Module 2 Time Series Exercises

Filipp Krasovsky

11/8/2021

3.1 [10 points] (Structural Regression Model). For the Johnson and Johnson data - In this problem, we are going to fit a special type of structural model, where T is a trend component, S is a season component, and N is the noise. In our case, time t is in quarters (1960.00, 1960.25) so one unit of time is a year.

1. Fit the regression model: xt=Bt+a1Q1 + a2Q2 + a3Q3 + a4Q4 + wt if time t corresponds to quarter i - 1,2,3,4 and zero otherwise. The Q’s are called indicator variables. We will assume for now that wt is a Gaussian white noise sequence. Hint: Detailed code is given in Appendix A, near the end of Section A.5.

require(astsa)

## Loading required package: astsa

require(tseries)

## Loading required package: tseries

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

jj = astsa::jj;  
trend = time(jj)-1970  
Q = factor(cycle(jj))  
  
#regress the log of JJ earnings against no inercept, 4 dummy variables for quarter, and the trend.  
reg = lm(log(jj) ~ 0 + trend + Q, na.action=NULL)  
#observe outcome  
summary(reg)

##   
## Call:  
## lm(formula = log(jj) ~ 0 + trend + Q, na.action = NULL)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.29318 -0.09062 -0.01180 0.08460 0.27644   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## trend 0.167172 0.002259 74.00 <2e-16 \*\*\*  
## Q1 1.052793 0.027359 38.48 <2e-16 \*\*\*  
## Q2 1.080916 0.027365 39.50 <2e-16 \*\*\*  
## Q3 1.151024 0.027383 42.03 <2e-16 \*\*\*  
## Q4 0.882266 0.027412 32.19 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.1254 on 79 degrees of freedom  
## Multiple R-squared: 0.9935, Adjusted R-squared: 0.9931   
## F-statistic: 2407 on 5 and 79 DF, p-value: < 2.2e-16

1. If the model is correct, what is the estimated annual increase in the logged earnings per share?

sum(reg$coefficients[2:length(reg$coefficients)])

## [1] 4.167

Presumably, if we count the estimated annual increase as the combination of all four seasonal coefficients, then there should be about a 4.16-unit increase in the log earnings of J&J per share.

1. If the model is correct, does the average logged earnings rate increase or decrease from the third quarter to the fourth quarter. And, by what percentage does it increase or decrease?

There is an average 23% decrease in the logged earnings rate.

coefs = as.numeric(reg$coefficients)  
100\*(coefs[5]-coefs[4])/coefs[4]

## [1] -23.34944

1. What happens if you include an intercept term in the model in 1.? Explain why there was a problem.

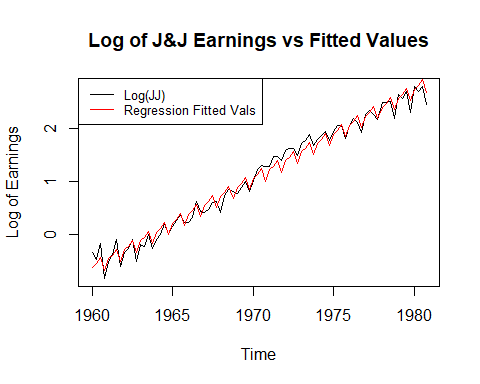
reg2=lm(log(jj) ~ trend + Q, na.action=NULL)  
summary(reg2)

##   
## Call:  
## lm(formula = log(jj) ~ trend + Q, na.action = NULL)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.29318 -0.09062 -0.01180 0.08460 0.27644   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.052793 0.027359 38.480 < 2e-16 \*\*\*  
## trend 0.167172 0.002259 73.999 < 2e-16 \*\*\*  
## Q2 0.028123 0.038696 0.727 0.4695   
## Q3 0.098231 0.038708 2.538 0.0131 \*   
## Q4 -0.170527 0.038729 -4.403 3.31e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.1254 on 79 degrees of freedom  
## Multiple R-squared: 0.9859, Adjusted R-squared: 0.9852   
## F-statistic: 1379 on 4 and 79 DF, p-value: < 2.2e-16

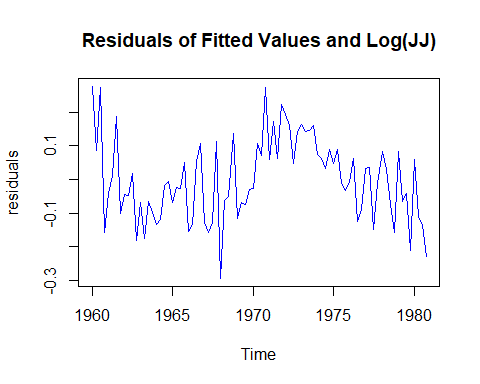
If we include an intercept in the original model, we lose the ability to meaningful interpret the model. In particular, the coefficient for the Q2 indicator variable stops being statistically significant, and the coefficient for Q1 disappears entirely.

