



Published on *STAT 510* (<https://onlinecourses.science.psu.edu/stat510>)

[Home](#) > Lesson 8: Regression with ARIMA errors, Cross correlation functions, and Relationships between 2 Time Series

Lesson 8: Regression with ARIMA errors, Cross correlation functions, and Relationships between 2 Time Series

Assignments:

- Read Pages 30-33 of your text
- Read Examples 2.2 and 2.3 on pages 53-57 and Section 5.6 on pages 293-296 of your text.
- Read through the Lesson 8 online notes that follow.
- Complete Lesson 8 Assignment.

Overview:

This week we'll start coverage of regression between two time series.

Learning Objectives:

After successfully completing this lesson, you should be able to:

- Recognize when and how to adjust for residuals with a time series structure
- Estimate the adjusted intercept and slope
- Interpret the cross-correlation function
- Identify and interpret transfer function models

8.1 Linear Regression Models with Autoregressive Errors

When we do regressions using time series variables, it is common for the errors (residuals) to have a time series structure. This violates the usual assumption of independent errors made in ordinary least squares regression. The consequence is that the estimates of coefficients and their standard errors will be wrong if the time series structure of the errors is ignored.

It is possible, though, to adjust estimated regression coefficients and standard errors when the errors have an AR structure. More generally, we will be able to make adjustments when the errors have a general ARIMA structure.

The Regression Model with AR Errors

Suppose that y_t and x_t are time series variables. A simple linear regression model with autoregressive errors can be written as

$$y_t = \beta_0 + \beta_1 x_t + \epsilon_t$$

with $\epsilon_t = \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + \cdots + w_t$, and $w_t \sim \text{iid } N(0, \sigma^2)$.

If we let $\Phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots$, then we can write the AR model for the errors as

$$\Phi(B)\epsilon_t = w_t.$$

If we assume that an inverse operator, $\Phi^{-1}(B)$, exists, then $\epsilon_t = \Phi^{-1}(B)w_t$.

So, the model can be written

$$y_t = \beta_0 + \beta_1 x_t + \Phi^{-1}(B)w_t,$$

where w_t is the usual white noise series.

Note: This generalizes to the multiple linear regression structure as well. We can have more than one x-variable (time series) on the right side of the equation. Each x-variable is adjusted in the manner described below.

Examining Whether This Model May be Necessary

1. Start by doing an ordinary regression. Store the residuals.
2. Analyze the time series structure of the residuals to determine if they have an AR structure.
3. If the residuals from the ordinary regression appear to have an AR structure, estimate this model and diagnose whether the model is appropriate.

Theory for the Cochrane-Orcutt Procedure

A simple regression model with AR errors can be written as

$$(1) \quad y_t = \beta_0 + \beta_1 x_t + \Phi^{-1}(B)w_t$$

$\Phi(B)$ gives the AR polynomial for the errors.

If we multiply all elements of the equation by $\Phi(B)$, we get

$$\Phi(B)y_t = \Phi(B)\beta_0 + \beta_1 \Phi(B)x_t + w_t$$

$$\text{Let } y_t^* = \Phi(B)y_t = y_t - \phi_1 y_{t-1} - \cdots - \phi_p y_{t-p}$$

$$\text{Let } x_t^* = \Phi(B)x_t = x_t - \phi_1 x_{t-1} - \cdots - \phi_p x_{t-p}$$

$$\text{Let } \beta_0^* = \Phi(B)\beta_0 = (1 - \phi_1 - \dots - \phi_p)\beta_0$$

Note: In the last of the equations just given β_0 is an unknown constant that does not move in time. Thus it is not affected by any backshift operation. That's why the B operations were not applied in that equation.

Using the items just defined, we can write the model as

$$(2) \quad y_t^* = \beta_0^* + \beta_1 x_t^* + w_t$$

Remember that w_t is a white noise series, so this is just the usual simple regression model.

We use results from model (2) to iteratively adjust our estimates of coefficients in model (1).

- The estimated slope $\hat{\beta}_1$ from model (2) will be the adjusted estimate of the slope in model (1) (and its standard error from this model will be correct as well).
- To adjust our estimate of the intercept β_0 in model (1), we use the relationship $\beta_0^* = (1 - \phi_1 - \dots - \phi_p)\beta_0$ to determine that

$$\hat{\beta}_0 = \frac{\hat{\beta}_0^*}{1 - \hat{\phi}_1 - \dots - \hat{\phi}_p}.$$

The standard error for $\hat{\beta}_0$ is

$$s.e.(\hat{\beta}_0) = \frac{s.e.(\hat{\beta}_0^*)}{1 - \hat{\phi}_1 - \dots - \hat{\phi}_p}.$$

This procedure is attributed to Cochrane and Orcutt (1949) and is repeated until the estimates converge, that is we observe a very small difference in our estimates between iterations. When the errors exhibit an AR(1) pattern, the `cochrane.orcutt` function found within the `orcutt` package in R iterates this procedure.

Additional Comment

For a higher order AR, the adjustment variables are calculated in the same manner with more lags. For instance, suppose the residuals were found to have an AR(2) with estimated coefficients 0.9 and -0.2. Then the y - and x - variables for the adjustment regression would be

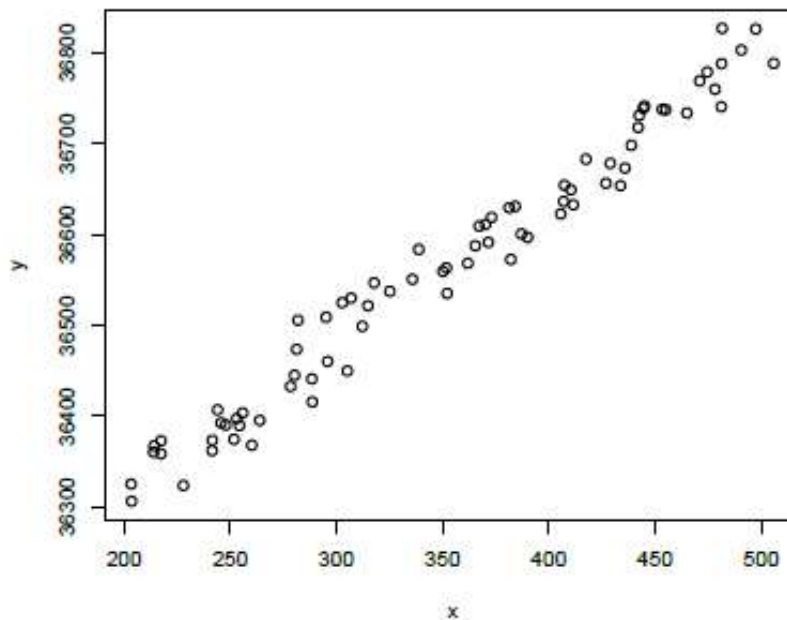
$$y_t^* = y_t - 0.9y_{t-1} + 0.2y_{t-2}$$

