Learning Objectives

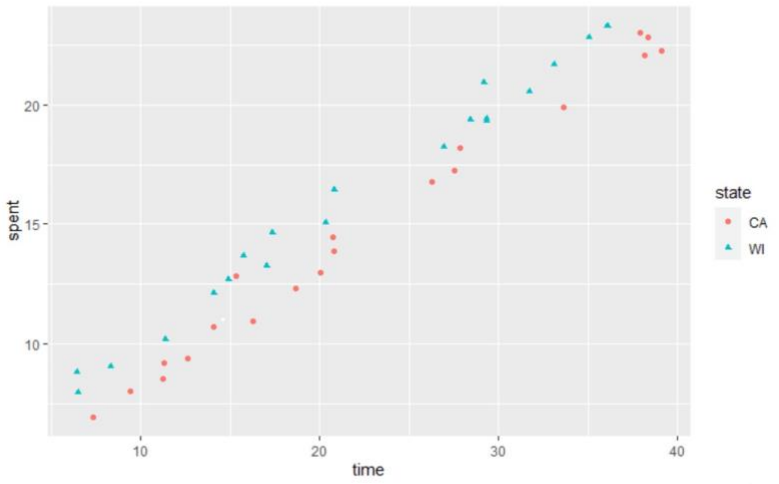
By the end of this lesson, you will be able to:

- Identify when an ordinary least squares regression model is inappropriate due to correlated error terms.
- Write the form of a covariance matrix for regression models with independent, grouped (compound symmetric), and autocorrelated error terms.
- Use generalized least squares in R to model data with grouped (compound symmetric) and autocorrelated error terms.

Generalized Least Squares

- Previously, we saw how to use WLS when different data points' noise terms ϵ_i have different **variances**.
- Generalized least squares allows for different noise terms to be **correlated**.

Example: Sales in Different States

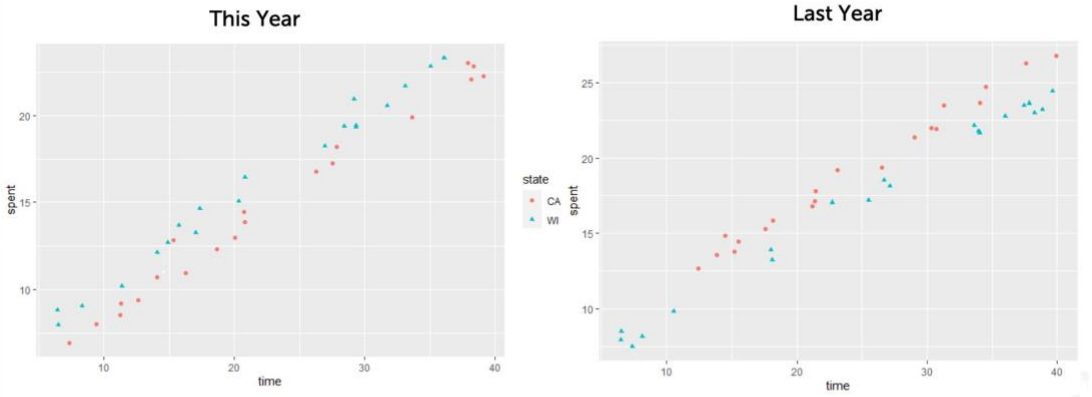


Suppose we want to investigate the relationship between the amount of time and money the customer spent in a store by taking samples of the same size from each of 20 different stores in Wisconsin and 20 different stores in California. We might get a result like this, where the y-intercept for the Wisconsin stores seems to be slightly higher than the y-intercept for the California stores.

Notes:

```
library(MASS)
library(MASS)
set.seed(60)
SigmaWI = matrix(rep(.7,20*20),nr=20)
SigmaCA = matrix(rep(.7,20*20),nr=20)
diag(SigmaWI) = 1
diag(SigmaCA) = 1
WItime = runif(20, 5, 40)
WIspent = .5*WItime + 5 + mvrnorm(1, rep(0,20), SigmaWI) #
generate 1 random vector of 20 noise terms
CAtime = runif(20, 5, 40)
CAspent = .5*CAtime + 5 + mvrnorm(1, rep(0,20), SigmaCA)

state_sales = data.frame(time = c(WItime, CAtime), spent =
c(WIspent, CAspent), state = rep(c("WI", "CA"), each = 20))

state_sales %>%
  gf_point(spent ~ time, color =~ state, pch =~ state)
```

Example: Sales in Different States



But maybe last year we did the same study and found the opposite result, that the y-intercept for California was higher than the y-intercept for Wisconsin. In this case, you might suspect that there's not a fixed difference between Wisconsin and California that should be modeled using multiple regression. Instead, it might be that the random noise term for each Wisconsin store tends to be correlated with the random noise term from other Wisconsin stores and similarly in California. This could be due to temporary effects of unmeasured variables such as the weather or the economy within a particular state.

Notes:

Same R code as previous slide, but the graph on the right uses `set.seed(64)`.