1. Graph the data and superimpose the fitted values on the graph. Examine the residuals and state your conclusions. Does it appear that the model fits the data well (do the residuals look white?).

plot(log(jj),main = "Log of J&J Earnings vs Fitted Values",xlab="Time",ylab="Log of Earnings")  
lines(reg$fitted.values,col="red")  
legend(x="topleft",legend=c("Log(JJ)","Regression Fitted Vals"),col=c("black","red"),cex=0.8,lty=1)



residuals = log(jj) - reg$fitted.values #or, if you prefer, reg$residuals   
plot(residuals,main="Residuals of Fitted Values and Log(JJ)",col="blue")



adf.test(residuals)

##   
## Augmented Dickey-Fuller Test  
##   
## data: residuals  
## Dickey-Fuller = -1.1176, Lag order = 4, p-value = 0.9144  
## alternative hypothesis: stationary

Based on the augmented dickey-fuller test, we do not have enough evidence to reject the null hypothesis that the residuals are non-stationary.

3.2 (a)For the mortality data in 3.5, add another component to regression 3.17 that accounts for the particulate count 4 weeks prior (Pt-4). State your conclusion.

#use dataset cmort  
#first, create the original model  
  
temp = tempr - mean(tempr)   
temp2 = temp\*\*2  
trend = time(cmort)  
fit = lm(cmort ~ trend + temp + temp2 + part, na.action=NULL)  
summary(fit)

##   
## Call:  
## lm(formula = cmort ~ trend + temp + temp2 + part, na.action = NULL)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -19.0760 -4.2153 -0.4878 3.7435 29.2448   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.831e+03 1.996e+02 14.19 < 2e-16 \*\*\*  
## trend -1.396e+00 1.010e-01 -13.82 < 2e-16 \*\*\*  
## temp -4.725e-01 3.162e-02 -14.94 < 2e-16 \*\*\*  
## temp2 2.259e-02 2.827e-03 7.99 9.26e-15 \*\*\*  
## part 2.554e-01 1.886e-02 13.54 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 6.385 on 503 degrees of freedom  
## Multiple R-squared: 0.5954, Adjusted R-squared: 0.5922   
## F-statistic: 185 on 4 and 503 DF, p-value: < 2.2e-16

#next, add the part-4 component.  
cmort2 = ts.intersect(cmort,part,temp,part4=lag(part,-4))  
trend = time(cmort2)  
temp = cmort2[,3] - mean(cmort2[,3])  
temp2 = temp\*\*2  
fit2 = lm(cmort ~ trend + temp + temp2 + part + part4,data=cmort2,na.action = NULL)  
summary(fit2)

##   
## Call:  
## lm(formula = cmort ~ trend + temp + temp2 + part + part4, data = cmort2,   
## na.action = NULL)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -18.228 -4.314 -0.614 3.713 27.800   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.808e+03 1.989e+02 14.123 < 2e-16 \*\*\*  
## trend -1.385e+00 1.006e-01 -13.765 < 2e-16 \*\*\*  
## temp -4.039e-01 3.525e-02 -11.458 < 2e-16 \*\*\*  
## temp2 2.155e-02 2.803e-03 7.688 8.02e-14 \*\*\*  
## part 2.029e-01 2.266e-02 8.954 < 2e-16 \*\*\*  
## part4 1.030e-01 2.485e-02 4.147 3.96e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 6.287 on 498 degrees of freedom  
## Multiple R-squared: 0.608, Adjusted R-squared: 0.6041   
## F-statistic: 154.5 on 5 and 498 DF, p-value: < 2.2e-16

On face value, it doesn’t appear that adding in a P(t-4) lag term improved model performance.

1. Using AIC, BIC is the model in (a) an improvement over the final model in example 3.5?

print(paste(AIC(fit),BIC(fit)))

## [1] "3332.28160799007 3357.66449667554"

print(paste(AIC(fit2),BIC(fit2)))

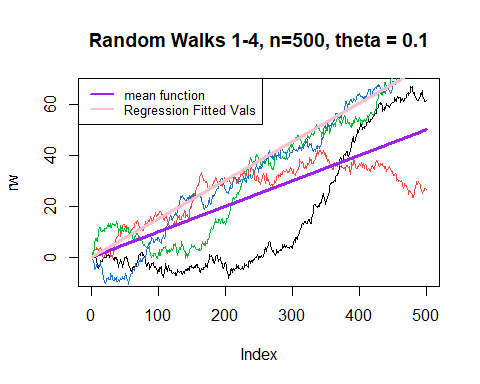
## [1] "3291.51955918572 3321.07759306222"

While, in technical terms, the AIC/BIC is lower for the new model, this improvement in performance is negligible.

3.3 - in this problerm we explore the difference between a random walk and a trend stationary process.

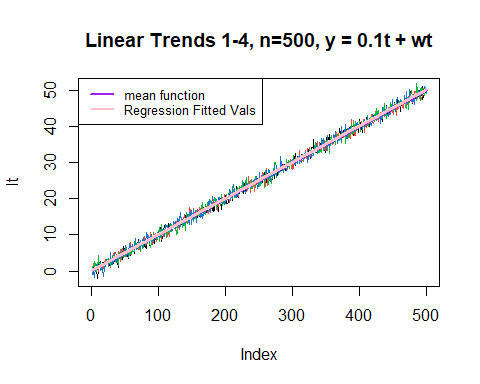
1. generate four series that are random walk with drift of length n = 500 with theta = 0.1 and a white noise variance of 1. call the data for t = 1:500. fit the regression Xt = Bt + Wt using least squares. plot the data, the true mean function (mu(t) = 0.1t), and the fitted line (xt = Bt) on the same graph.

set.seed(420)  
#create four random walks   
for (i in 1:4){  
 w = rnorm(500)  
 wd = w + 0.1  
 rw = cumsum(wd)  
   
 if(i==1){  
 plot(rw,main="Random Walks 1-4, n=500, theta = 0.1",col=i,type="l")  
 }else{  
 lines(rw,col=i)  
 }  
}  
  
#plot the true mean function   
meanFunc = 0.1 \* 1:500  
lines(meanFunc,col='purple',lwd=3)  
  
#plot the fitted line  
trend = time(rw)  
reg = lm(rw ~ 0 + trend,na.action=NULL)  
lines(reg$fitted.values,col='pink',lwd=3)  
  
#legend  
legend(x="topleft",legend=c("mean function","Regression Fitted Vals"),col=c("purple","pink"),cex=0.8,lty=1,lwd=2)



1. generate four series of n=500 that are linear trend plus noise, y = 0.1t + wt with wt~(0,1). fit the regression yt = bt+wt using least squares, plot the data, the true mean function, and the fitted line.

set.seed(420)  
n=500  
delta = 0.1  
#create four linear trends   
for (i in 1:4){  
 w = rnorm(n)  
 t = 1:n  
 lt = (t\*delta) + w  
   
 if(i==1){  
 plot(lt,main="Linear Trends 1-4, n=500, y = 0.1t + wt",col=i,type="l")  
 }else{  
 lines(lt,col=i)  
 }  
}  
  
#plot the true mean function   
meanFunc = delta \* 1:n  
lines(meanFunc,col='purple',lwd=3)  
  
#plot the fitted line  
trend = time(lt)  
reg = lm(lt ~ 0 + trend,na.action=NULL)  
lines(reg$fitted.values,col='pink',lwd=3)  
  
#legend  
legend(x="topleft",legend=c("mean function","Regression Fitted Vals"),col=c("purple","pink"),cex=0.8,lty=1,lwd=2)

 c) commentary: the variance of a linear trend with noise is relatively constant, insofar that the second time series is stationary around a trend. Furthermore, the difference between the fitted values and the mean function is much larger for the random walk because of the accumulation of white noise as t increases.

3.4 Consider a process consisting of a linear trend with an additive noise term consisting of independent random vars wt ~(0,sigma^2) Xt = B0 + B1t + Wt where B0 and B1t are fixed constants.

1. prove that Xt is nonstationary. E(Xt) = E(b0) + E(B1t) + E(wt) = B0 + B1t Because the B1 coefficient depends on the value of t, the function is nonstationary given that the mean is subject to change over time.
2. prove that the first difference is stationary by finding its mean and autocovariance function. First, we begin by defining Xt with two components:

Xt = Mu(t) + Yt where Mu(t) = B0 + B1t and Yt = Wt We can then reimagine Mu(t) as a random walk with drift:

Mu(t) = theta + Mu(t-1) + Wt, where Wt is independent of Yt.

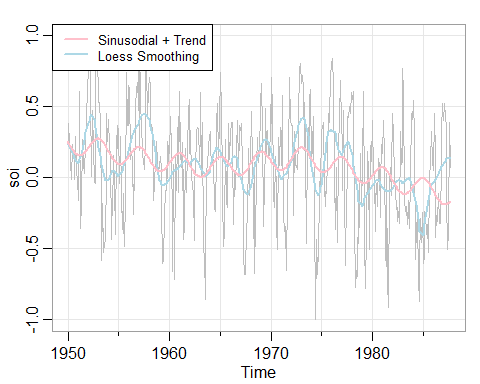
Thus: Xt = Mu(t) + Y(t) Xt = theta + Mu(t-1) + Wt + Yt Xt - (X(t-1)) = Mu(t) + Yt - Mu(t-1) - Y(t-1)

Simplify: B(Xt,1) = Mu(t-1) + wt + yt - Mu(t-1) - Y(t-1) = wt + yt - y(t-1) MEAN : E(B(xt,1)) = E(wt) + E(yt) - E(y(t-1)) = 0 + 0 + 0 AUTOCOVARIANCE: z = yt - y(t-1) Gamma(z) = cov(zt+h,zt) = cov(yt+h - y(t-1+h),yt - y(t-1)) = 2Gamma(h) - gamma(y)(h+1) - gamma(y)(h-1)

Therefore, the first difference is stationary because the mean and autocovariance do not depend on time.

3.8 In section 3.3 we saw that the El Nino/La Nina cycle was approximately 4 years. To investigate whether there is a strong 4-year cycle, compare a sinusoidal (one cycle every four years) fit to the Southern Oscillation Index to a lowess fit (as in Example 3.18). In the sinusoidal fit, include a term for the trend. Discuss the results.

#data = soi   
#lowess fit   
tsplot(soi,col='gray')  
lines(lowess(soi,f=.05),lwd=2,col='lightblue')  
#lines(lowess(soi),lty=2,lwd=2,col=2) #trend   
lo = predict(loess(soi ~ time(soi)),se=TRUE)  
trnd = ts(lo$fit, start = 1950,freq=12)  
#lines(trnd,col=6,lwd=2)  
  
#sinusodial + trend  
t <- time(soi)  
ssp <- spectrum(soi,plot=FALSE)   
per <- 4#1/ssp$freq[ssp$spec==max(ssp$spec)]  
reslm <- lm(soi ~trnd + sin(2\*pi/per\*t)+cos(2\*pi/per\*t))  
sinusodial\_fit <- ts(data=reslm$fitted.values,start=1950,frequency = 12)  
lines(sinusodial\_fit,col='pink',lwd=2)   
  
legend(x="topleft",legend=c("Sinusodial + Trend","Loess Smoothing"),col=c("pink","lightblue"),cex=0.8,lty=1,lwd=2)



#get actual model comparrison  
lm\_loess = loess(soi~time(soi))

#find sse  
sse\_loess <- sum(lm\_loess$residuals^2)  
sse\_sin <- sum(reslm$residuals^2)  
  
#find ssr  
ssr\_loess <- sum((lm\_loess$fitted)-mean(soi))  
ssr\_sin <- sum((reslm$fitted.values)-mean(soi))  
  
#find sst  
sst\_loess <- sse\_loess + ssr\_loess  
sst\_sin <- sse\_sin + ssr\_sin  
  
print(round(c(sse\_loess,ssr\_loess,sst\_loess),2))

## [1] 62.88 0.98 63.86

print(round(c(sse\_sin,ssr\_sin,sst\_sin),2))

## [1] 61.62 0.00 61.62

print(c(ssr\_loess/sst\_loess))

## [1] 0.01536841

print(c(ssr\_sin/sst\_sin))

## [1] 5.382381e-17

Based on SSE,SSR, and SST, it appears that the sinusodial model performs slightly better.