ch_9_serially_correlated_errors

Jack Wright

11/4/2021

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(here)
```

```
## here() starts at C:/Users/jwright/Documents/GitHub/business_analytics/Modern_approach_with_r
```

```
library(ggResidpanel)
```

in many situations, data is collected over time.

these often exhibit `serial correlation`

serial correlation: -resulsts from current time period are correlated with results from earlier time periods -*violates assumption that the errors are independent*, which is an important assumption for least-squares-based regression

autocorrelation:

-correlation between a variable at different time points.

generalized least squares: -used to fit models with autocorrelated errors

transform GLS model to LS model

autocorrelation

EX:

estimating price elasticity of a food product

-want to understand effect of price on sales, particularly to develop a model to estimate the percentage effect on sales of a 1% increase in price

consider weekly sales of `Brand 1` at a major US supermarket chain over a year as a function of price each week.

$$log(Sales_t) = \beta_0 + \beta_1 log(Price_t) + e$$

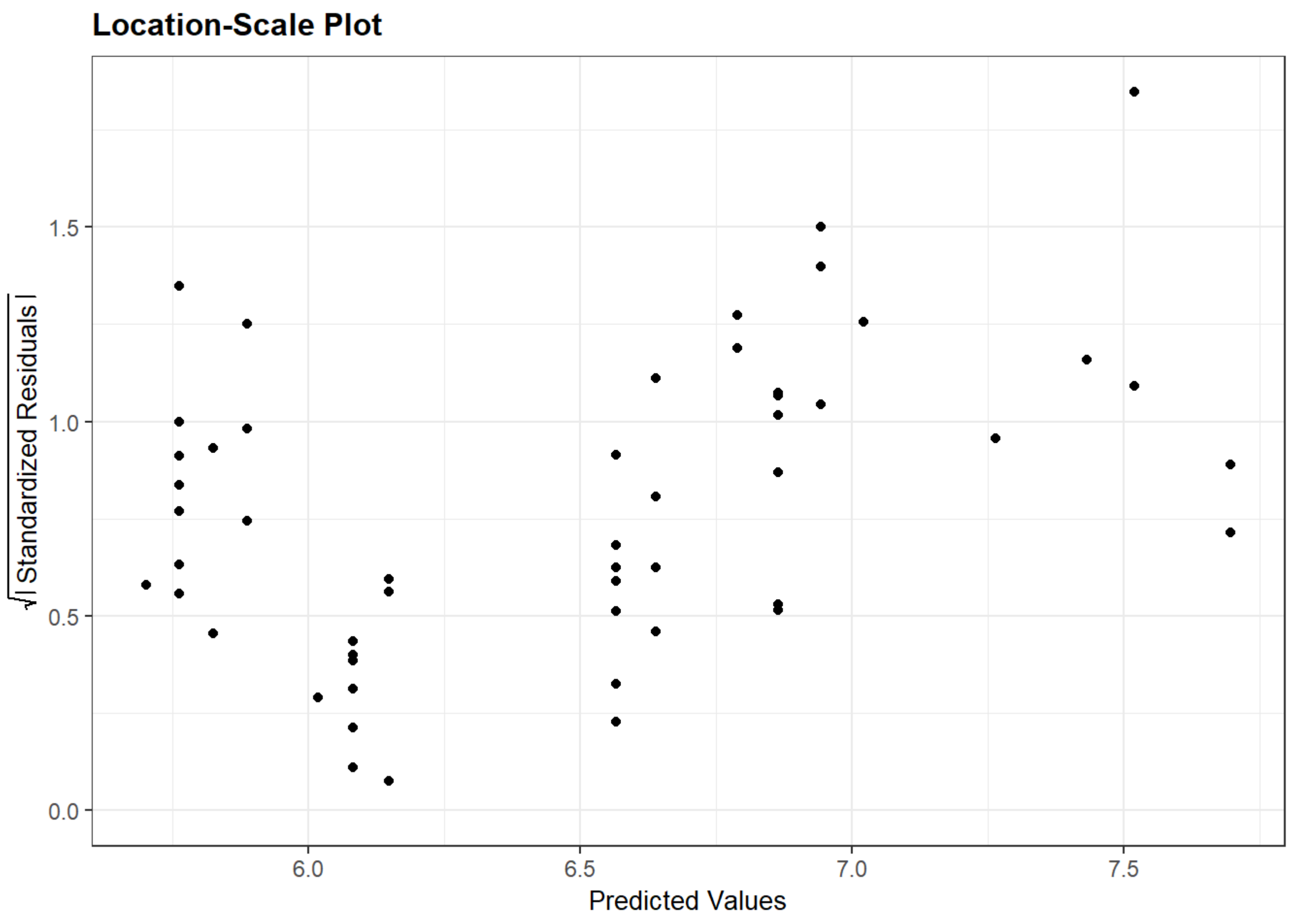
subscript t denotes week. (price at week t, sales at week t)

We find a nonrandom pattern in the plot of standardized residuals.

```
file<-here('data','confood2.txt')
df<-read.table(file, header=TRUE)

lm.mod<-lm(log(Sales)~log(Price), data=df)

resid_panel(lm.mod, plots='ls')
```



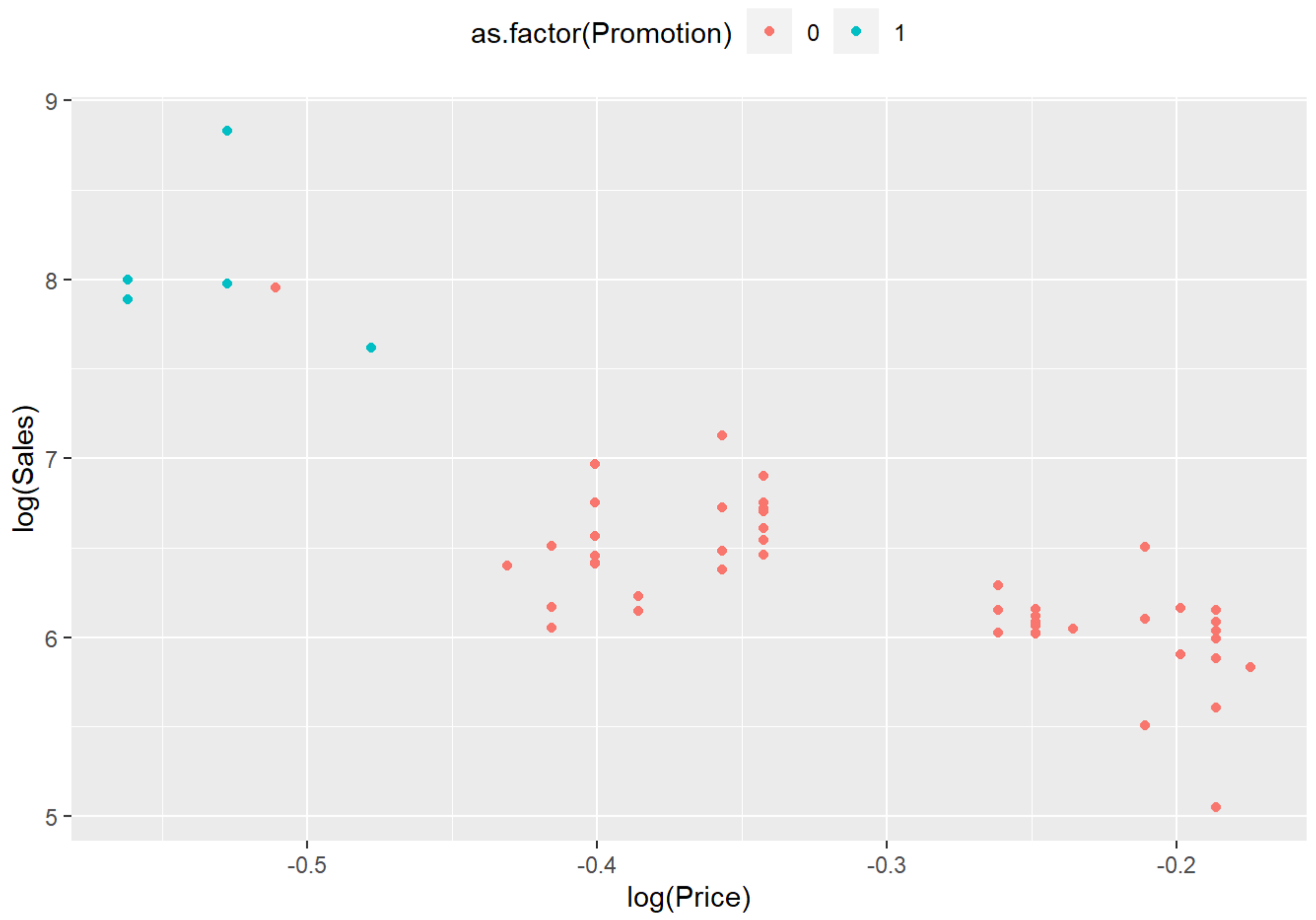
two other potential predictors:

-week

-promotion: dummy variable, promotion/no promotion

take a look at log(sales) against log(Price)