Matrices in Ordinary Least Squares Regression

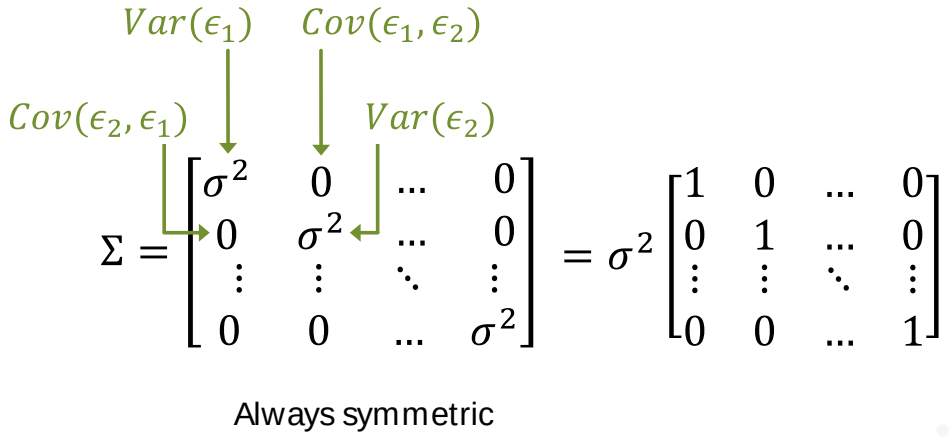
$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

$$\epsilon_i \sim N(0, \sigma^2)$$

$$\begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix} \sim MVN\left(\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma^2 & 0 & \dots & 0 \\ 0 & \sigma^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sigma^2 \end{bmatrix} \right)$$

Recall that in ordinary least-squares regression, if we have a single predictor variable x we get the formula shown here for the response value of the I'th data point. In this model, epsilon sub I is the noise term for the I'th point. And it has a normal distribution with Mean 0 and variance sigma squared. If we stack up the noise terms from all of the data points, we could get a vector. And we would say that this has a multivariate normal distribution with mean equal to a vector of 0s and variance equal to a matrix.

Covariance Matrix



We call this the covariance matrix, sigma. The elements on the diagonal of the covariance matrix give the variances of the different noise terms where the first diagonal element is the variance of epsilon sub 1, the second diagonal element is the variance of epsilon sub 2, and so on. The elements off the diagonal represent the covariances between different noise terms. The element in the first row and second column is the covariance between epsilon sub 1 and epsilon sub 2 while the element in the second row and first column is the covariance between epsilon sub 2 and epsilon sub 1.

Of course, we can take the covariance of two variables in either order and get the same value, which means that the covariance matrix will always be symmetric across the diagonal. We can rewrite the covariance matrix by splitting it up in terms of the part we need to estimate and the part we suspect we know based on the structure of the data. For ordinary, least-squares regression, we need to estimate sigma squared, the variance of the noise terms.

We suspect we know that the overall structure is that of an identity matrix. The 1s on the diagonal represent the fact that different noise terms have the same variance. The 0s off the diagonal represent the fact that different noise terms are not correlated.

Notes:

For more about covariance matrices, see Covariance Matrices, Covariance Structures, and Bears, Oh My! - Karen Grace-Martin

Compound Symmetric Covariance Matrix

$$\Sigma = \sigma^2 \begin{bmatrix} 1 & \rho & 0 & 0 \\ \rho & 1 & 0 & 0 \\ 0 & 0 & 1 & \rho \\ 0 & 0 & \rho & 1 \end{bmatrix}$$

WI 1, WI 2, CA 1, CA 2 (row and column labels)

grouping variable

```
library(nlme)
fit_state = gls(spent ~ time,
     correlation = corCompSymm(form = ~1 | state),
     data = state_sales)
```

For our example with Wisconsin and California, the part that we think we know based on the structure of the data is that different stores in the same group or state all have the same correlation with each other, which we'll denote by the Greek letter rho. We'll need to estimate rho at the same time as we estimate sigma squared. The correlation between different stores in different states will be 0. This is known as a compound symmetric covariance matrix.

We can model this in R using the GLS function for generalized least squares, which is in the NLME library. We specify that we want a compound symmetric covariance matrix using the correlation argument. The variable that comes after the vertical line is the grouping variable. If we wanted all of our observations to be in the same group or state, we could omit this. And that would represent a situation where all of the stores were equally correlated with each other.

Generalized Least Squares in R

```
> summary(fit_state)
Generalized least squares fit by REML
  Model: spent ~ time
  Data: NULL
       AIC      BIC    logLik
  89.59323 96.14358 -40.79662

Correlation Structure: Compound symmetry
 Formula: ~1 | state
 Parameter estimate(s):
      Rho
0.8175994

Coefficients:
             Value Std.Error  t-value p-value
(Intercept) 4.309712 0.8810038  4.89182       0
time        0.500142 0.0090843 55.05546       0
```

