). The untransformed gtemp dataset has a high degree of autocorrelation: at all time lag values, there is a positive correlation. At small lags, there is a higher degree of correlation than at long time lag values.

Now, take a look at the detrended and differenced ACF functions. Can you see why we might wish to use the differenced dataset when performing statistical analyses that require iid?

Of course, there are some other transformations you can use as well—these are the most common.

3. How is this time series related to that time series?

Let's switch gears (and datasets) and examine two other datasests available from the astsa package. The first is the Southern Oscillation Index (related to El Nino cycles) and the second is recruitment of fish (number of first-year fish at a given time).

First, take a look at the data. The datasets are automatically installed when you load astsa and are called `soi` and `rec`, respectively. Use `data()` to call them up. Hint: you could use your graphics parameter setup `par()` to look at both time series at the same time…

What happens when you try to plot a standard linear regression with your two datasets, using soi as x and recruitment as y?

Next, examine the autocorrelation functions for `soi` and `rec`.

Autocorrelation can also be thought of as the match between a future part of the time series and the current time series. If one unit of "lag" corresponds to one month (and each tick mark on the lag axis represents a year), over what periods are these two series positively autocorrelated? Over what periods are they negatively autocorrelated?

Finally, examine the *cross-correlation function* (ccf). Ask R about ccf to find out what it does and how to structure your request (`?ccf`).

You'll notice that lag on the ccf function goes from positive to negative values. A negative lag indicates that the Y value is leading the X value. Positive values of the ccf still correspond to positive correlation—this time, correlation between the series.

Over what periods of time are the series positively correlated? When do they move in the opposite direction?

When do changes in recruitment lead soi? When do changes in soi lead recruitment?