```
ggplot(df%>%group_by(as.factor(Promotion)), aes(x=log(Price),y=log(Sales)))+geom_point(aes(color
=as.factor(Promotion))) +
  theme(legend.position = "top")
```



appears to be linearly related with promotions having a big effect.

this analysis ignores that the data was collected over time

now look at the log(Sales) in week t vs log(sales) in week t-1

```
library(ggpubr)
Sales_t_1<-head(as.vector(df$Sales),-1)
Sales<-df$Sales[2:length(as.vector(df$Sales))]
df.sales<-data.frame(t_1=Sales_t_1,t=Sales)
a<-ggplot(df, aes(x=Week,y=log(Sales)))+geom_point(aes(shape=Promotion, color=Promotion))+geom_l
ine()
b<-ggplot(df.sales, aes(x=log(t_1),y=log(t)))+geom_point()
```

We see that there is a positive correlation between the previous weeks sales (sales_t-1) and the current weeks sales (Sales_t)

Is there also a positive correlation between sales in week t and sales in week t-2, or t-3…

TOO MUCH WORK.

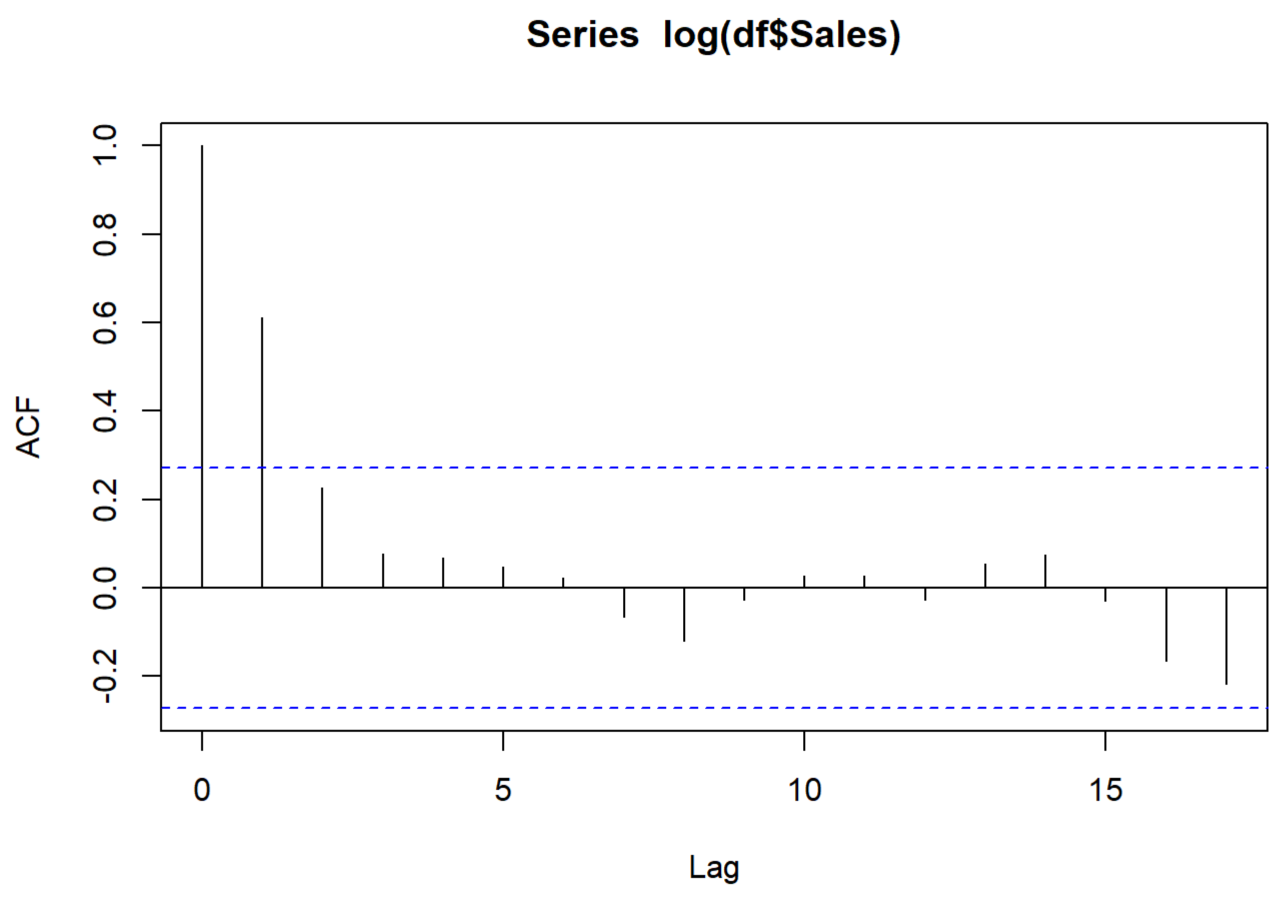
use *autocorrelations*

correlation between Y and various values of lagged Y for different periods

**autocorrelation of lag “l” is the correlation between Y and the values of Y lagged by ‘l’ periods between Y_t and Y_{t-l}

$$Autocorrelation(l) = \frac{\sum(y_t - \bar{y})(y_{t-l} - \bar{y})}{\sum(y_t - \bar{y})^2}$$