$$x_t^* = x_t - 0.9x_{t-1} + 0.2x_{t-2}$$

Example 1: Economic Measure

There are $n = 76$ annual observations in sequence. Suppose x is a leading economic indicator (predictor) for a country and y = a measure of the state of the economy. The

following plot shows the relationship between x and y for 76 years.

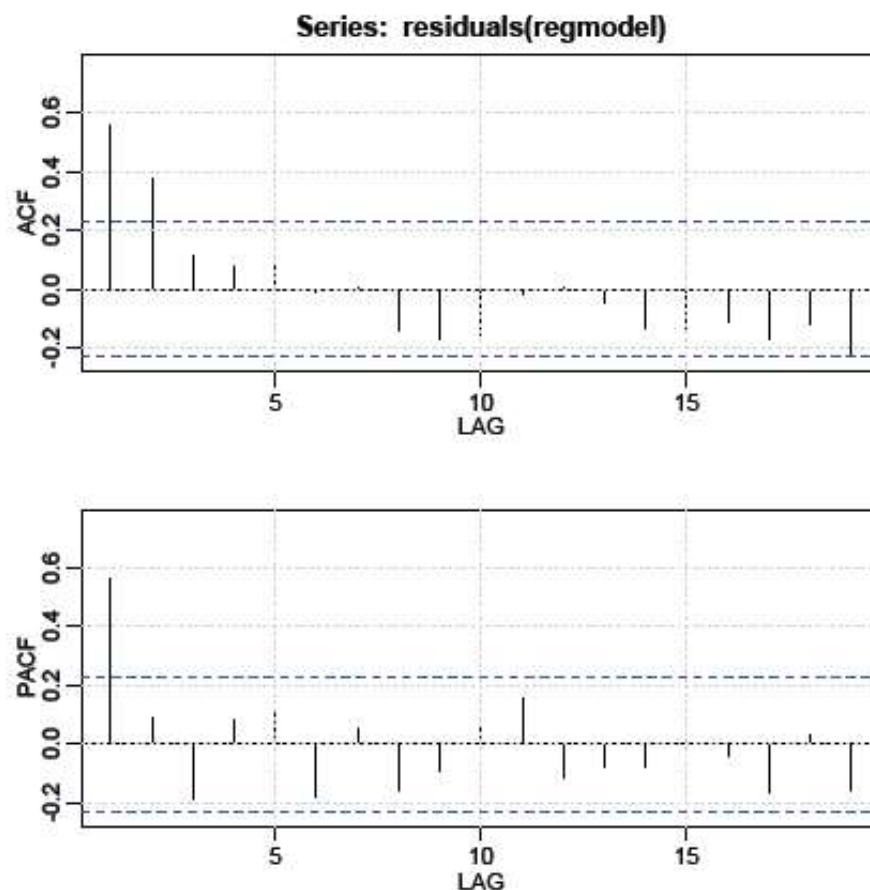


Suppose that we want to estimate the linear regression relationship between y and x at concurrent times.

Step 1: Estimate the usual regression model. Results from R are:

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 3.599e+04  1.167e+01 3082.99  <2e-16 ***
x            1.628e+00  3.222e-02   50.55  <2e-16 ***
```

Step 2: Examine the AR structure of the residuals. Following are the ACF and PACF of the residuals. It looks like the errors from Step 1 have an AR(1) structure.



Step 3: Estimate the AR coefficients (and make sure that the AR model actually fits the residuals). For this example, the R estimate of the AR(1) coefficient is

```
Coefficients:
      ar1
      0.5627
s.e.    0.0943
```

Model diagnostics (not shown here) were okay.

Step 4: Calculate variables to use in the adjustment regression:

$$x_t^* = x_t - 0.5627x_{t-1}$$

$$y_t^* = y_t - 0.5627y_{t-1}$$

Step 5: Use ordinary regression to estimate the model $y_t^* = \beta_0^* + \beta_1 x_t^* + w_t$. For this example, the results are

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.574e+04   9.181e+00  1714.9  <2e-16 ***
xnew         1.592e+00   5.666e-02   28.1   <2e-16 ***
```

The slope estimate (1.592) and its standard error (0.05666) are the adjusted estimates

for the original model.

The adjusted estimate of the intercept of the original model is $15740/(1-0.5627) = 35993.6$. The estimated standard error of the intercept is $9.181/(1-0.5627) = 20.995$.

This procedure is iterated until the estimates converge. In R, the `Cochrane.orcutt` function iterates the steps:

```

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
XB(Intercept) 3.601e+04  2.214e+01  1626.6   <2e-16 ***
XBx           1.585e+00  5.959e-02   26.6   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 20.28 on 73 degrees of freedom
Multiple R-squared:      1,    Adjusted R-squared:      1
F-statistic: 2.042e+07 on 2 and 73 DF,  p-value: < 2.2e-16

$rho
[1] 0.5908156

$number.interaction
[1] 11

```

The slope estimate (1.585) and its standard error (0.05959) are the adjusted estimates for the original model.

Thus our estimated relationship between y_t and x_t is

$$y_t = 36010 + 1.585x_t$$

The errors have the estimated relationship $e_t = 0.5908e_{t-1} + w_t$.

Note that that the predicted y is a linear function of x at this time and the residual at the previous time.