$\hat{\rho}$

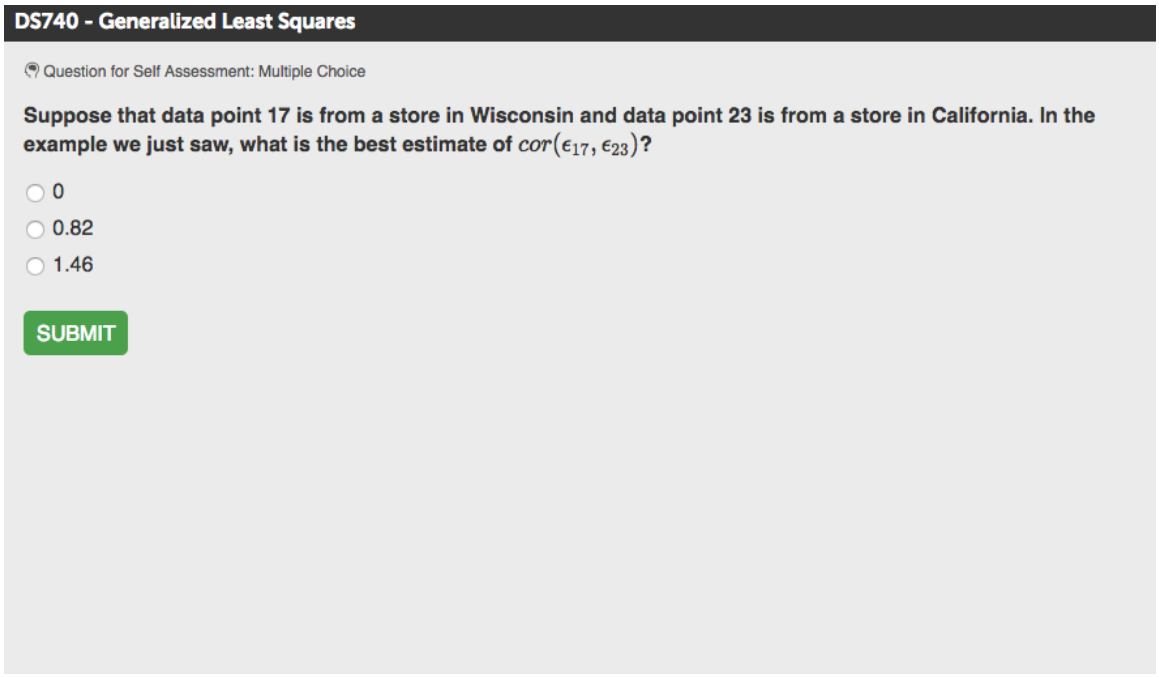
The summary of our GLS output looks like this. The coefficient section has the same structure as what we know from a linear model, with the first column representing the estimated coefficients. We also get an estimate of rho, the correlation between different observations in the same group.

Estimated Covariance Matrix

```
> getVarCov(fit_state)
Marginal variance covariance matrix
        [,1]   [,2]   [,3]
 [1,] 1.7813 1.4564 1.4564 ← ρ̂ * σ̂²
 [2,] 1.4564 1.7813 1.4564
 [3,] 1.4564 1.4564 1.7813 ← σ̂²
```

Using the GetVerCov function, we can see the estimated covariance matrix. The elements along the diagonal are the estimated values of sigma squared. And the off-diagonal elements are the estimated values of rho times sigma squared. You won't see any 0s in this matrix. To save space, R omits them.

Question 1

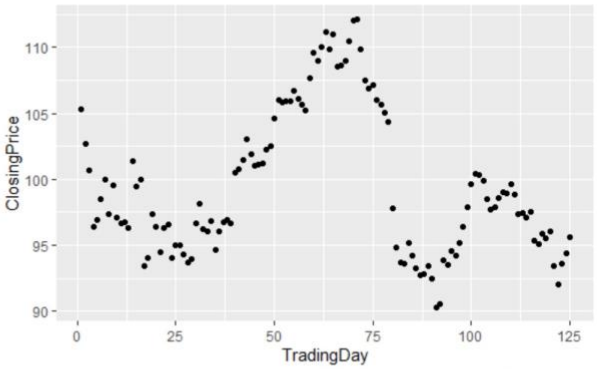


Answer is at the end of the transcript

Sample Problem

Model closing stock price as a function of company and log(shares traded).

Allow for the possibility that consecutive days may have correlated error terms.




```
stock = read_csv("Stock 2016.csv")
```

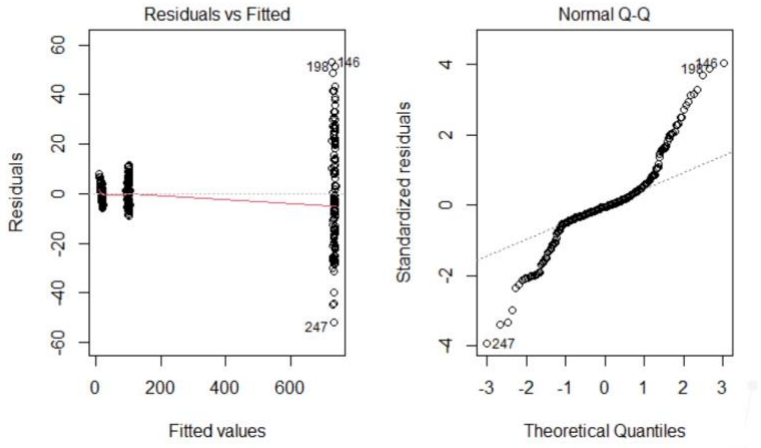
Notes:

Stock prices from Jan 4, 2016 to June 30, 2016. Retrieved from Yahoo Finance via quantmod package in R on 15 August 2016.

```
stock = read_csv("Stock 2016.csv")
stock %>%
  filter(Company == "Apple") %>%
  gf_point(ClosingPrice ~ TradingDay)
```

Ordinary Least-Squares Regression: Diagnostic Plots

```
fit_stock = lm(ClosingPrice ~ logVolume + Company, data=stock)
par(mfrow=c(2,2))
plot(fit_stock)
```



To analyze the closing price of stocks, we'll start by performing ordinary least-squares regression using the company and the log volume of shares traded as the predictor variables. Looking at our diagnostic plots, the residuals versus fitted values graph actually doesn't look too bad. The three distinct clusters of points are not that surprising given that we have three distinct companies in our data. However, the normal QQ graph shows us that our residuals are not even close to being normally distributed. In addition, R has flagged three of the points as being potential outliers.