```
stats::acf(log(df$Sales),lag.max=17,plot=TRUE)
```



first 17 autocorrelations of log(Sales)

Statistically significantly different if they are outside $\pm 2/n$ where n is number of samples. (the blue lines)

week 1 only exceeds the test, because clearly t-0 is going to have correlation of 1 with t

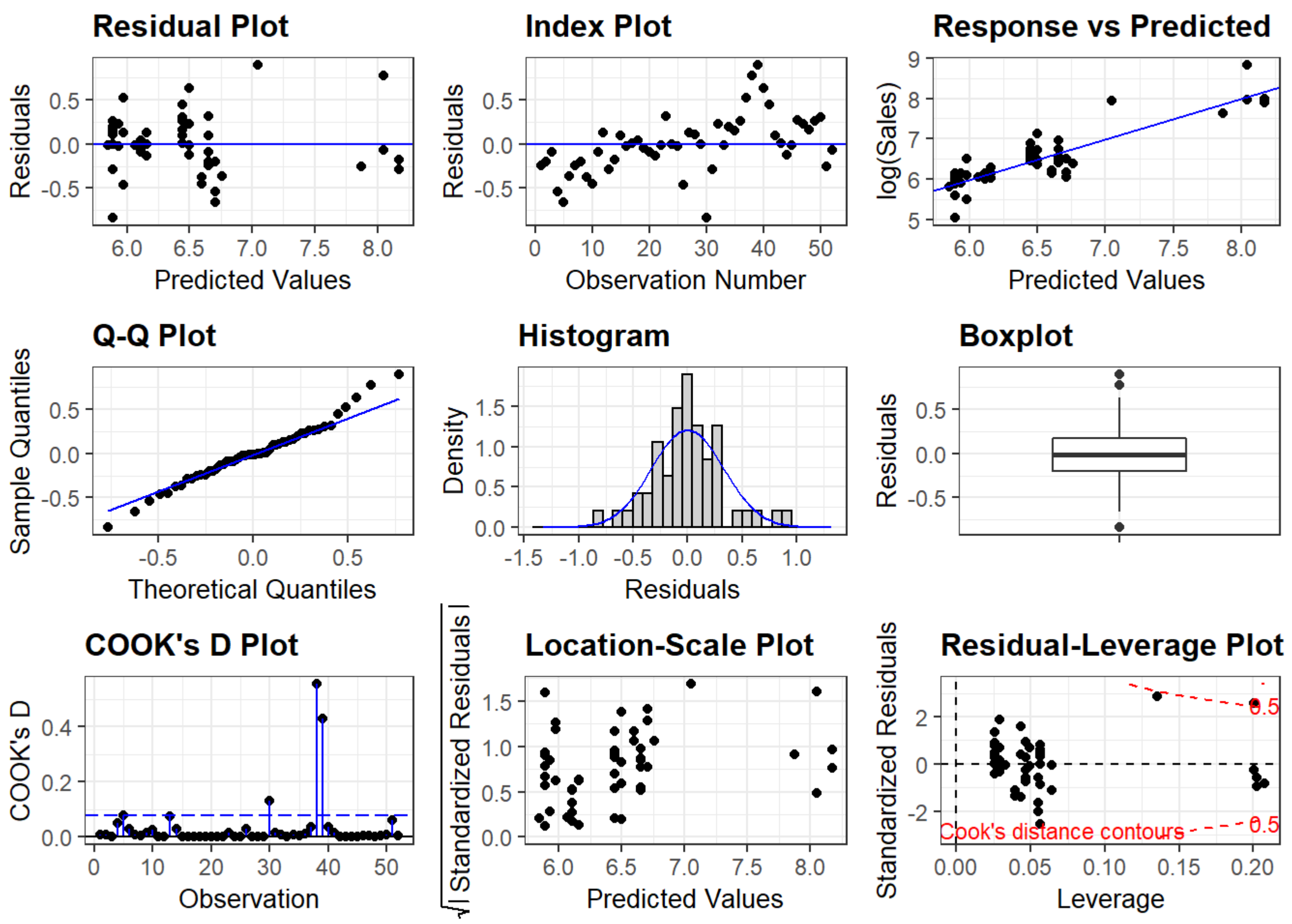
Ignoring the autocorrelation effect

DEMONSTRATION OF IGNORANCE OF AUTOCORRELATION

1.

build model without it (assume errors are independent)

```
lm.ignore<-lm(log(Sales)~log(Price)+Promotion, data=df)
resid_panel(lm.ignore,plots='all')
```



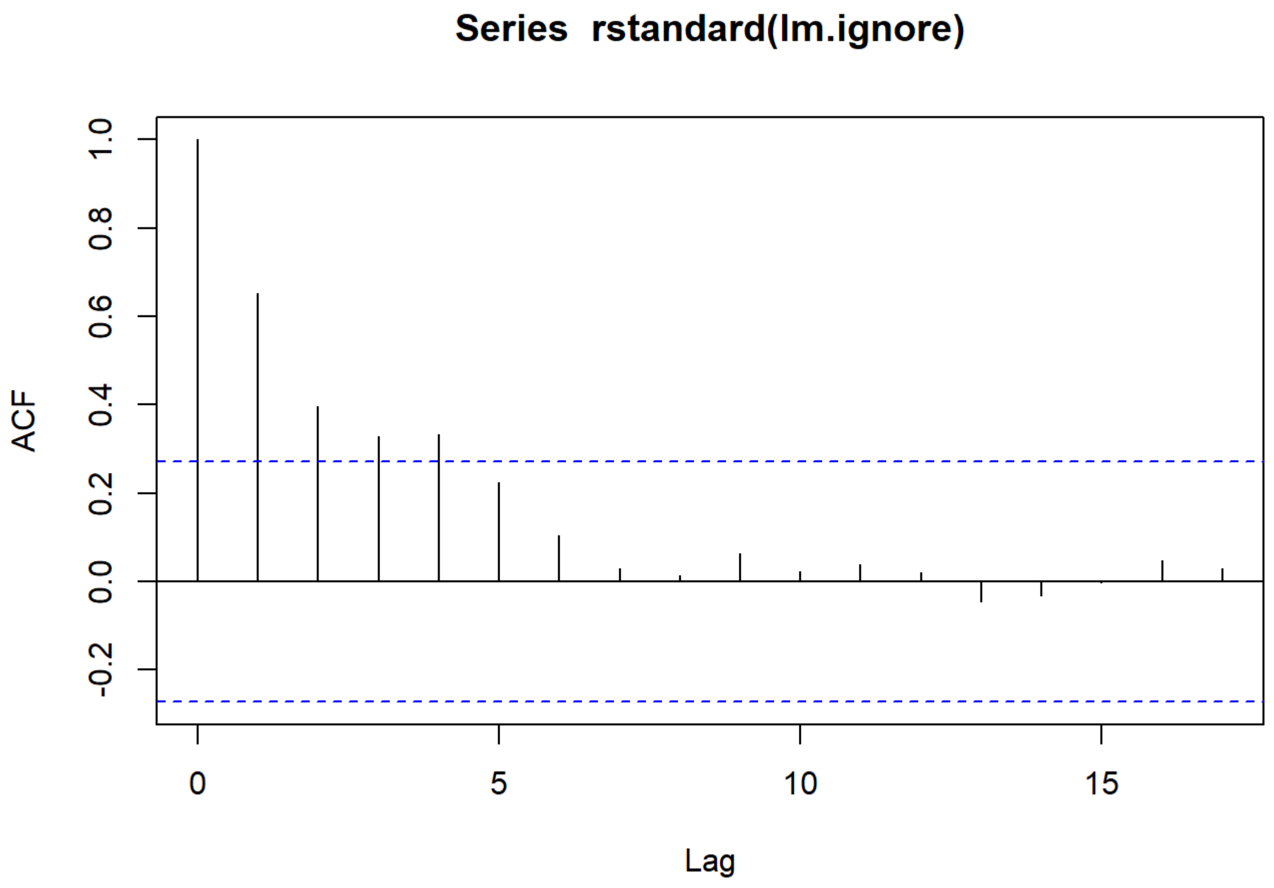
look at the *Index Plot*

highly organized residuals

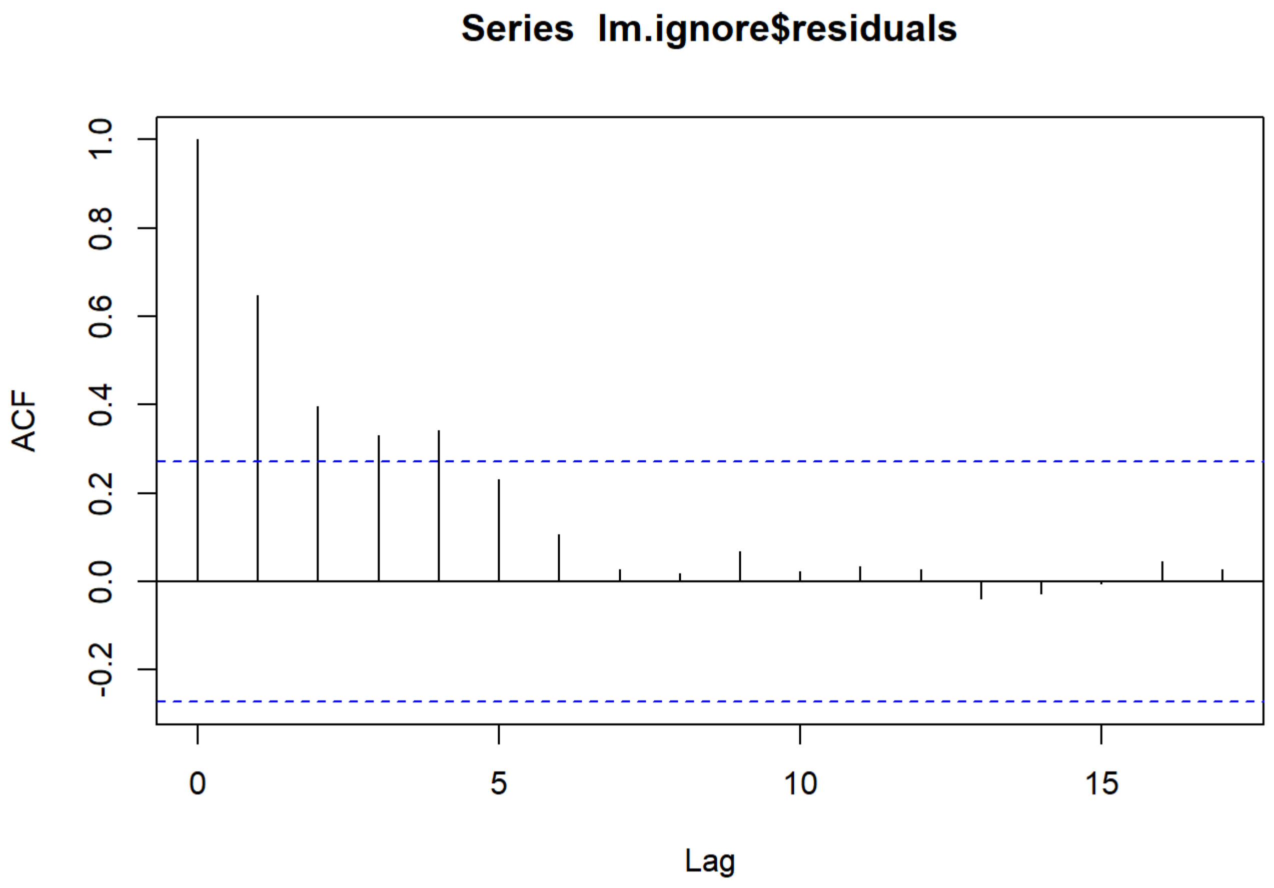
BOOK SAYS: postitive (negative) standardized residuals, followed by positive(negative) standardized residuals, thus there is *positive autocorrelation* present in the residuals.

To examine futher examine a plot of the autocorrelation function of the standardized residuals from the model