The R Program

```

x=ts(scan("econpredictor.dat"))
y=ts(scan("econmeasure.dat"))
plot.ts(x,y,xy.lines=F,xy.labels=F)

regmodel=lm(y~x) #Step 1
summary(regmodel)
acf2(residuals(regmodel)) #Step 2

arlres = arima (residuals (regmodel), order = c(1,0,0),
include.mean = FALSE) #AR(1) Step 3
sarima (residuals (regmodel), 1,0,0, no.constant = T) #Step 3
xl = ts.intersect(x, lag(x,-1)) # Step 4 Create matrix with x and

```

```

lag 1 x as elements
xnew=x1[,1] - 0.5627*x1[,2] # Step 4 Create x variable for
adjustment regression
yl = ts.intersect(y,lag(y,-1)) # Step 4 Create matrix with y and
lag 1 y as elements
ynew=yl[,1]-0.5627*yl[,2] # Step 4 Create y variable for
adjustment regression

adjustreg = lm(ynew~xnew) # Step 5 Adjustment regression
summary(adjustreg)
acf2(residuals(adjustreg))

install.packages("orcutt")
library(orcutt)
cochrane.orcutt(regmodel)

```

A potential problem with the Cochrane-Orcutt procedure is that the residual sum of squares is not always minimized and therefore we now present a more modern approach following our text.

The Regression Model with ARIMA Errors

Estimating the Coefficients of the Adjusted Regression Model with Maximum Likelihood

The method used here depends upon what program you're using. In R (with *gls* and *arima*) and in SAS (with PROC AUTOREG) it's possible to specify a regression model with errors that have an ARIMA structure. With a package that includes regression and basic time series procedures, it's relatively easy to use an iterative procedure to determine adjusted regression coefficient estimates and their standard errors. Remember, the purpose is to adjust "ordinary" regression estimates for the fact that the residuals have an ARIMA structure.

Carrying out the Procedure

The basic steps are:

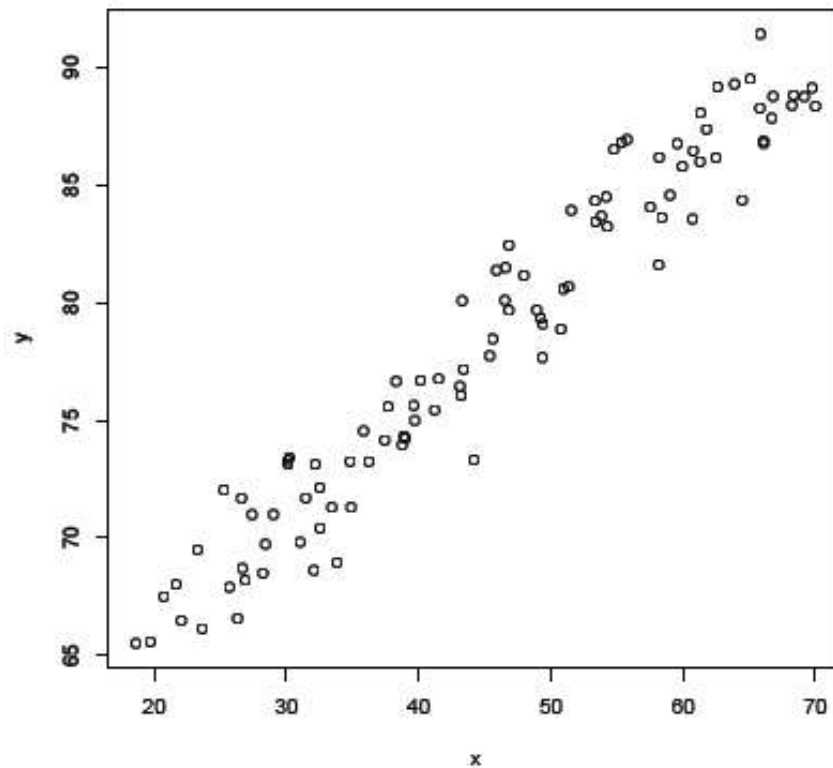
1. Use ordinary least squares regression to estimate the model

$$y_t = \beta_0 + \beta_1 t + \beta_2 x_t + \epsilon_t$$
 - *Note:* We are modeling a potential trend over time with $\beta_0 + \beta_1 t$ and may avoid the need for differencing to de-trend. Please see Section 2.3 of the text for a discussion on de-trending vs. differencing.
2. Examine the ARIMA structure (if any) of the sample residuals from the model in step 1.
3. If the residuals do have an ARIMA structure, use maximum likelihood to simultaneously estimate the regression model using ARIMA estimation for the residuals.
4. Examine the ARIMA structure (if any) of the sample residuals from the model in step 3. If white noise is present, then the model is complete. If not, continue to adjust the ARIMA model for the errors until the residuals are white noise.

Example 2: Simulated

The following plot shows the relationship between a simulated predictor x and response y for

100 annual observations.



Suppose that we want to estimate the linear regression relationship between y and x at concurrent times.

Step 1: Estimate the usual regression model. Results from R are:

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  57.76368    3.84725   15.014  <2e-16 ***
trend         0.03450    0.09416    0.366    0.715
x             0.41798    0.18859    2.216    0.029 *
```

Residual standard error: 1.774 on 97 degrees of freedom
Multiple R-squared: 0.9416, Adjusted R-squared: 0.9404
F-statistic: 782.1 on 2 and 97 DF, p-value: < 2.2e-16

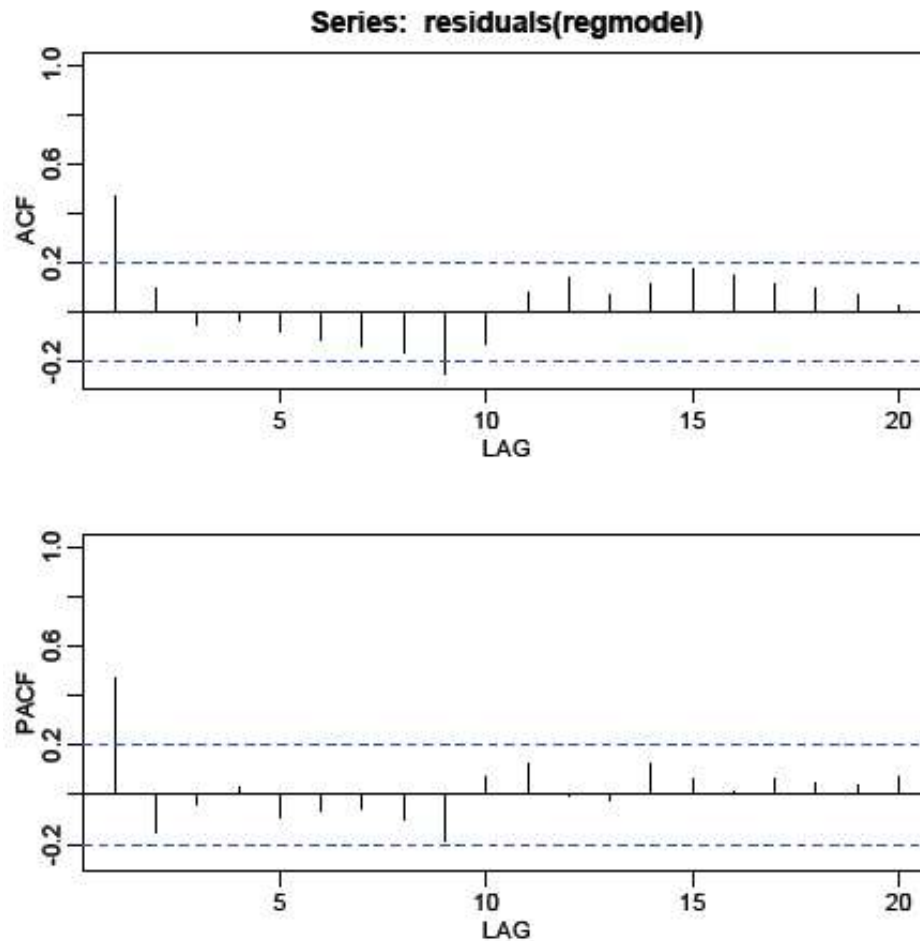
Because trend is not significant, we may drop it from the model:

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  56.37066    0.58468   96.41  <2e-16 ***
x             0.48693    0.01226   39.73  <2e-16 ***
```

Residual standard error: 1.767 on 98 degrees of freedom
Multiple R-squared: 0.9415, Adjusted R-squared: 0.9409
F-statistic: 1578 on 1 and 98 DF, p-value: < 2.2e-16

Step 2: Examine the ARIMA structure of the residuals. Following are the ACF and PACF of the residuals. Because both the ACF and PACF spike and then cut off, we should compare AR(1), MA(1), and ARIMA(1,0,1). We will continue with the MA(1) model in the

notes.



Step 3: Estimate the adjusted model with a MA(1) structure for the residuals (and make sure that the MA model actually fits the residuals). For this example, the R estimate of the model is

```

Coefficients:
      mal  intercept      x
0.4567    56.4107    0.4858
s.e.  0.0755     0.7350    0.0154

sigma^2 estimated as 2.384:  log likelihood = -185.44,  aic = 378.88

```

Step 4: Model diagnostics, (not shown here), suggested that the model fit well.

Thus our estimated relationship between y_t and x_t is

$$y_t = 56.4107 + 0.4858x_t$$

The errors have the estimated relationship $e_t = w_t + 0.4567w_{t-1}$, where $w_t \sim \text{iid } N(0, \sigma^2)$.

The R Program

The data are in two files: l8.1x.dat and l8.1y.dat. They are in the Week 8 folder, so you can

reproduce this if you wish.

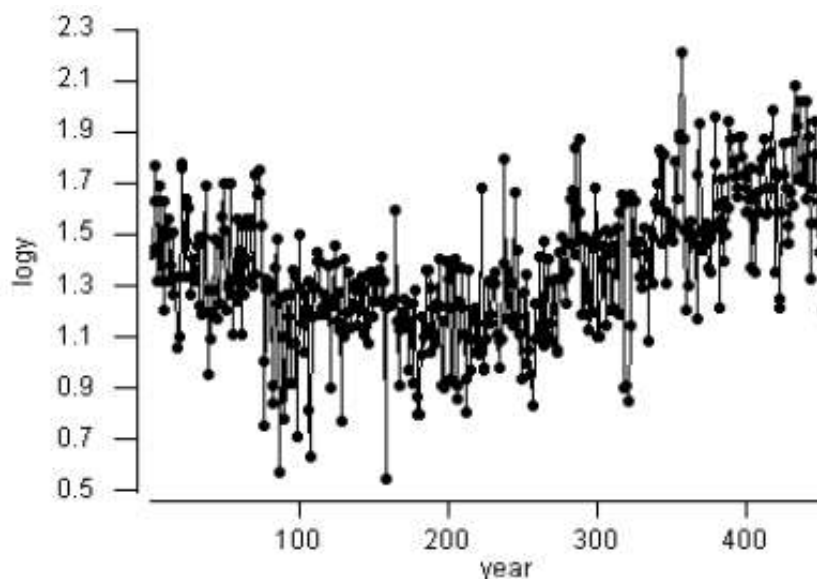
```
library(astsa)
x=ts(scan("l8.1.x.dat"))
y=ts(scan("l8.1.y.dat"))
plot(x,y, pch=20,main = "X versus Y")

trend = time(y)
regmodel=lm(y~trend+x) # Step 1 first ordinary regression.
regmodel=lm(y~x) # Step 1 first ordinary regression without
trend.
summary(regmodel) # This gives us the regression results
acf2(resid(regmodel)) # Step 2 ACF and PACF of the residuals

adjreg = sarima (y, 0,0,1, xreg=cbind(trend,x)) #Step 3 This is
the adjustment regression with MA(1)
residuals
adjreg = sarima (y, 0,0,1, xreg=x) #Step 3 Drop insignificant
trend
adjreg #Step 4 Results of adjustment regression. White noise
should be suggested
```

Example 3: Glacial Varve

Note that in this example it might work better to use an ARIMA model as we have a univariate time series, but we'll use the approach of these notes for illustrative purposes. We analyze the glacial varve data described in Example 2.5, page 62 of the text. The response is a measure of the thickness of deposits of sand and silt (varve) left by spring melting of glaciers about 11,800 years ago. The data are annual estimates of varve thickness at a location in Massachusetts for 455 years beginning 11,834 years ago. There are nonconstant variance and outlier problems in the data, so we might try a log transformation with hopes of stabilizing the variance and diminishing the effects of outliers. Here's a time series plot of the log10 series.