Investigating the Outliers

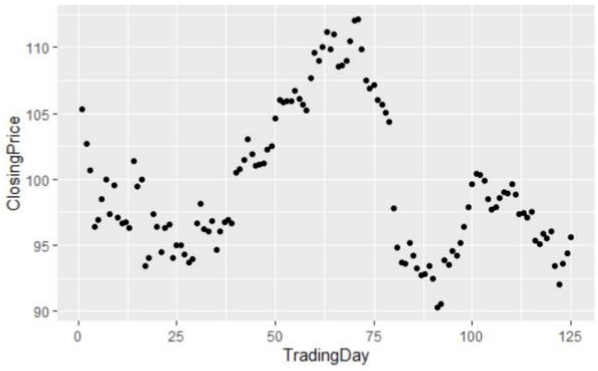
```
> stock[c(146,198,247),]
    TradingDay ClosingPrice logVolume Company DayOfYear
146         21       780.91  15.73130  Google        33
198         73       787.68  14.33252  Google       109
247        122       681.14  14.88274  Google       179
```

This data set doesn't have row names. So if we want to know more about the outliers such as what company they come from, we need to look at the rows of the stock data frame that correspond to the row numbers of the outliers.

Autocorrelation

When successive data points (or residuals) are correlated.

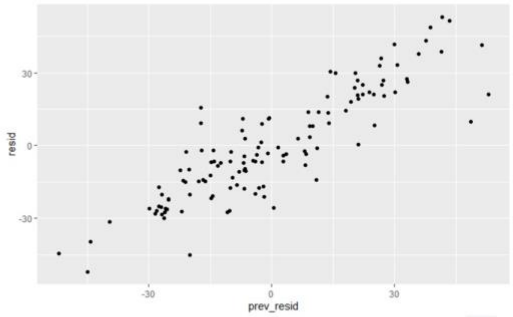
Common in time series data.



In this data set, we might suspect that we have autocorrelation, which refers to when successive data points or residuals are correlated with each other. This is common in time-series data due to unmeasured variables that change gradually over time such as public sentiment about a company or the state of the economy.

Visualizing Autocorrelation

```
stock_Google <- stock %>%
  mutate(resid = fit_stock$residuals) %>%
  filter(Company == "Google")

stock_Google <- stock_Google %>%
  mutate(prev_resid = lag(resid))

stock_Google %>%
  gf_point(resid ~ prev_resid)
```



To get a visual understanding of whether there's autocorrelation in our data, we can make a scatterplot of each residual as a function of the lagged or previous residual.

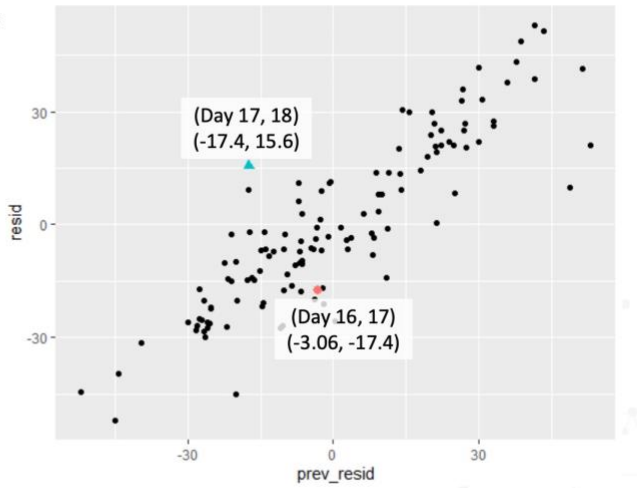
Notes:

```
stock_Google <- stock %>%
  mutate(resid = fit_stock$residuals) %>%
  filter(Company == "Google")

stock_Google <- stock_Google %>%
  mutate(prev_resid = lag(resid))

stock_Google %>%
  gf_point(resid ~ prev_resid)
```

Visualizing Autocorrelation

Each point represents 2 days. Y-coordinate of one point becomes x-coordinate of next point.

```
cor(stock_Google$resid,
    stock_Google$prev_resid,
    use = "pairwise.complete.obs")
[1] 0.8824997
```



This produces a scatterplot, in which each point represents the residuals from two consecutive days of trading. For example, the red circle shown here has as its x-coordinate the residual from the 16th day of trading, and as its y-coordinate, the residual from the 17th day of trading. The blue triangle has as its x-coordinate day 17 and its y-coordinate is day 18.