```
a<-acf(rstandard(lm.ignore),lag.max=17, plot=TRUE)
```



```
b<-acf(lm.ignore$residuals,lag.max=17, plot=TRUE)
```



my plots arent the same as the book….

using generalized least squares when the errors are AR(1)

examine methods based on generalized least squares which allow the errors to be autocorrelated (another name is serially correlated).

$$Y_t = \beta_0 + \beta_1 x_1 + e_t$$

where

$$e_t = \rho e_{t-1} + v_t$$

errors e_t follow an autoregressive process of order 1 (AR(1))

WHAT IS v_t?

it says v_t is iid (every residual is independent and identically distributed) with $N(0, \sigma^2)$ (mean zero and variance sigma^2)

the rho e_t-1 term is the signal in the variance from the previous week times a scalar + v_t which is the random variance

The expected value of e_t is rho * the expected value of the time series error + the truly random error = 0

$$E(e_t) = \rho E(e_{t-1}) + E(v_t) = 0$$

and the variance for e_t is the variance of rho^2variance e times the true varaince ^2

$$\sigma^2$$

since v_t is independent of e_t-1

$$\sigma_e^2 = \frac{\sigma_v^2}{1-\rho^2}$$

variance of the full error is equal to the variance of the truly random error/ 1- the constant rho^2

first order autocorrelation among the errors e_t is equal to rho (i guess the height of the value on the graph is the rho?)

$$Corr(e_t, e_{t-1}) = \rho$$

we can also show that the autocorrelation for any level

$$Corr(e_t, e_{t-l}) = \rho^l$$

when rho is less than 1, the correlations get smaller as l increases

MATH looking at the estimate for β_1

shows how the autocorrelation effects the estimate

conclusion

using least squares and ignoring the autocorrelation when it exists will result in consistent estimates of beta_1, but incorrect estimates of the variance of beta_1 least squares. Invalidating the confidence interval and hypothesis test.

Generalized least squares estimation

EX:

estimating the price elasticity of a food product continued

```
library(nlme)
```

```
##
## Attaching package: 'nlme'
```

```
## The following object is masked from 'package:dplyr':
##
##     collapse
```

```
gls.food<-gls(log(Sales)~log(Price)+Promotion+Week,data=df, correlation = corAR1(form = ~Week),m
ethod='ML')
summary(gls.food)
```