There's a curvilinear pattern, so we'll try the ordinary regression model

$$\log_{10}y = \beta_0 + \beta_1(t - \bar{t}) + \beta_2(t - \bar{t})^2 + \epsilon,$$

where t = year numbered 1, 2, ...455.

Centering the time variable creates uncorrelated estimates of the linear and quadratic terms in the model. Regression results found using R are:

Coefficients:

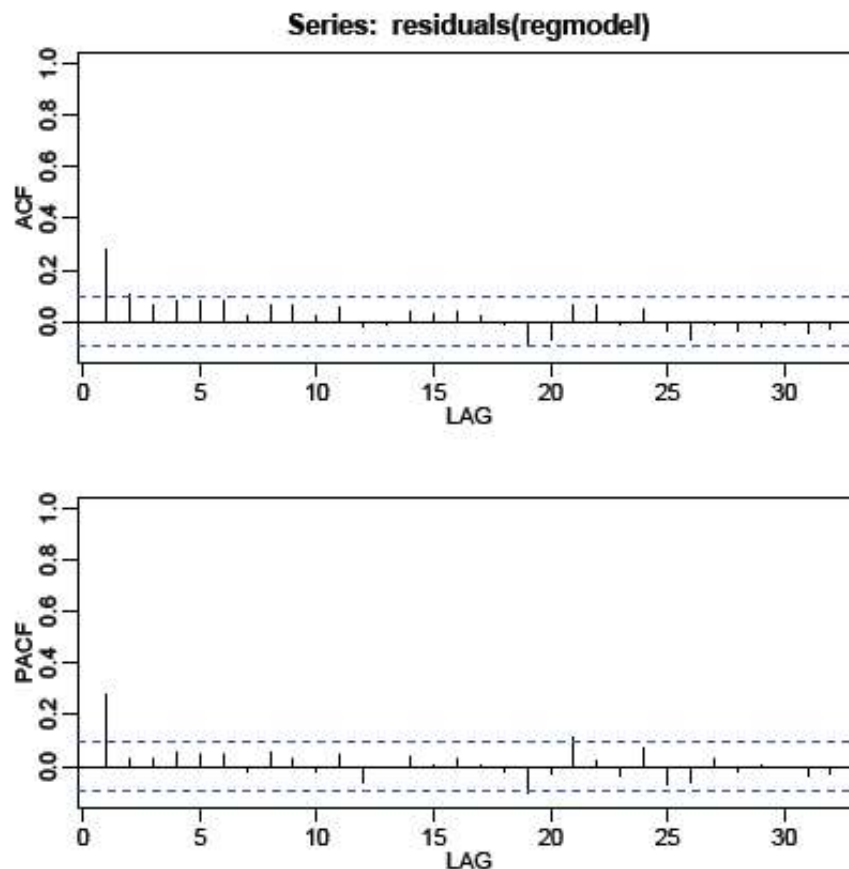
	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	1.220e+00	1.502e-02	81.19	<2e-16 ***
trend	9.047e-04	7.626e-05	11.86	<2e-16 ***
trend2	8.283e-06	6.491e-07	12.76	<2e-16 ***

Residual standard error: 0.2137 on 452 degrees of freedom

Multiple R-squared: 0.4018, Adjusted R-squared: 0.3991

F-statistic: 151.8 on 2 and 452 DF, p-value: < 2.2e-16

The autocorrelation and partial autocorrelation functions of the residuals from this model follow.



Similar to example 1, we might interpret the patterns either as an ARIMA(1,0,1), an AR(1), or a MA(1). We'll pick the AR(1) – in large part to show an alternative to the MA(1) in Example 2.

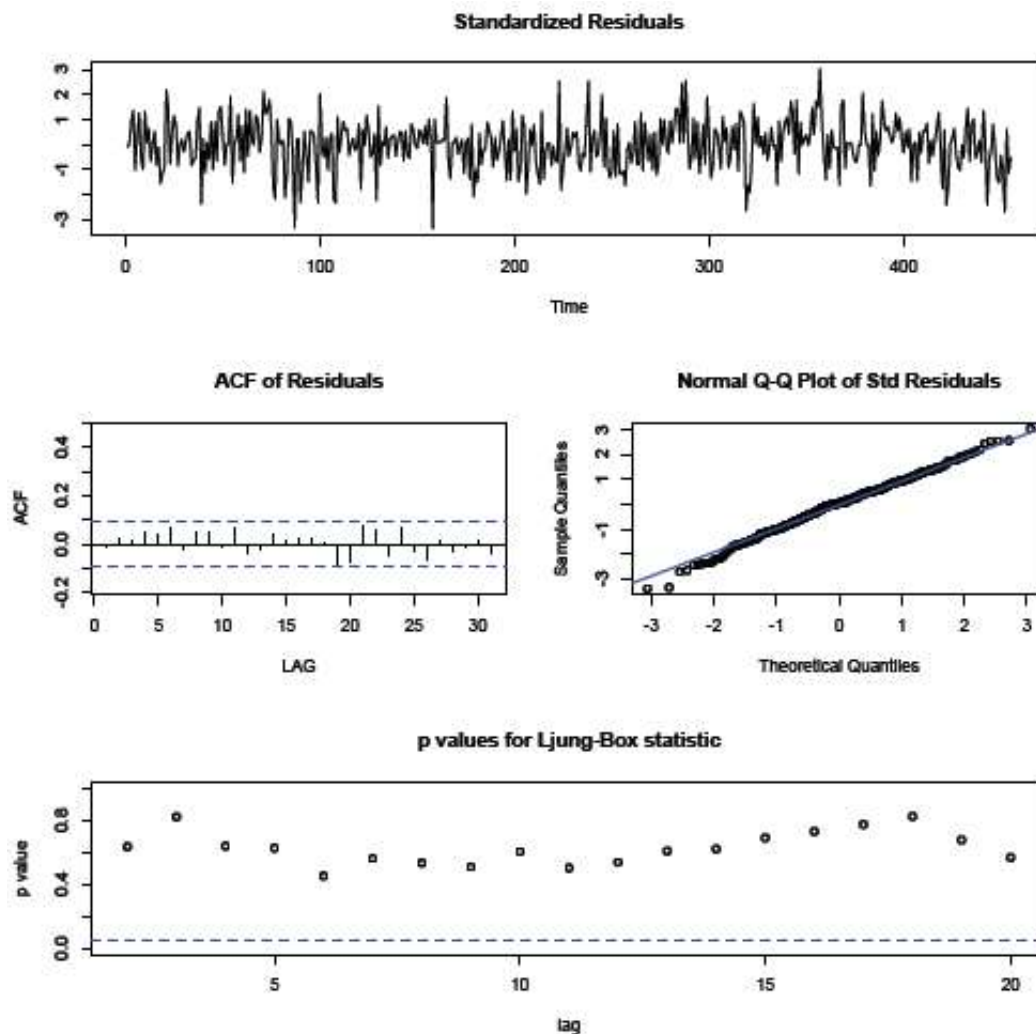
The ARIMA results for a AR(1):