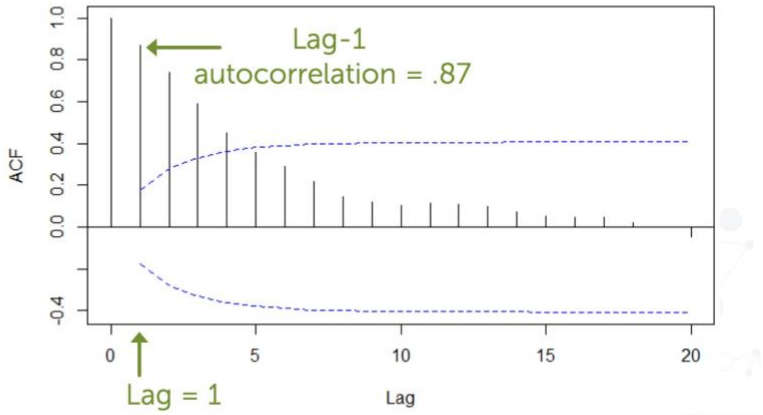
Here, we have an upward sloping trend in our scatterplot, which indicates that each residual tends to be positively correlated with the one that comes after it, so we have positive autocorrelation. We can put a numerical value to this by looking at the correlation coefficient between the residuals and the lagged residuals.

Notes:

```
example_points <- stock2 %>%
  slice(c(17,18))

stock2 %>%
  slice(c(-17, -18)) %>%
  gf_point(resid ~ prev_resid) %>%
  gf_point(resid ~ prev_resid, data = example_points,
           color =~ factor(TradingDay), pch =~
factor(TradingDay),
           size = 2.5)

cor(stock_Google$resid, stock_Google$prev_resid,
    use = "pairwise.complete.obs")
```

Autocorrelation Plot

```
acf(stock_Google$resid,
    ci.type = "ma")
```



Another way to visualize the autocorrelation is with the ACF function. The x-axis on this plot shows the lag, or number of time units between residuals being compared. If we look at a lag of one, then we're looking at the correlation between residuals from consecutive days, that is observations that were one time unit apart.

We see that ACF has computed the lag one autocorrelation for the Google stock as 0.87. This is very close to the 0.88 that we got using the core function, COR, but with a slight difference due to differences in when the two functions compute the standard deviations. To meet the assumption of independent noise terms for ordinary least squares regression to be appropriate, we want the autocorrelation to be near zero.

The dashed blue lines indicate how near zero. For a data set of this same size, where the noise terms were completely independent white noise, 95% of the time, the autocorrelation of the residuals would be between these blue lines. The argument CI type equals MA makes this confidence interval more accurate for lags larger than one.

Notes:

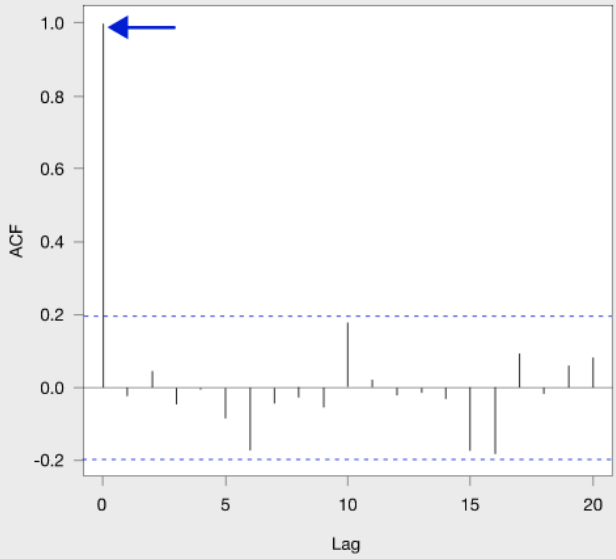
For more about covariance matrices, see:
stackoverflow.com/questions/16399788/why-do-i-get-different-results-using-ccf-and-cor-in-r

Question 2

DS740 - Generalized Least Squares

Question for Self Assessment: Multiple Choice

True or false:
In this autocorrelation plot, the line indicated by the blue arrow indicates that the error terms are not independent.

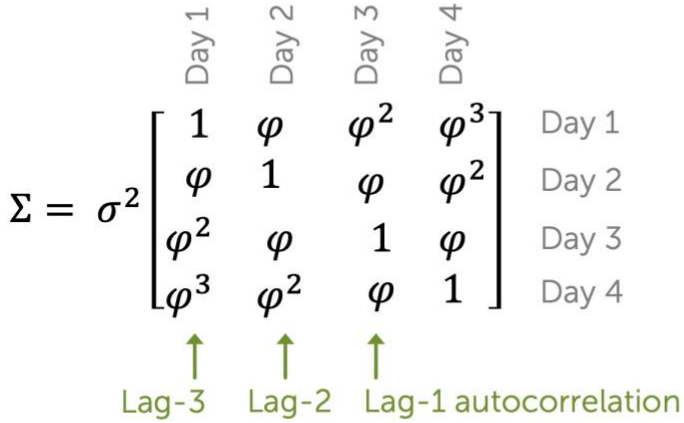


- True
- False

SUBMIT

Answer is at the end of the transcript

The AR(1) Model



We can model autocorrelation with a first-order autoregressive model, or AR1. In this model, the covariance matrix states that the correlation between any two consecutive days is phi. That's the lag 1 autocorrelation, and it needs to be estimated from the data. The correlation between observations that are two days apart is phi squared and between observations that are three days apart phi cubed and so on.