```
## Generalized least squares fit by maximum likelihood
##   Model: log(Sales) ~ log(Price) + Promotion + Week
##   Data: df
##        AIC     BIC   logLik
##   6.537739 18.2452 2.731131
##
## Correlation Structure: AR(1)
##  Formula: ~Week
##  Parameter estimate(s):
##       Phi
## 0.5503593
##
## Coefficients:
##                  Value Std.Error   t-value p-value
## (Intercept)  4.675667 0.2383703 19.615142   0.000
## log(Price)  -4.327391 0.5625564 -7.692368   0.000
## Promotion    0.584650 0.1671113  3.498565   0.001
## Week         0.012517 0.0046692  2.680813   0.010
##
##  Correlation:
##            (Intr) lg(Pr) Promtn
## log(Price)  0.807
## Promotion   0.559  0.682
## Week       -0.625 -0.157 -0.206
##
## Standardized residuals:
##        Min         Q1        Med         Q3        Max
## -2.9473082 -0.6095076  0.1031472  0.5769989  2.9558179
##
## Residual standard error: 0.2740294
## Degrees of freedom: 52 total; 48 residual
```

This isn't exactly the output in the book…

the AIC BIC and logLik are all significantly different.

the Coefficients st error t-value and p-value are all pretty close

the residual standard error in mine is a little higher…

the phi is a little higher..

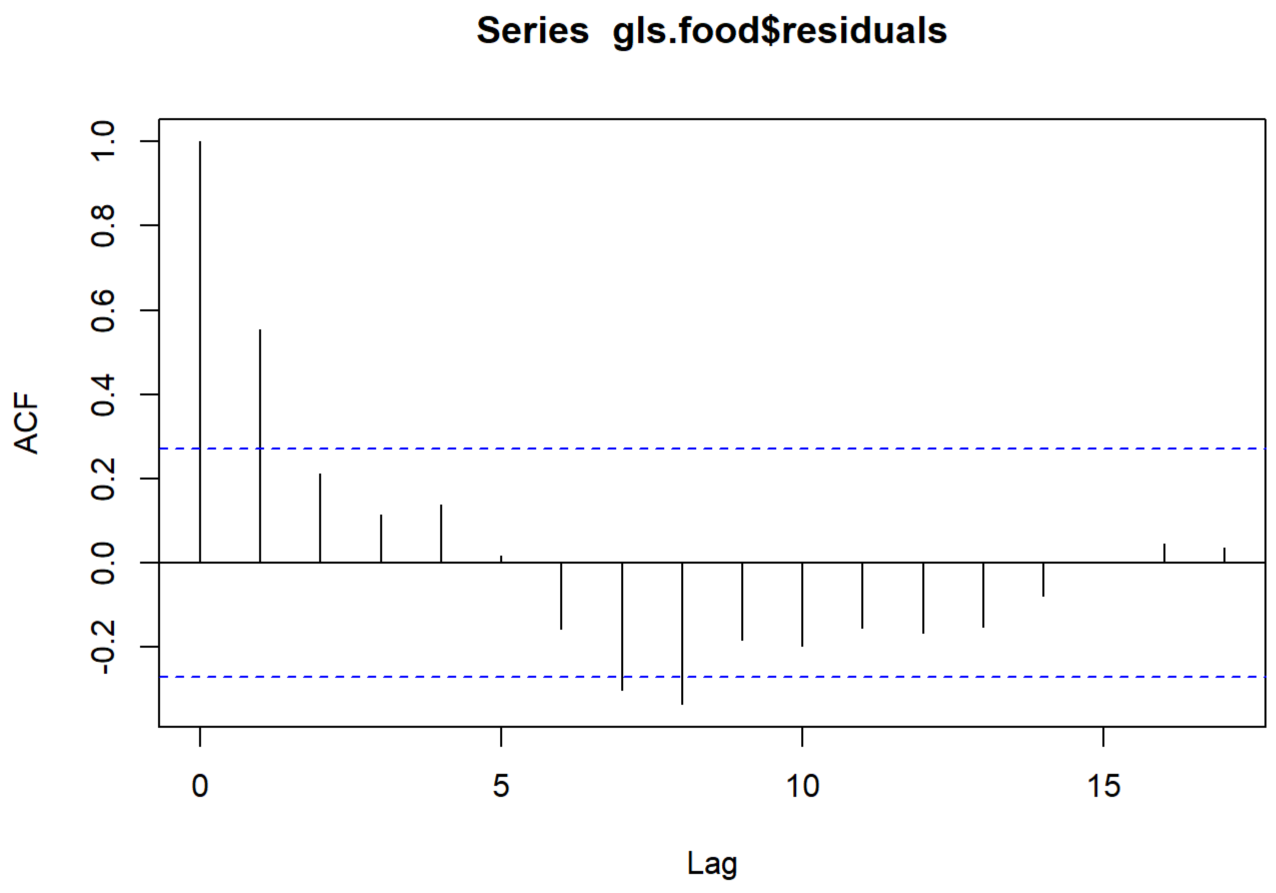
IT IS RIGHT IF WE PUT METHOD='ML'

```
nlme::intervals(gls.food)
```

```
## Approximate 95% confidence intervals
##
##  Coefficients:
##                    lower        est.       upper
## (Intercept)  4.196391300  4.67566686  5.15494243
## log(Price)  -5.458486702 -4.32739122 -3.19629575
## Promotion    0.248649971  0.58464986  0.92064974
## Week         0.003129195  0.01251724  0.02190529
## attr(,"label")
## [1] "Coefficients:"
##
##  Correlation structure:
##         lower      est.     upper
## Phi 0.2867307 0.5503593 0.7365028
## attr(,"label")
## [1] "Correlation structure:"
##
##  Residual standard error:
##     lower      est.     upper
## 0.2113301 0.2740294 0.3553309
```

Output associated with fitting model using maximum liklihood and assuming the errors are AR(1)

```
acf(gls.food$residuals)
```



Looking at the ACF for the residuals (this time they look the same as the book) we see that the lag 1 autocorrelation at around .6 is highly statistically significant for the GLS residuals

the high positive autocorrelation that is in the data can still produce nonrandom patterns in the diagnostic plots based on these residuals, even when the fitted model accounts for them and is correct

We will transform the model with the AR(1) errors into a related model with uncorrelated errors so we can use the diagnostic plots

transforming a model with AR(1) errors into a model with iid Errors

transform e_t to $\rho e_{t-1} + v_t$

MATH

we get

$$Y_t - \rho Y_{t-1} = (1-\rho)\beta_0 + \beta_1(x_t - \rho x_{t-1}) + v_t$$

rewrite as

$$Y_t^* = \beta_0 x_{t1}^* + \beta_1 x_{t2}^* + v_t \ : t = 2, \dots., n$$

this is called the *Cochrane-Orcutt transformation*

since it is only valid for t=2…n we need to deal with the first observation

MATH (not sure if this matters)

General approach to transforming GLS to LS

MATH:

recall:

you use thevariance-covariance matrix of the errors multiplied in a certain way with the matrix X (predictors) and Y (dependent variable)

$\sum$ is the variance-covariance matrix, where each element is equal to the covariance of errors between the predictor of that row/column interaction (recall variance within the error of a predictor is along the diagonal)

You can decompose a matrix into upper and lower triangular matrices

so

$$\sum = SS'$$

where S is the lower triangular matrix with posititve diagonal entries

this is the Cholesky decompostion of $\sum$

it is the ‘square root’ of the variance-covariance matrix

multiply your model through by the INVERSE of S and look at the error term’s variance

$$Var(S^{-1}e = I$$

(the identity matrix)

this means *pre multiplying by S (the square root of the variance-covariance matrix of errors) produces a linear model with uncorrelated errors.*

now we can obtain the GLS estimate of the betas using least squares

MATH to prove the betas for the least squares=betas for generalized least squares

EX:

estimating price elasticity of a food product

assume errors are AR(1) using least squares based on the transformed versions of the response and predictor variables

MY TRY:

how to calculate chocrane-orcutt procedure

1.

standard linear regression, we will be doing operations on it

```
confood2<-read.table(here('data','confood2.txt'),header=TRUE)
g <- lm(log(Sales)~log(Price)+Promotion+Week,data=confood2)
```

2.

run a generalized least squares to get the rho value (‘its called phi in the output’)

```
m1 <- gls(log(Sales)~log(Price)+Promotion+Week,correlation=corAR1(form=~Week),data=confood2,meth
od="ML")
summary(m1)
```

```
## Generalized least squares fit by maximum likelihood
##   Model: log(Sales) ~ log(Price) + Promotion + Week
##   Data: confood2
##        AIC     BIC   logLik
##   6.537739 18.2452 2.731131
##
## Correlation Structure: AR(1)
##  Formula: ~Week
##  Parameter estimate(s):
##       Phi
## 0.5503593
##
## Coefficients:
##                  Value Std.Error   t-value p-value
## (Intercept)  4.675667 0.2383703 19.615142   0.000
## log(Price)  -4.327391 0.5625564 -7.692368   0.000
## Promotion    0.584650 0.1671113  3.498565   0.001
## Week         0.012517 0.0046692  2.680813   0.010
##
##  Correlation:
##            (Intr) lg(Pr) Promtn
## log(Price)  0.807
## Promotion   0.559  0.682
## Week       -0.625 -0.157 -0.206
##
## Standardized residuals:
##        Min         Q1        Med         Q3        Max
## -2.9473082 -0.6095076  0.1031472  0.5769989  2.9558179
##
## Residual standard error: 0.2740294
## Degrees of freedom: 52 total; 48 residual
```

```
#set Phi by looking, cant figure out how to get it out of the summary
rho<-.5504
```

3.

make a model matrix for the basic linear model

```
x<-model.matrix(g)
```

4.

build the Cochrane Orcutt diagonal matrix.