Coefficients:

	arl	intercept	trend	trend2
	0.2810	1.2200	9e-04	8e-06
s.e.	0.0466	0.0202	1e-04	4e-05

sigma^2 estimated as 0.04176: log likelihood = 76.86, aic = -143.72

Check diagnostics:



The autocorrelation and partial autocorrelation functions of the residuals from this estimated model include no significant values. The fit seems is satisfactory, so we'll use these results as our final model. The estimated model is

$$\log_{10} y = 1.22018 + 0.0009029(t - \bar{t}) + 0.00000826(t - \bar{t})^2,$$

with errors $e_t = 0.2810e_{t-1} + w_t$ and $w_t \sim \text{iid } N(0, \sigma^2)$.

Note that that the predicted y is a linear function of time and the residual at the previous time.

The R Program

The data are in varve.dat in the Week 8 folder, so you can reproduce this analysis or compare to a MA(1) for the residuals.

```

library(astsa)
varve=scan("varve.dat")
varve=ts(varve[1:455])
lvarve=log(varve,10)
trend = time(lvarve)-mean(time(lvarve))
trend2=trend^2
regmodel=lm(lvarve~trend+trend2) # first ordinary regression.
summary(regmodel)
acf2(resid(regmodel))
adjreg = sarima(lvarve, 1,0,0, xreg=cbind(trend,trend2)) #AR(1)
for residuals
adjreg #Note that the squared trend is not significant and may be
dropped
adjreg$fit$coef #Note that R actually prints 0's for trend2
because the estimates are so small. This command prints the
actual values.

```

8.2 Cross Correlation Functions and Lagged Regressions

The basic problem we're considering is the description and modeling of the relationship between two time series.

In the relationship between two time series (y_t and x_t), the series y_t may be related to past lags of the x -series. The **sample cross correlation function (CCF)** is helpful for identifying lags of the x -variable that might be useful predictors of y_t .

In R, the *sample CCF* is defined as the set of sample correlations between x_{t+h} and y_t for $h = 0, \pm 1, \pm 2, \pm 3$, and so on. A negative value for h is a correlation between the x -variable at a time before t and the y -variable at time t . For instance, consider $h = -2$. The CCF value would give the correlation between x_{t-2} and y_t .

- When one or more x_{t+h} , with h *negative*, are predictors of y_t , it is sometimes said that **x leads y** .
- When one or more x_{t+h} , with h *positive*, are predictors of y_t , it is sometimes said that **x lags y** .

In some problems, the goal may be to identify which variable is leading and which is lagging. In many problems we consider, though, we'll examine the x -variable(s) to be a leading variable of the y -variable because we will want to use values of the x -variable to predict future values of y .

Thus, we'll usually be looking at what's happening at the negative values of h on the CCF plot.

Note to Minitab Users: Minitab calculates its sample CCF as the set of sample correlations between x_t and y_{t+h} . Hence, the " x leading y " side of the plot is for h positive. That's where x

comes before y in time.

Transfer Function Models

In a full **transfer function model**, we model y_t as potentially a function of past lags of y_t and current and past lags of the x -variables. We also usually model the time series structure of the x -variables as well. We'll take all of that on next week. This week we'll just look at the use of the CCF to identify some relatively simple regression structures for modeling y_t .

Sample CCF in R

The CCF command is

```
ccf(x-variable name, y-variable name).
```

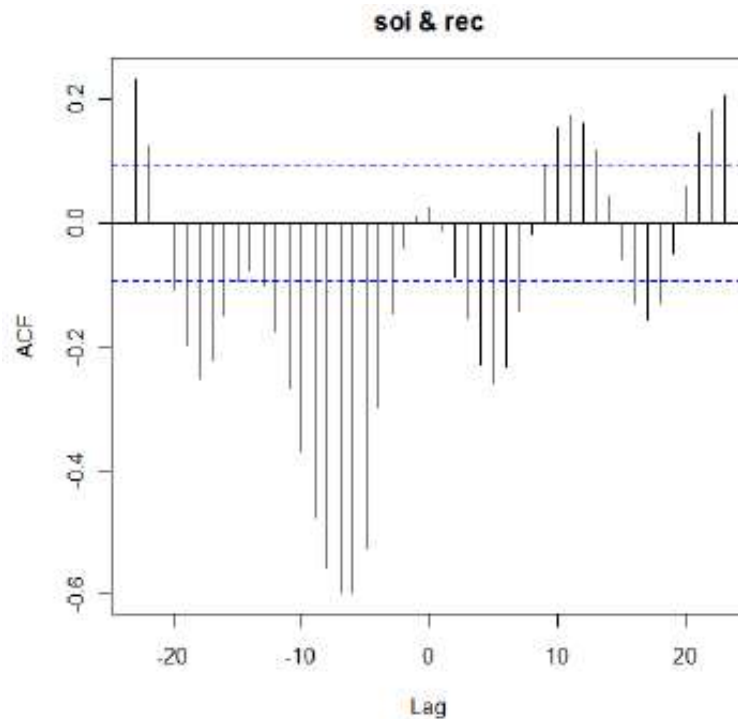
If you wish to specify how many lags to show, add that number as an argument of the command. For instance, `ccf(x,y, 50)` will give the CCF for values of $h = 0, \pm 1, \dots, \pm 50$.