Autoregressive Model in R

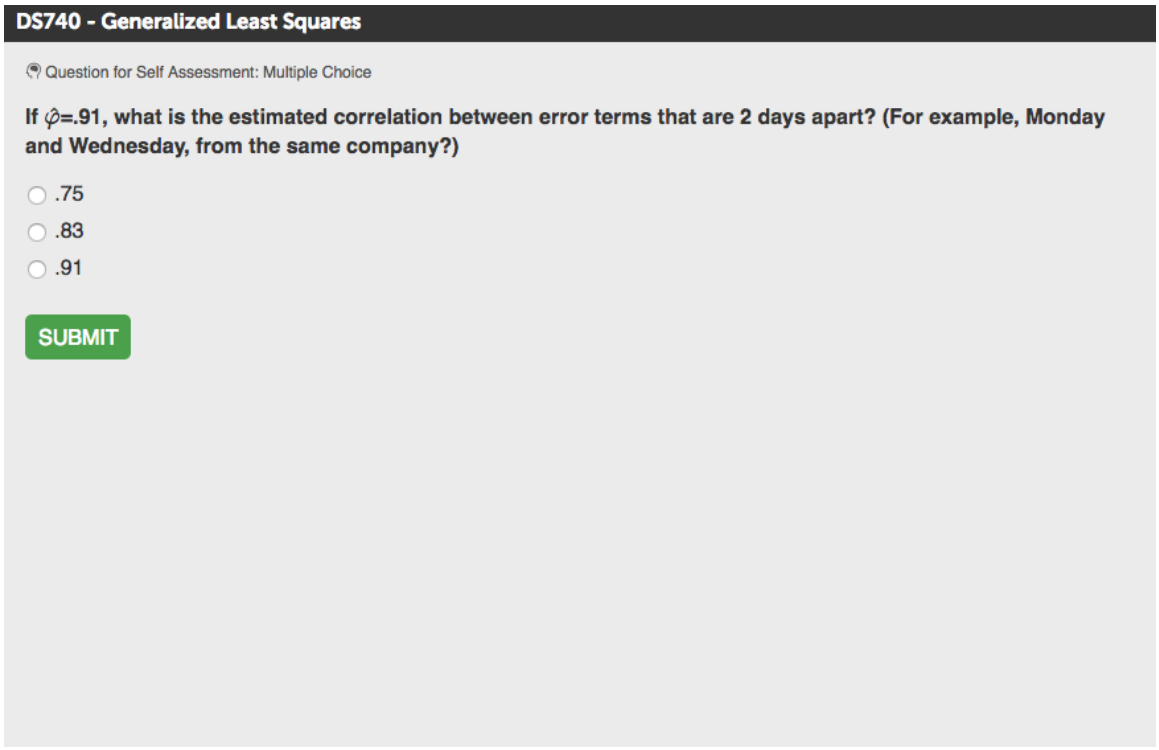
```
library(nlme)
fit_stock2 = gls(ClosingPrice ~ logVolume + Company, data=stock,
                 correlation = corAR1(form = ~1 | Company))
summary(fit2)
```

```
Generalized least squares fit by REML
  Model: ClosingPrice ~ logVolume + Company
  Data: stock
       AIC      BIC    logLik
  2429.402 2452.899 -1208.701

Correlation Structure: AR(1)
 Formula: ~1 | Company
 Parameter estimate(s):
      Phi
0.9064086

Coefficients:
                Value Std.Error   t-value p-value
(Intercept)   132.0016 18.263568   7.22759  0.0000
logVolume      -1.8446  0.992692  -1.85821  0.0639
CompanyGoogle 629.5182  8.344872  75.43773  0.0000
CompanyTwittr -83.9735  7.840659 -10.71001  0.0000
```

$\hat{\varphi}$

To implement the AR1 model in R, we can again use the GLS function. But this time our correlation will equal COR AR1 with formula equal to Squiggle 1, and the grouping variable is now company. In the output, we again see the estimated coefficients and our estimated value for phi-- the correlation.

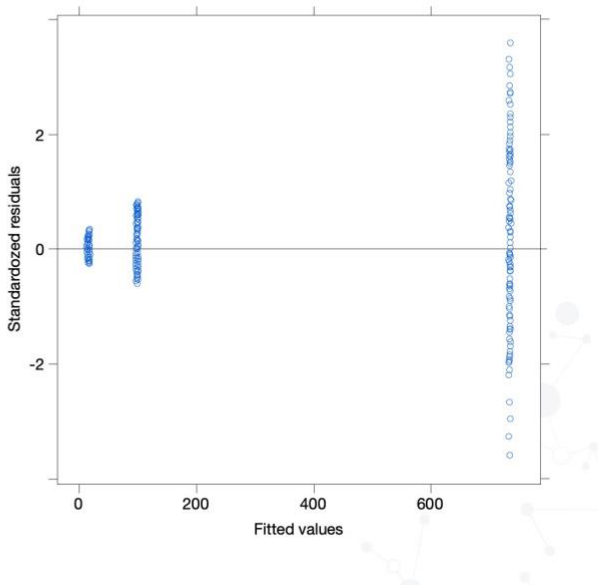
Question 3



Answer is at the end of the transcript

Diagnostic Plot

```
plot(fit_stock2)
```



The plot function will give us a diagnostic plot for our GLS object, the standardized residuals versus the fitted values. This plot still isn't great, although the range of the standardized residuals is somewhat smaller for our GLS model compared to our ordinary least-squares model. It might be good to use the weights argument of GLS to allow for different companies to have different variances.