```
Sigma <- diag(length(confood2$Week))
#this is the variance,covariane matrix for the errors
Sigma <- rho^abs(row(Sigma)-col(Sigma))

#get the S matrix, which is the lower triangular decomposition of Sigma
sm <- chol(Sigma)
smi <- solve(t(sm))
```

4.

build transformed matrices from the S matrix, accounting for autocorrelation

```
xstar <- smi %*% x
ystar <- smi %*% log(confood2$Sales)
```

5.

fit simple linear regression with transformed variables

```
m1tls <- lm(ystar ~ xstar-1)
summary(m1tls)
```

```
##
## Call:
## lm(formula = ystar ~ xstar - 1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.99263 -0.12518  0.01946  0.13055  0.67210
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## xstar(Intercept)  4.67566    0.23838  19.614  < 2e-16 ***
## xstarlog(Price)  -4.32741    0.56256  -7.692 6.44e-10 ***
## xstarPromotion    0.58464    0.16711   3.499  0.00102 **
## xstarWeek         0.01252    0.00467   2.681  0.01004 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2852 on 48 degrees of freedom
## Multiple R-squared:  0.9943, Adjusted R-squared:  0.9939
## F-statistic:  2106 on 4 and 48 DF,  p-value: < 2.2e-16
```

now we can trust these R^2 and p-values, they account for autocorrelation

Case study

data demonstrates hazards with ignoring autocorrelation in fitting and when examining diagnostics

EX:

saving and loan associations in bay area have monopoly in residential real estate in 1990s

banks had a small portion

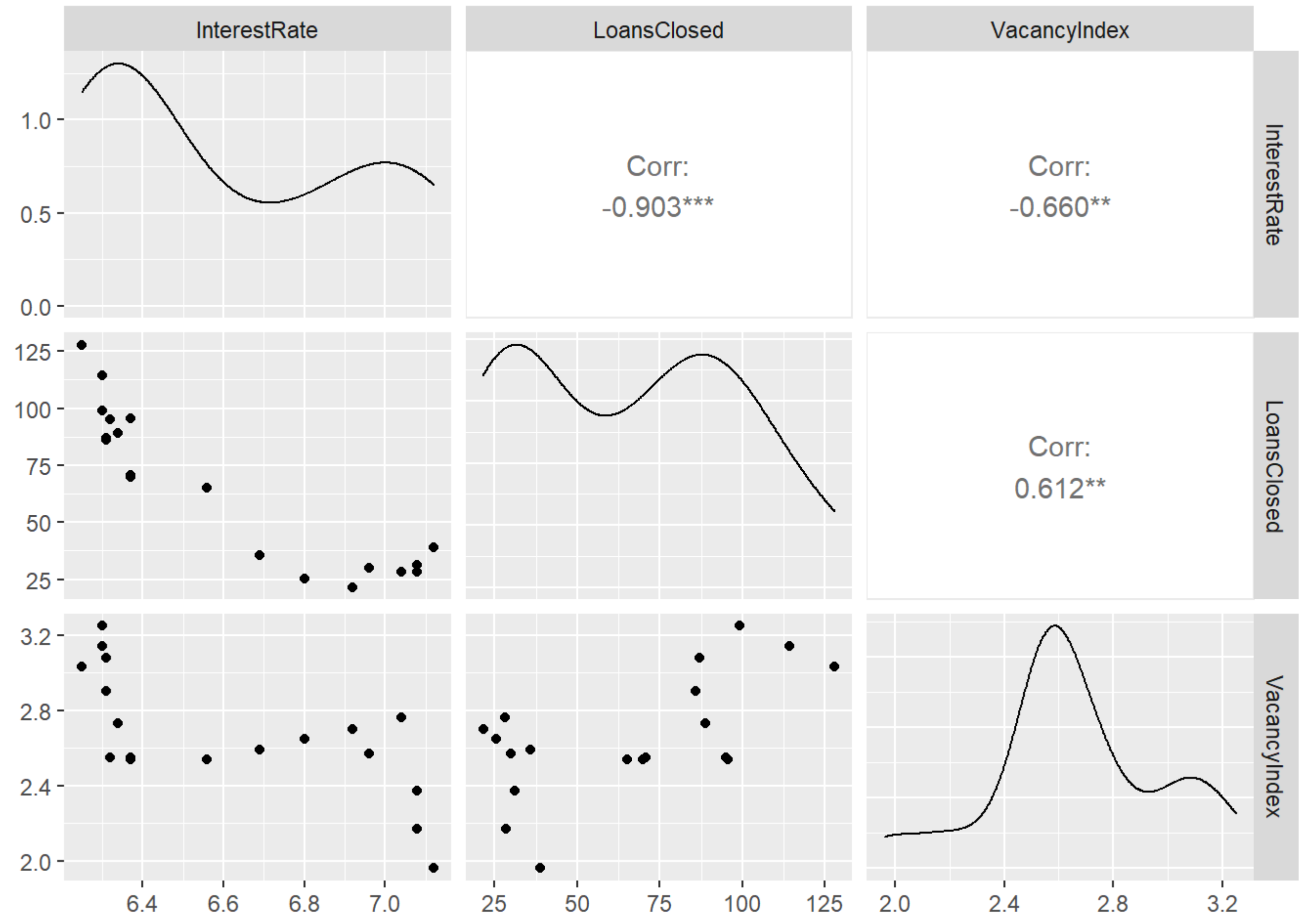
Develop a regression model to predict interest rates (Y) from x_1(amount of loands closed in MM)

x_2 (vacancy index)

```
file<-here('data','BayArea.txt')
bay_area<-read.table(file,header=TRUE)
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
ggpairs(bay_area%>%select(c(InterestRate,LoansClosed,VacancyIndex)))
```

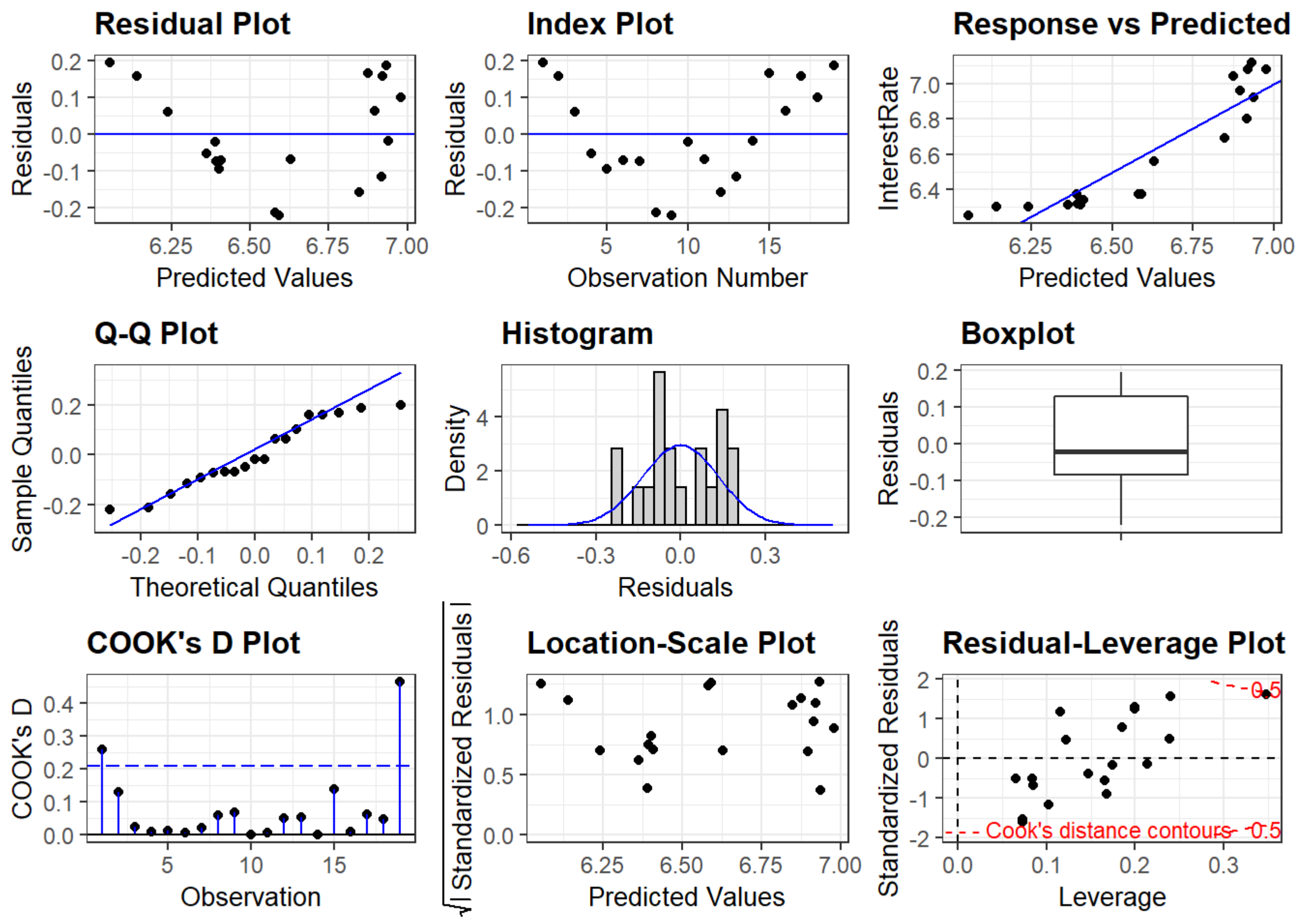


FIT A MODEL WITHOUT INCLUDING AUTOCORRELATION