Example: Southern Oscillation Index and Fish Populations in the southern hemisphere.

The text describes the relationship between a measure of weather called the Southern Oscillation Index (SOI) and "recruitment," a measure of the fish population in the southern hemisphere. The data are monthly estimates for $n = 453$ months. We see SOI as a potential predictor of recruit.

The data are in two different files. The CCF below was created with these commands:

```
soi= scan("soi.dat")
rec = scan("recruit.dat")
soi=ts (soi)
rec = ts(rec)
ccf (soi, rec)
```



The most dominant cross correlations occur somewhere between $h = -10$ and about $h = -4$. It's difficult to read the lags exactly from the plot, so we might want to give an object name to the ccf and then list the object contents. The following two commands will do that for our example.

```
ccfvalues = ccf(soi,rec)
ccfvalues
```

The result, showing lag (the h in x_{t+h}) and correlation with y_t :

-23	-22	-21	-20	-19	-18	-17	-16	-15	-14	
-13	0.235	0.125	0.000	-0.108	-0.198	-0.253	-0.222	-0.149	-0.092	-0.076
-0.103										
-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2
-0.175	-0.267	-0.369	-0.476	-0.560	-0.598	-0.599	-0.527	-0.297	-0.146	
-0.042										
-1	0	1	2	3	4	5	6	7	8	9
0.011	0.025	-0.013	-0.086	-0.154	-0.228	-0.259	-0.232	-0.144	-0.017	
0.094										
10	11	12	13	14	15	16	17	18	19	
20	0.154	0.174	0.162	0.118	0.043	-0.057	-0.129	-0.156	-0.131	-0.049
0.060										

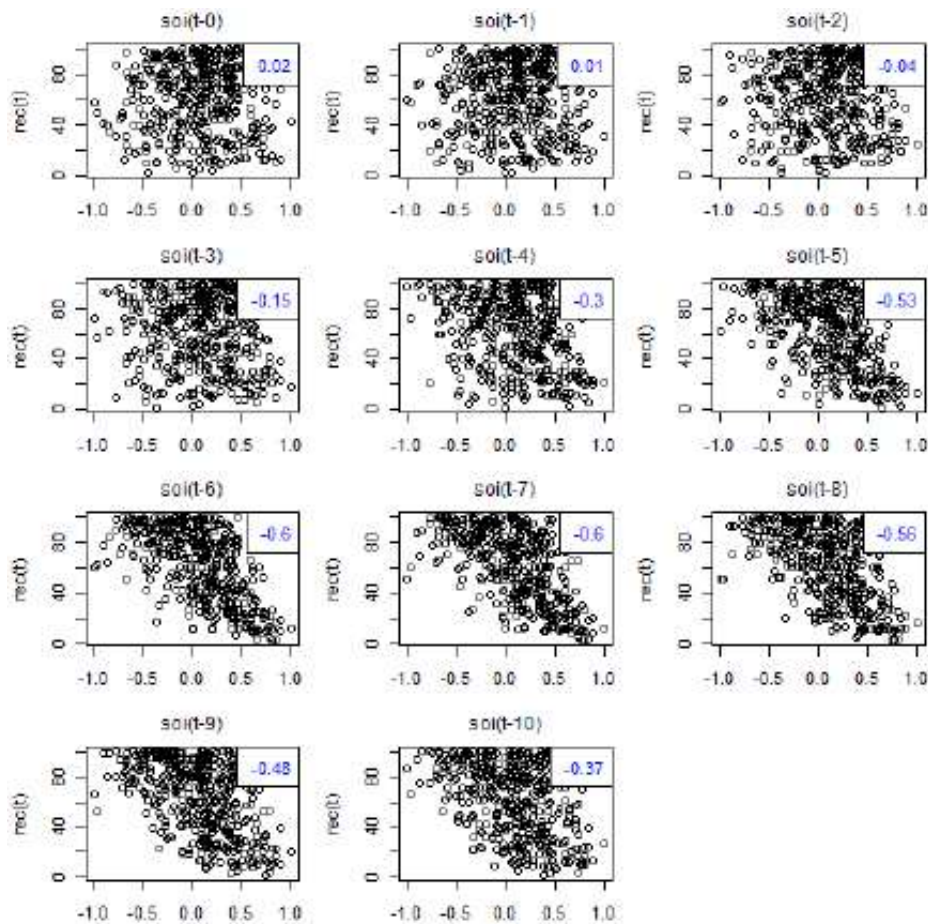
There are nearly equal maximum values at $h = -5$, -6 , -7 , and -8 with tapering occurring in both directions from that peak.

Note that the correlations in this region are negative, indicating that an above average value of SOI is likely to lead to a below average value of “recruit” about 6 months later. And, a below average of SOI is associated with a likely above average recruit value about 6 months later.

Scatterplots

In the "astsa" library that we've been using, Stoffer included a script that produces scatterplots of y_t versus x_{t+h} for negative h from 0 back to a lag that you specify. The command is `lag2.plot`.

The result of the command `lag2.plot(soi, rec, 10)` is shown below. In each plot, (recruit variable) is on the vertical and a past lag of SOI is on the horizontal. Correlation values are given on each plot.



Regression Models

There are a lot of models that we could try based on the CCF and lagged scatterplots for these data. For demonstration purposes, we'll first try a multiple regression in which y_t , the recruit variable, is a linear function of (past) lags 5, 6, 7, 8, 9, and 10 of the SOI variable. That model works fairly well. Following is some R output. All coefficients are statistically significant and the R-squared is about 62%.

The residuals, however, have an AR(2) structure, as seen in the graph following the regression output. We might try the method described in Lesson 8.1 to adjust for that, but we'll take a different approach that we'll describe after the output display.