Comparison with Uncorrelated Model

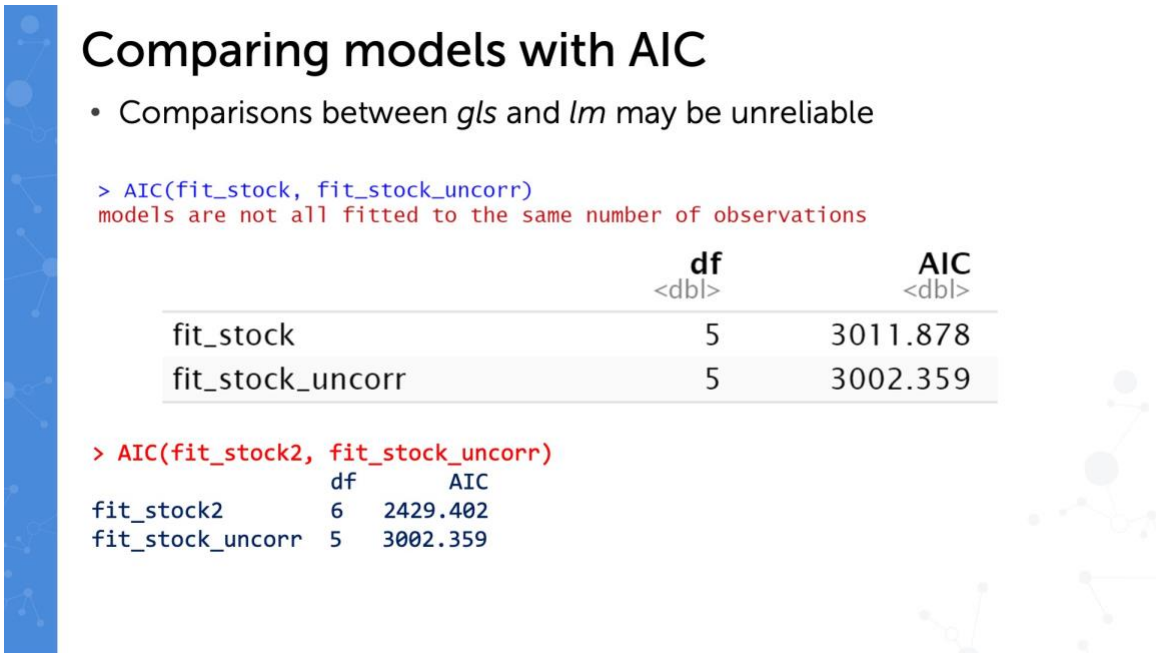
```
> fit_stock_uncorr = gls(ClosingPrice ~ logVolume + Company,
                         data=stock)
> fit _stock_uncorr
Generalized least squares fit by REML
  Model: ClosingPrice ~ logVolume + Company
  Data: stock
  Log-restricted-likelihood: -1496.18

Coefficients:
  (Intercept)     logVolume CompanyGoogle CompanyTwittr
   204.205432     -5.978419    617.845529    -86.706656

Degrees of freedom: 375 total; 371 residual
Residual standard error: 13.31548
```

If we want, we can also fit the uncorrelated model using GLS, simply by omitting the correlation argument. The estimated coefficients that we get out of this model are typically very similar to what we got using LM, because we're again fitting a model with the same predictor variables and no correlations. The only difference is how we're fitting it, using ordinary least squares in the case of GLM or using restricted maximum likelihood in the case of GLS.

Notes:

If you're comparing models with different sets of predictor variables, use the argument method = "ML" in gls() for both models. See these Stack Exchange questions:

REML or ML to compare two mixed effects models with differing fixed effects, but with the same random effect?
Why does one have to use REML (instead of ML) for choosing among nested var-covar models? - Answer by fcop



So why would we bother using GLS to fit an uncorrelated model? The reason has to do with comparing models using AIC. Because the functions GLS and LM fit the models in different ways, using AIC to compare the models can be unreliable.

You can see this if you try using the AIC function to compare our two uncorrelated models. You get a warning message saying that the models are not all fitted to the same number of observations, and you get different AIC values, even though they should be the same model, just fit in different ways. So to get a fair comparison, we'll compare the autocorrelated model with the uncorrelated model that we fit using GLS.

Here, we see that, even though the residual plot of our fit stock 2 didn't look perfect, we have achieved a significant decrease in AIC compared to the uncorrelated model. The difference between their AICs is more than two or six, so the uncorrelated model is not a reasonable alternative to the correlated model.

Notes:

It's also unreliable to use AIC to compare gls models with different predictors: https://stats.stackexchange.com/questions/116770/reml-or-ml-to-compare-two-mixed-effects-models-with-differing-fixed-effects-but

This page recommends using restricted maximum likelihood (REML, what the gls function uses) to compare models with different correlation structures on all the reasonable predictor variables, then using maximum likelihood (ML) to compare models with different fixed effects (predictor variables).
http://users.stat.umn.edu/~rend0020/Teaching/EPSY8282-2011Spring/class/class12.pdf

To fit a model with correlation structure using ML, use the argument *method = "ML"* in gls() or the argument *REML = FALSE* in lmer() from the lme4 package. (Then, once you've chosen the best model, re-fit it with *method = "REML" or REML = TRUE* to get the best coefficient estimates.)