```
summary(lm.bay_ignore<-lm(InterestRate~LoansClosed+VacancyIndex,data=bay_area))
```

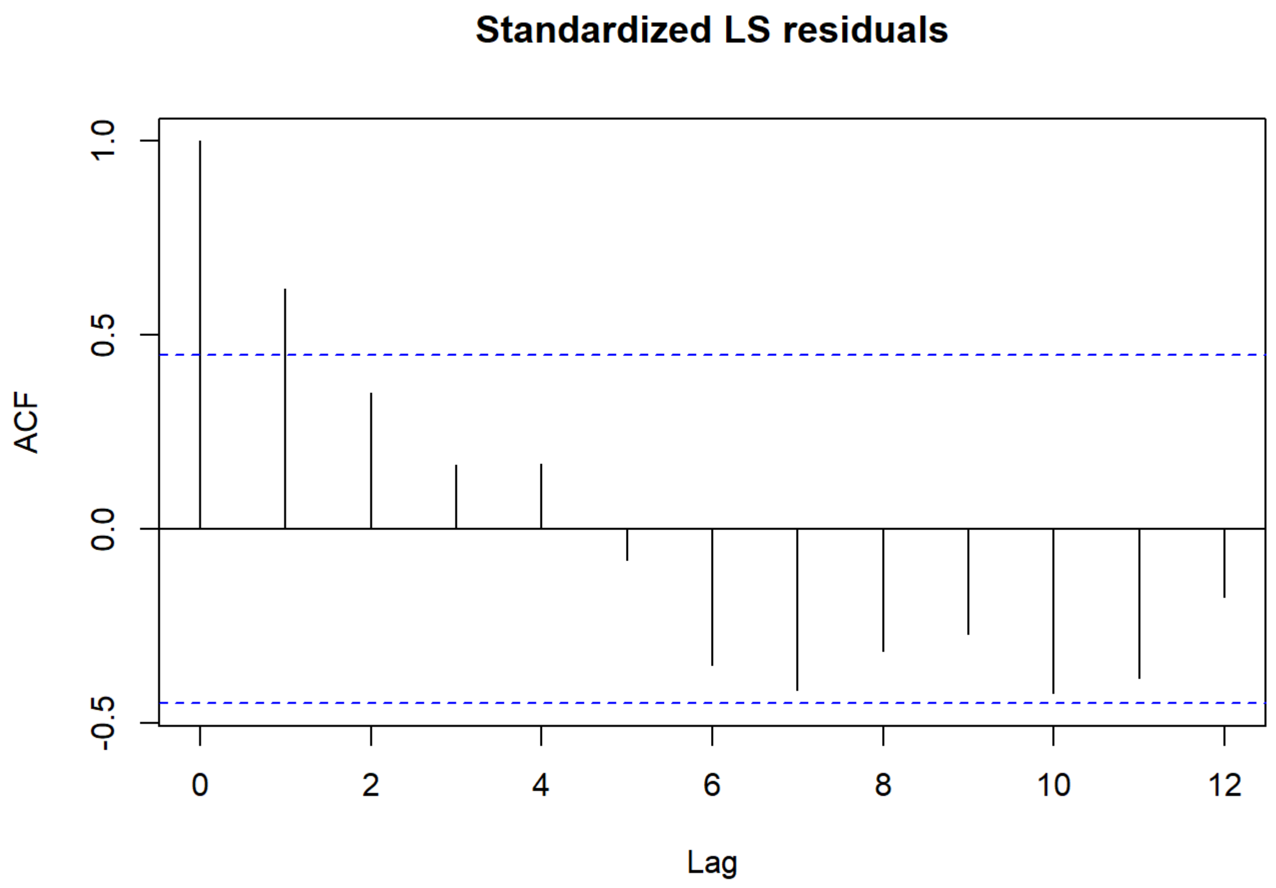
```
##
## Call:
## lm(formula = InterestRate ~ LoansClosed + VacancyIndex, data = bay_area)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.22079 -0.08295 -0.01989  0.13077  0.19638
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.580687   0.312308   24.27 4.74e-14 ***
## LoansClosed  -0.007756   0.001249   -6.21 1.25e-05 ***
## VacancyIndex -0.176577   0.132733   -1.33    0.202
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1432 on 16 degrees of freedom
## Multiple R-squared:  0.8344, Adjusted R-squared:  0.8137
## F-statistic: 40.31 on 2 and 16 DF,  p-value: 5.651e-07
```

```
resid_panel(lm.bay_ignore, plots='all')
```



residuals are highly nonrandom with a quadratic pattern, *obvious autocorrelation* among the standardized residuals

```
acf(rstandard(lm.bay_ignore), lag.max = 12,plot = TRUE,main='Standardized LS residuals')
```



modeling the autocorrelation effect as AR(1)

THIS IS RIGHT, YOU HAVE TO PUT IN METHOD='ML' for maximum likelihood

```
gls.bay<-gls(data=bay_area, InterestRate~LoansClosed+VacancyIndex,correlation = corAR1(form = ~M
onth) ,method='ML')
summary(gls.bay)
```

```
## Generalized least squares fit by maximum likelihood
##   Model: InterestRate ~ LoansClosed + VacancyIndex
##   Data: bay_area
##         AIC       BIC   logLik
##   -35.30833 -30.58613 22.65416
##
## Correlation Structure: AR(1)
##  Formula: ~Month
##  Parameter estimate(s):
##       Phi
## 0.9572093
##
## Coefficients:
##                  Value Std.Error   t-value p-value
## (Intercept)   7.122990 0.4182065 17.032232  0.0000
## LoansClosed  -0.003432 0.0011940 -2.874452  0.0110
## VacancyIndex -0.076340 0.1307842 -0.583710  0.5676
##
##  Correlation:
##              (Intr) LnsCls
## LoansClosed  -0.316
## VacancyIndex -0.822  0.117
##
## Standardized residuals:
##        Min         Q1        Med         Q3        Max
## -1.3439744 -1.1533069 -0.8047658  0.4467079  1.1783687
##
## Residual standard error: 0.2377426
## Degrees of freedom: 19 total; 16 residual
```