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	69.2743	0.8703	79.601	< 2e-16	***
soilag5	-23.8255	2.7657	-8.615	< 2e-16	***
soilag6	-15.3775	3.1651	-4.858	1.65e-06	***
soilag7	-11.7711	3.1665	-3.717	0.000228	***
soilag8	-11.3008	3.1664	-3.569	0.000398	***
soilag9	-9.1525	3.1651	-2.892	0.004024	**
soilag10	-16.7219	2.7693	-6.038	3.33e-09	***

Residual standard error: 17.42 on 436 degrees of freedom

(20 observations deleted due to missingness)

Multiple R-squared: 0.6251, Adjusted R-squared: 0.62

Next week we'll discuss more about ways to interpret the CCF. One feature that will be described in more detail (with the "why") is that a peak in a CCF followed by a tapering pattern is an indicator that lag 1 and possibly lag 2 values of the y-variable may be helpful predictors.

So, our try number 2 for a regression model will be to use lag 1 and lag 2 values of the y-variable as well as lags 5 through 10 of the x-variable as linear predictors. Here's the outcome:

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	11.43047	1.33384	8.570	< 2e-16	***
reclag1	1.25702	0.04316	29.128	< 2e-16	***
reclag2	-0.41946	0.04120	-10.182	< 2e-16	***
soilag5	-21.19210	1.11838	-18.949	< 2e-16	***
soilag6	9.77648	1.56238	6.257	9.4e-10	***
soilag7	-1.19189	1.32247	-0.901	0.3679	
soilag8	-2.17345	1.30806	-1.662	0.0973	.
soilag9	0.56520	1.30035	0.435	0.6640	
soilag10	-2.58630	1.19529	-2.164	0.0310	*

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7.034 on 434 degrees of freedom

(20 observations deleted due to missingness)

Multiple R-squared: 0.9392, Adjusted R-squared: 0.938

The R-squared value has gone to about 94%. Not all sample coefficients are statistically significant. Although it's dangerous to drop too much from a model at once, we might think about dropping lags 7, 8, 9, and maybe 10 of SOI from the model. You might disagree with dropping lag 10 of SOI, but we'll try it because it seems odd to have a "stray" term like that.

So our third attempt is to predict y_t using lags 1 and 2 of itself and lags 5 and 6 of the x-variable (SOI). Here's what happens:

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	8.78498	1.00171	8.770	< 2e-16	***
reclag1	1.24575	0.04314	28.879	< 2e-16	***
reclag2	-0.37193	0.03846	-9.670	< 2e-16	***
soilag5	-20.83776	1.10208	-18.908	< 2e-16	***
soilag6	8.55600	1.43146	5.977	4.68e-09	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

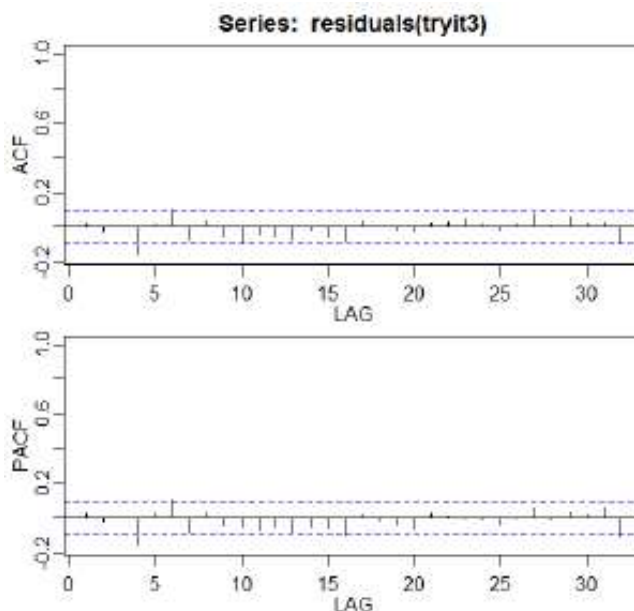
Residual standard error: 7.069 on 442 degrees of freedom

(16 observations deleted due to missingness)

Multiple R-squared: 0.9375, Adjusted R-squared: 0.937

F-statistic: 1658 on 4 and 442 DF, p-value: < 2.2e-16

All sample coefficients are significant and the R-squared is about 94%. The ACF and PACF of the residuals look pretty good. There's a barely significant residual autocorrelation at lag 4 which we may or may not want to worry about.



Complications

The CCF pattern is affected by the underlying time series structures of the two variables and the trend each series has. It often (perhaps most often) is helpful to de-trend and/or take into account the univariate ARIMA structure of the x-variable before graphing the CCF. We'll play with this a bit in the homework this week and will take it on more fully next week.

R code

Here's the full R code for this handout. The `alldata=ts.intersect()` command preserves proper alignment between all of the lagged variables (and defines lagged variables). The `tryit=lm()` commands are specifying the various regression models and saving results as named objects.

```
library(astsa)
soi= scan("soi.dat")
rec = scan("recruit.dat")
soi=ts (soi)
rec = ts(rec)
ccfvalues =ccf (soi, rec)
ccfvalues
lag2.plot (soi, rec, 10)
alldata=ts.intersect(rec,reclag1=lag(rec,-1), reclag2=lag(rec,-2), soilag5 =
lag(soi,-5),
soilag6=lag(soi,-6), soilag7=lag(soi,-7), soilag8=lag(soi,-8),
soilag9=lag(soi,-9),
soilag10=lag(soi,-10))
tryit = lm(rec~soilag5+soilag6+soilag7+soilag8+soilag9+soilag10, data = alldata)
summary (tryit)
acf2(residuals(tryit))
tryit2 =
lm(rec~reclag1+reclag2+soilag5+soilag6+soilag7+soilag8+soilag9+soilag10,
data = alldata)
```

```
summary (tryit2)
acf2(residuals(tryit2))
tryit3 = lm(rec~reclag1+reclag2+ soilag5+soilag6, data = alldata)
summary (tryit3)
acf2(residuals(tryit3))
```

Source URL: <https://onlinecourses.science.psu.edu/stat510/node/53>