Changing How Time is Measured

```
> stock[1:6,]
  TradingDay ClosingPrice logVolume Company DayOfYear
1          1       105.35  18.02985   Apple         4
2          2       102.71  17.83712   Apple         5
3          3       100.70  18.04172   Apple         6
4          4        96.45  18.21112   Apple         7
5          5        96.96  18.07534   Apple         8
6          6        98.53  17.72231   Apple        11
```

form = ~1 treats successive data points as being 1 time unit apart.

To account for weekends:

```
fit_stock3 = gls(ClosingPrice ~ logVolume + Company, data=stock,
             correlation = corAR1(form = ~DayOfYear | Company))
```

Another modification that we could try for our autoregressive model would be to change how time is measured. Looking at the first few rows of our data, you'll notice that we have two different ways of measuring time. The first column tells the trading day, the number of days that the stock market has been open in 2016 starting with 1 and increasing by 1 for each successive row within a given company. The last column is the day of the year, which is the actual number of days since January 1.

The correlation formula tilde 1 treats successive data points as being one time unit apart. In other words, it uses the trading day as the measurement of time. This means that it treats all consecutive pairs of trading days as being equally correlated. But if you look at the day of the year column, you'll notice that Trading Days 5 and 6 were actually three days apart because they were separated by a weekend. So we might suspect that the closing price on Days 5 and 6 was actually less correlated than the closing price on Days 1 and 2.

To account for this possibility, we can modify our correlation formula to include Squiggle Day of Year where Day of Year is the variable that we want to use to measure time. In the case of this stock data set, this turns out not to improve our model. But it is a useful tool to know.

Other Correlation Structures

Many correlation structures are possible in real data and in `gls()`.

Here are a few commonly-used models:

- corARMA: for more complicated autocorrelation models.
- corCAR1: autocorrelation with continuous time.
- corLin: when points near each other in space are correlated.

Notes:

For more about spatial correlation and ARMA, see Hierarchical models II: Correlated observations - syr.edu

Summary: Generalized Least Squares

- Generalized least squares allows different noise terms to be correlated.
- It can be applied with the `gls()` function in R.

Summary: Common Types of Covariance Matrix

Compound Symmetry

- All error terms within a group of data points have the same correlation.

Autocorrelation

- Correlation = φ^n, where n = number of time units separating the data points.

Group 1 Group 2

$$\Sigma = \sigma^2 \begin{bmatrix} 1 & \rho & 0 & 0 \\ \rho & 1 & 0 & 0 \\ 0 & 0 & 1 & \rho \\ 0 & 0 & \rho & 1 \end{bmatrix}$$

Group 1

Group 2

$$\Sigma = \sigma^2 \begin{bmatrix} 1 & \varphi & \varphi^2 & \varphi^3 \\ \varphi & 1 & \varphi & \varphi^2 \\ \varphi^2 & \varphi & 1 & \varphi \\ \varphi^3 & \varphi^2 & \varphi & 1 \end{bmatrix}$$

Question 1 answer

DS740 - Generalized Least Squares

Feedback for Self Assessment

✓ Correct!

Suppose that data point 17 is from a store in Wisconsin and data point 23 is from a store in California. In the example we just saw, what is the best estimate of $cor(\epsilon_{17}, \epsilon_{23})$?

Your answer:
0

Correct answer:
0

Feedback:
Yes! In this example, Wisconsin and California formed different groups. In a compound symmetry model, we assume the correlation between the error terms of data points in different groups is 0.

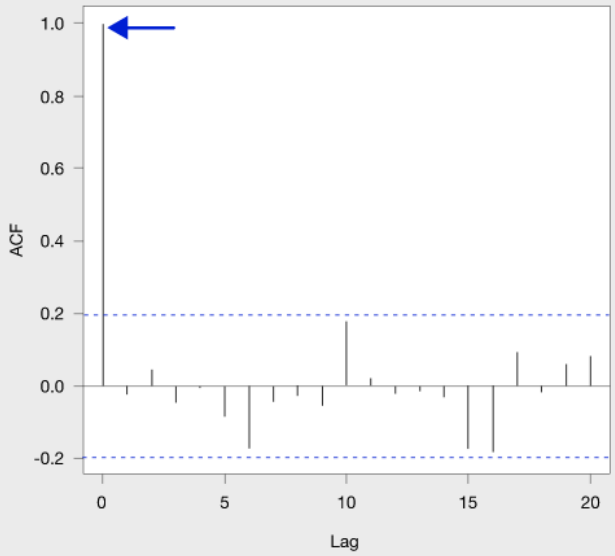
Question 2 answer

DS740 - Generalized Least Squares

Feedback for Self Assessment

✓ Correct!

True or false:
In this autocorrelation plot, the line indicated by the blue arrow indicates that the error terms are not independent.



Your answer:
False

Correct answer:
False

Question 3 answer

DS740 - Generalized Least Squares

Feedback for Self Assessment

✓ Correct!

If $\hat{\varphi}$=.91, what is the estimated correlation between error terms that are 2 days apart? (For example, Monday and Wednesday, from the same company?)

Your answer:
.83

Correct answer:
.83

Feedback:
Yes, the estimated value of $cor(\epsilon_i, \epsilon_{i+2})$ is $.91^2$.