```
nlme::intervals(gls.bay)
```

```
## Approximate 95% confidence intervals
##
##  Coefficients:
##                     lower         est.         upper
## (Intercept)   6.236431638  7.122989795  8.0095479516
## LoansClosed  -0.005963412 -0.003432182 -0.0009009516
## VacancyIndex -0.353590009 -0.076339971  0.2009100658
## attr(,"label")
## [1] "Coefficients:"
##
##  Correlation structure:
##         lower      est.     upper
## Phi 0.5284235 0.9572093 0.9969063
## attr(,"label")
## [1] "Correlation structure:"
##
##  Residual standard error:
##      lower       est.      upper
## 0.06868792 0.23774259 0.82287451
```

Use process to assume the errors are AR(1) using least squares based on the transformed versions of predictor and response

how to calculate chocrane-orcutt procedure

1.

store residuals from simple lm in RESI

```
RESI<-lm.bay_ignore$residuals
```

2.

calculate a lag-1 residual variable lagRESI

(i guess this is from a gls with the correlation set to AR(1) and method ='ML')

```
lagRESI<-gls.bay$residuals
```

3.

fit a simple linear regression with response RESI and predictor lagRESI and *no intercept*

```
lm.resi<-lm(RESI~lagRESI)
summary(lm.resi)
```

```
##
## Call:
## lm(formula = RESI ~ lagRESI)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.148074 -0.064598  0.000173  0.038518  0.239711
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.03542    0.02715   1.305  0.20944
## lagRESI      0.38850    0.11821   3.286  0.00436 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1086 on 17 degrees of freedom
## Multiple R-squared:  0.3885, Adjusted R-squared:  0.3525
## F-statistic:  10.8 on 1 and 17 DF,  p-value: 0.004356
```

```
#get the slope out, this is rho

rho<-lm.resi$coefficients[2]
```

4.

get Y_star and X_star

```
Y_star<-bay_area$InterestRate-rho*lag(bay_area$InterestRate,1)
xstarLoansClosed<-bay_area$LoansClosed-lag(rho*bay_area$LoansClosed,1)
xstarVacancyIndex<-bay_area$VacancyIndex-lag(rho*bay_area$VacancyIndex,1)
```

fit simple linear regression with transformed variables

THIS IS WRONG

```
summary(lm_auto<-lm(Y_star~xstarLoansClosed+xstarVacancyIndex))
```

```
##
## Call:
## lm(formula = Y_star ~ xstarLoansClosed + xstarVacancyIndex)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.17880 -0.05496 -0.01367  0.06172  0.19024
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        4.770081   0.188060  25.365 9.85e-14 ***
## xstarLoansClosed  -0.007708   0.001389  -5.551 5.55e-05 ***
## xstarVacancyIndex -0.266353   0.125994  -2.114   0.0517 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1053 on 15 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.7806, Adjusted R-squared:  0.7514
## F-statistic: 26.69 on 2 and 15 DF,  p-value: 1.145e-05
```

```
df_auto<-data.frame(Y_star,xstarLoansClosed,xstarVacancyIndex)
ggpairs(df_auto)
```

```
## Warning: Removed 1 rows containing non-finite values (stat_density).
```

```
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value
```

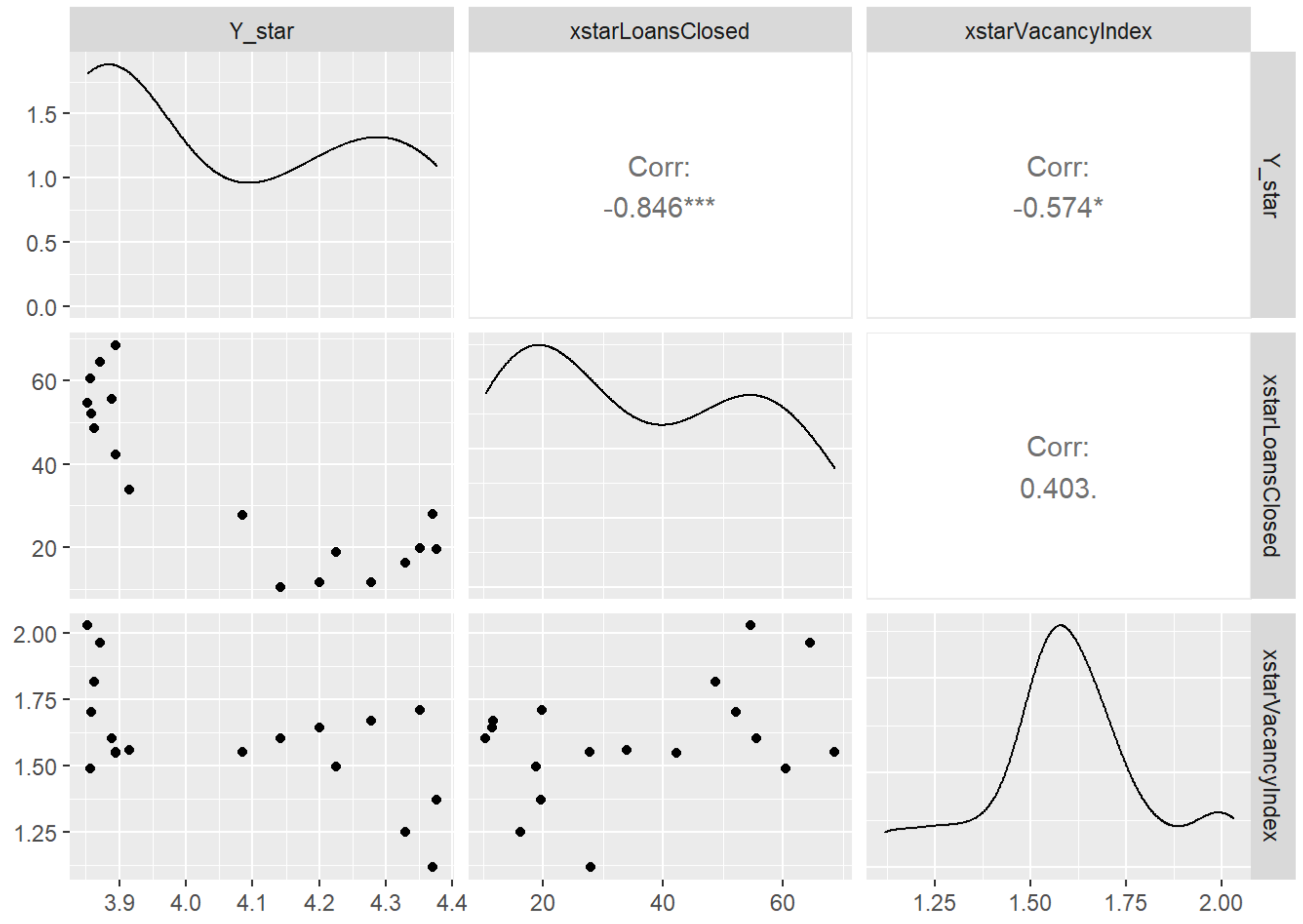
```
## Warning: Removed 1 rows containing missing values (geom_point).
```

```
## Warning: Removed 1 rows containing non-finite values (stat_density).
```

```
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value
## Warning: Removed 1 rows containing missing values (geom_point).

## Warning: Removed 1 rows containing missing values (geom_point).
```

```
## Warning: Removed 1 rows containing non-finite values (stat_density).
```


Y_star
xstarLoansClosed
xstarVacancyIndex
Corr:
-0.846***
Corr:
-0.574*
Corr: