Time Series HW 3

Justin Gomez and Andrea Mack

**Disclaimer:** Justin and Andrea are NEW partners, we have never worked together in one of Dr. Greenwood's classes on an assignment until now.

## HW 3

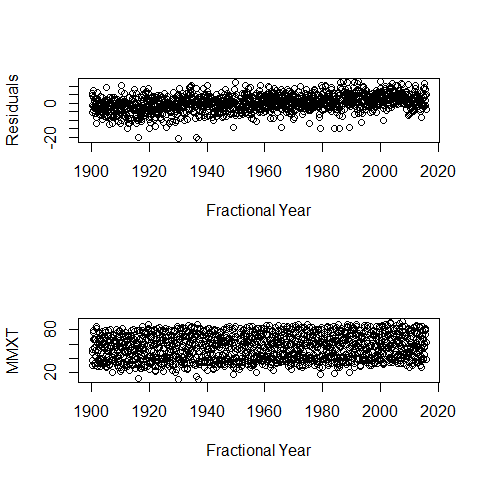
You can alone or in pairs that may or may not be people you worked with before. You can discuss it with old partners but should try work as much as possible with new collaborators. 5% bonus if you find someone completely new to work with - that you did not work with on first two assignments.

I mentioned de-seasonalizing of time series, where the seasonal variation is removed from the series to highlight variation at either higher or lower frequencies. There are a variety of techniques for doing this but the simplest is to just subtract the mean for each month from the observations. And the easiest way to find that is using lm(y~month,data=...).

1. For the Bozeman temperature data from HW 1 and 2, estimate a model with month only, subtract its fitted values from the responses (or just extract residuals). Plot the residuals vs the fractional Year variable and compare the plot of this result to the plot of the original time series.

When plotting the original MMXT by fractional year, there just appears to be random scatter. The plot of the residuals reduces the noise and is suggestive of an upward positive trend in mean MMXT over time. The plot of the residuals makes potential outliers more apparent than in plotting the original MMXT by fractional year.

par(mfrow=c(2,1))  
plot(lm1.resid ~ rawd$year.frac, xlab = "Fractional Year", ylab = "Residuals")  
  
plot(MMXT ~ year.frac, data = rawd, xlab = "Fractional Year", ylab = "MMXT")



1. In the de-seasonalized Bozeman temperature data set, re-assess evidence for the linear trend. Compare the result (test statistic, degrees of freedom and size of p-value) of just fitting a linear time trend in these de-seasonalized responses to the result from our original model with a linear year component and a month adjustment (not the quadratic trend model).

CHECK IF THIS IS WHAT HE WANTS

lm2 <- lm(lm1.resid ~ rawd$year)  
options(show.signif.stars = FALSE)  
lm2.sum <- summary(lm2)  
  
xtable(lm2.sum)

% latex table generated in R 3.3.1 by xtable 1.8-2 package % Mon Sep 19 11:33:35 2016

lm3 <- lm(temp ~ year + monthf, data = rawd)  
lm3.sum <- summary(lm3)  
  
xtable(lm3.sum)

% latex table generated in R 3.3.1 by xtable 1.8-2 package % Mon Sep 19 11:33:35 2016

1. I briefly discussed the HADCRUT data set in class. We will consider the HADCRUT4 series of temperature anomalies for the Nothern Hemisphere. The fully up to date data set is available at: <http://www.metoffice.gov.uk/hadobs/hadcrut4/data/current/time_series/HadCRUT.4.4.0.0.monthly_nh.txt>

Download the ensemble median monthly northern hemisphere temperature data. We will use the entire time series that is currently available (January 1850 to July 2016). You might want to read <http://www.metoffice.gov.uk/hadobs/hadcrut4/data/current/series_format.html> for more information on the columns in the data set.

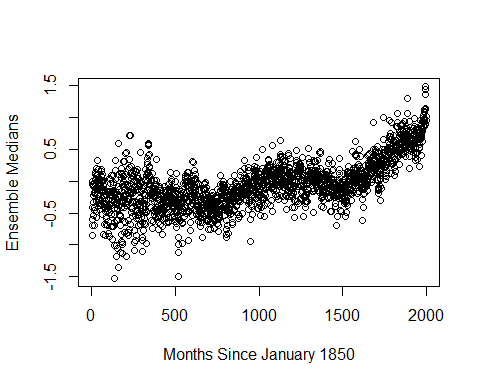
Because the time series is complete over the time frame under consideration, you can use ts() to create a time variable instead of messing around with their Year/Month variable.

Make a plot versus time of the ensemble medians and use that as your response variable in the following questions. Discuss trend, seasonality, outliers, and variability.

SO, to me the ensemble is the difference from the median temperature that is normal for that time of year. It's not clear to me whether the median is computed for each month before the difference is taken, or whether the overall median (regardless of month) is subtracted from each temperature value to get the ensemble variable.

DISCUSS

#col2 are the ensemble medians  
  
plot(hadcrut$V2 ~ hadcrut$date, xlab = "Months Since January 1850", ylab = "Ensemble Medians")

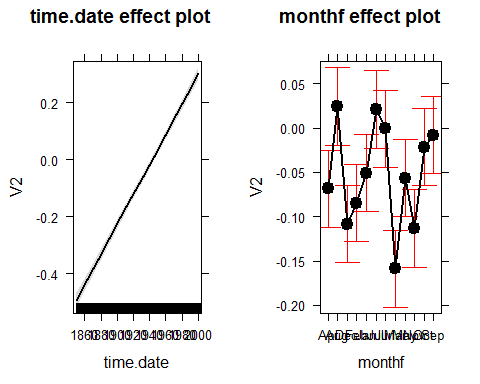


#an ensemble data set in which the 100 constituent  
#ensemble members sample the distribution of likely surface temperature anomalies given our  
#current understanding of these uncertainties  
#Sea Surface Temperature anomalies in degrees Celsius, or "SST anomalies" for short, are how much temperatures depart from what is normal for that time of year.

1. Our main focus with these data will be on estimating the long-term trend, starting with polynomial trend models. But first, check for seasonality in a model that accounts for a linear time trend. Use our previous fractional year for the trend. Report an effects plot and a test for the month model component. SEE COMMENT IN CODE CHUNK ALSO, DO WE USE FRACTIONAL YEAR AS IN PART (A) OR DO WE USE TS OBJECT?

Based on an Fstat of 6.74 on 11 and 1986 degrees of freedom, with an associated pvalue of $\<$ 0.001, there is strong evidence that the effect of date on median ensemble changes by month.

hadcrut$monthf <- as.factor(month.abb[as.numeric(substr(hadcrut$V1,6,7))])  
hadcrut$time.date <- time(hadcrut$date)  
lm4 <- lm(V2 ~ time.date + monthf, data = hadcrut)#don't need monthf as explanatory bc month already induced in time??  
  
plot(allEffects(lm4))



lm4.anova <- Anova(lm4, type = "II")  
print(xtable(lm4.anova))

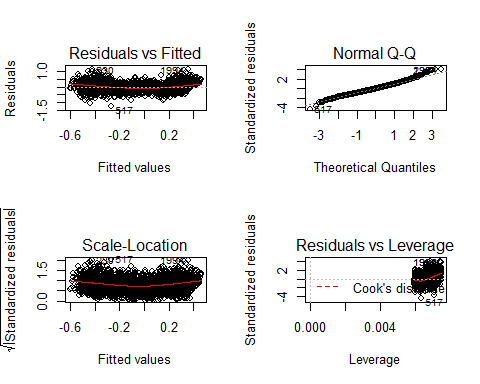
% latex table generated in R 3.3.1 by xtable 1.8-2 package % Mon Sep 19 11:33:36 2016

Note: when you use time() to generate the Year variable from a time series object it retains some time series object information that can cause conflicts later on. Create a new variable in your data.frame that uses something like as.vector(time(tsdataname)).

1. Check the residuals versus fitted values for any evidence of nonlinearity in the residuals vs fitted that was missed by the model with a linear trend and month component. Also note any potential issues with the constant variance assumption.

Viewing the residuals vs. fitted values plot below, there appears to be a quadratice relationship between the fitted values and residuals that was not captured by the model. For fitted values less than -0.3, there appears to be more variation in the residuals than for fitted values above -0.3. The model is less accurately predicting median ensembles when the original median ensemble was below -0.3, which to me means that we may have left out a variable or structure that explains when median ensemble was lower than expected.

par(mfrow = c(2,2))  
plot(lm4)



1. You can add higher order polynomial terms to models using x1+I(x1^2)+I(x1^3)... or using the poly function, such as poly(x1,3,raw=T) for a cubic polynomial that includes the linear and quadratic components (we want this!). The raw=T keeps the variables in their raw or input format. Estimate the same model but now using polynomial trends that steps up from linear (poly(time,1,raw=T)) and stop when you get a failure to estimate a part of the model. Briefly discuss what happened.

When I got to the fifth degree polynomial, the estimate could not be computed. WHY? My guess is that it has something to do with the seasonality of the time.data variable, or it be perfectly collinear with another variable.

lm.d1 <- lm(V2~poly(time.date,1,raw=TRUE) + monthf, data = hadcrut)  
summary(lm.d1)

##   
## Call:  
## lm(formula = V2 ~ poly(time.date, 1, raw = TRUE) + monthf, data = hadcrut)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.23925 -0.19175 -0.02096 0.18253 1.13640   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -1.030e+01 2.586e-01 -39.815 < 2e-16  
## poly(time.date, 1, raw = TRUE) 5.291e-03 1.333e-04 39.695 < 2e-16  
## monthfAug 9.312e-02 3.141e-02 2.965 0.00306  
## monthfDec -3.975e-02 3.141e-02 -1.266 0.20582  
## monthfFeb -1.594e-02 3.136e-02 -0.508 0.61133  
## monthfJan 1.790e-02 3.136e-02 0.571 0.56826  
## monthfJul 8.962e-02 3.136e-02 2.858 0.00431  
## monthfJun 6.784e-02 3.136e-02 2.163 0.03063  
## monthfMar -8.994e-02 3.136e-02 -2.868 0.00418  
## monthfMay 1.193e-02 3.136e-02 0.380 0.70366  
## monthfNov -4.457e-02 3.141e-02 -1.419 0.15601  
## monthfOct 4.699e-02 3.141e-02 1.496 0.13477  
## monthfSep 6.051e-02 3.141e-02 1.927 0.05415  
##   
## Residual standard error: 0.2866 on 1986 degrees of freedom  
## Multiple R-squared: 0.4537, Adjusted R-squared: 0.4504   
## F-statistic: 137.5 on 12 and 1986 DF, p-value: < 2.2e-16

lm.d <- NULL  
  
for(i in 1:25){  
lm.d[i] <- lm(V2~poly(time.date,i,raw=TRUE) + monthf, data = hadcrut)  
}

## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length  
  
## Warning in lm.d[i] <- lm(V2 ~ poly(time.date, i, raw = TRUE) + monthf, data  
## = hadcrut): number of items to replace is not a multiple of replacement  
## length

lm.d[4]

## [[1]]  
## (Intercept) poly(time.date, i, raw = TRUE)1   
## 2.227918e+05 -4.628043e+02   
## poly(time.date, i, raw = TRUE)2 poly(time.date, i, raw = TRUE)3   
## 3.604719e-01 -1.247703e-04   
## poly(time.date, i, raw = TRUE)4 monthfAug   
## 1.619328e-08 9.608623e-02   
## monthfDec monthfFeb   
## -3.719075e-02 -1.573210e-02   
## monthfJan monthfJul   
## 1.820326e-02 8.928785e-02   
## monthfJun monthfMar   
## 6.762569e-02 -8.983179e-02   
## monthfMay monthfNov   
## 1.182315e-02 -4.190825e-02   
## monthfOct monthfSep   
## 4.975718e-02 6.338385e-02

lm.d[5]

## [[1]]  
## (Intercept) poly(time.date, i, raw = TRUE)1   
## 2.227918e+05 -4.628043e+02   
## poly(time.date, i, raw = TRUE)2 poly(time.date, i, raw = TRUE)3   
## 3.604719e-01 -1.247703e-04   
## poly(time.date, i, raw = TRUE)4 poly(time.date, i, raw = TRUE)5   
## 1.619328e-08 NA   
## monthfAug monthfDec   
## 9.608623e-02 -3.719075e-02   
## monthfFeb monthfJan   
## -1.573210e-02 1.820326e-02   
## monthfJul monthfJun   
## 8.928785e-02 6.762569e-02   
## monthfMar monthfMay   
## -8.983179e-02 1.182315e-02   
## monthfNov monthfOct   
## -4.190825e-02 4.975718e-02   
## monthfSep   
## 6.338385e-02

1. If we center or, even better, make the polynomial functions orthogonal to one another, we can avoid the issue in the previous question. Use poly(x1,?,raw=F) and step up the polynomial order for time until the p-value for the last coefficient (use summary()) is "large", reporting the single test result for each step in the building process. Then drop back one order and re-fit the model. Report the effects plot of the resulting model and describe the estimated trend. Note: aside from access to orthogonal polynomials the poly function interfaces with Anova and the effects package really nicely.
2. Check the diagnostic plots from your final model. Does anything improve from the first version. Also plot the residuals vs time and compare that plot to residuals vs fitted.
3. Run the following code so I can see what version of R you are now using:

### Documenting R version

getRversion()

## [1] '3.3.1'