Final Project

ALY 6020 20942 Predictive Analytics SEC 02

Winter 2018 CPS Quarter Graduate

Instructor——Marco Montes de Oca

LIANG SHI

28/03/2018

Summary

Objective

1. Research the relationship between copy number, RNA expression and protein expression based on specific cell line and construct a generalized linear regression model.

2. Choose 500 records at random and build a tree to classify cell lines according to protein expression. Use it to predict the protein expression of the remain records. Calculate the prediction error.

3. Use a random forest to fit the "MYC protein expression". Determine the fit as a function of number of MYC copy number and MYC RNA expression.

4. Use a gradient boosting to fit the "MYC protein expression". Determine the fit as a function of number of MYC copy number and MYC RNA expression.

Main Results

1. Defining protein expression as the dependent variable, copy number and RNA expression as independent variables, there is a linear relationship

Protein = 0.1533 * RNA_expression +0.0374 * copy_number -0.9171

This model passed t test and p value is small enough, while R square is only 31.82% which means the change of protein is not well explained by the change of RNA expression and copy number and in fact, a linear model won't be a great fit.

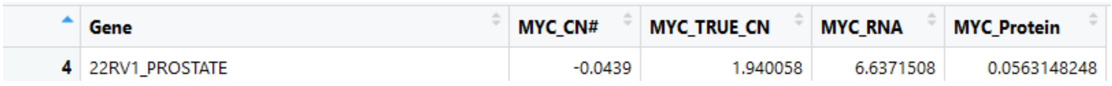
2. The classification tree has 10 terminal nodes. When the protein expression is -0.0129, 0.961, 0.347 and 0.926 the patient is relatively safe, while if the level is 0.876 and 1.3, the possibility of having tumor is very high.

3. The mean absolute error of fitted random forest function is 0.197, and the one of gradient boosting is 0.100, so gradient boosting works better than random forecast in predicting the protein expression.

Introduction

I will use the dataset called “03-08-18-Data Summary for MYC Gene”. It is consisted of 3 tables and each table has 1458 observations, the three tables will be merged by R studio.

Cancer is always a topic that people are afraid to talk about because we human seem not to have perfect methods to deal with it. In this case I am going to talk about a specific gene called myc. Theoretically, for each cell line of specific cancer or tumor, MYC will have specific copy numbers, RNA expression and protein expression, for example



	Gene	MYC_CN#	MYC_TRUE_CN	MYC_RNA	MYC_Protein
4	22RV1_PROSTATE	-0.0439	1.940058	6.6371508	0.0563148248

22RV1 is the name of a cell line and it comes from a patient who has prostate cancer. For this cell line, the MYC gene has 1.94 copy number, 6.64 RNA expression and 0.056 protein expression.

Another reason why I choose this topic is because Boston is one of the largest pharmaceutical factories, a huge amount of clinical data can be generalized every day, and the clinical data is strongly supported by epidemiology. So when epidemiology is combined with analytics, the relationship between variables can be tested and specific indicators can be predicted, which can make this project interesting and be of great practical importance.

Questions addressed in this project will include researching the relationship between copy number, RNA expression and protein expression, predicting protein number based on classification trees and using random forest and gradient to fit the function of RNA expression and copy number.

Implementation

1. Acquire and Clean Data

The initial dataset is consisted of three separate tables, so I have to merge them together as a new data frame

```
d1 <-merge(copy_number,RNA_expression, by="Gene")
mydata <- merge(d1, protein_expression, by="Gene")
```

Not each cell line has its corresponding copy number, RNA and protein at the same time, so I have to eliminate the rows with null values.

```
clean_data <-mydata[complete.cases(mydata),]
```

My new dataset has 879 observations in total.

2. Generalized Linear Regression

Normally, when the MYC copy number of a patient is around 2, he is relatively safe, while when the copy number is over 2, the probability of having a tumor may over 70% or the existing tumor is hard to treat. Then if the copy number is higher, the probability can be much higher. However, comparing to copy number and RNA expression, protein expression level is more difficult to observe, so I assume there is a generalized linear relationship between these three variables, defining protein as the dependent variable, copy number and RNA as independent variables. Then I can get the corresponding level of protein regarding to specific copy number.

```
#linear regression and generalized regression
#fit <-lm(MYC_Protein~MYC_RNA +MYC_TRUE_CN, data = clean_data)
mod <-glm(MYC_Protein~MYC_RNA +MYC_TRUE_CN, data = clean_data, family = gaussian(link = "identity"))
```

Here I choose the cleaned data which has no null value. In this case, the error distribution obeys the gaussian distribution, and that’s why I choose family function as “gaussian” and the link as “identity”.

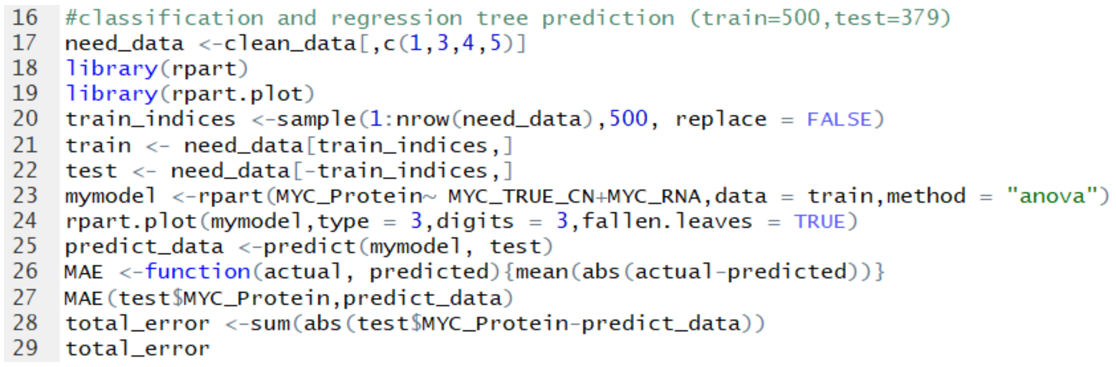
Then, the generalized linear model is contradicted to standard linear regression analysis, no assumption is made that the relationship is represented by a straight line(GEOG 414/514), so I have to test it.

```
# calculate and store predicted values
cn_data <-mutate(clean_data,predicted=predict(mod,type = "response"))

#order by copy numer and then RNA then protein
cn_data <-arrange(cn_data,MYC_RNA, MYC_TRUE_CN)
scatter.smooth(x=cn_data$MYC_Protein, y=cn_data$predicted)
```

3. Classification and Regression Tree

By classifying the cell lines by protein expression, we can see which level is relatively safe according our existing knowledge about the safe range of corresponding copy number. That’s the reason why I build the tree and make predictions.

Besides, I choose 500 records as train set and construct the model. Then I will use the model to predict the protein expression of rest 379 records, using mean absolute error to measure the quality of my model and prediction result.

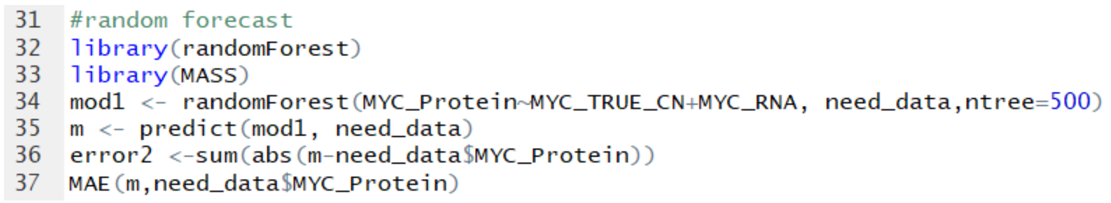


```r
#classification and regression tree prediction (train=500,test=379)
need_data <-clean_data[,c(1,3,4,5)]
library(rpart)
library(rpart.plot)
train_indices <-sample(1:nrow(need_data),500, replace = FALSE)
train <- need_data[train_indices,]
test <- need_data[-train_indices,]
mymodel <-rpart(MYC_Protein~ MYC_TRUE_CN+MYC_RNA,data = train,method = "anova")
rpart.plot(mymodel,type = 3,digits = 3,fallen.leaves = TRUE)
predict_data <-predict(mymodel, test)
MAE <-function(actual, predicted){mean(abs(actual-predicted))}
MAE(test$MYC_Protein,predict_data)
total_error <-sum(abs(test$MYC_Protein-predict_data))
total_error
```

I extracted out the 1st, 3rd ,4th and 5th column of the original dataset because the 2nd one just helps us to get the true copy number. Here I choose the method as “anova” because the dependent variable chosen here is not a factor but a number, and it only has one column, so I will not choose “class” or “poisson”. Besides, I calculate both total error and mean absolute error here to check the prediction accuracy.

4. Random Forest

Choosing random forest to fit the function of MYC copy number and RNA expression is because I want to make the prediction more accurate and persuasive by constructing more trees and see the result.

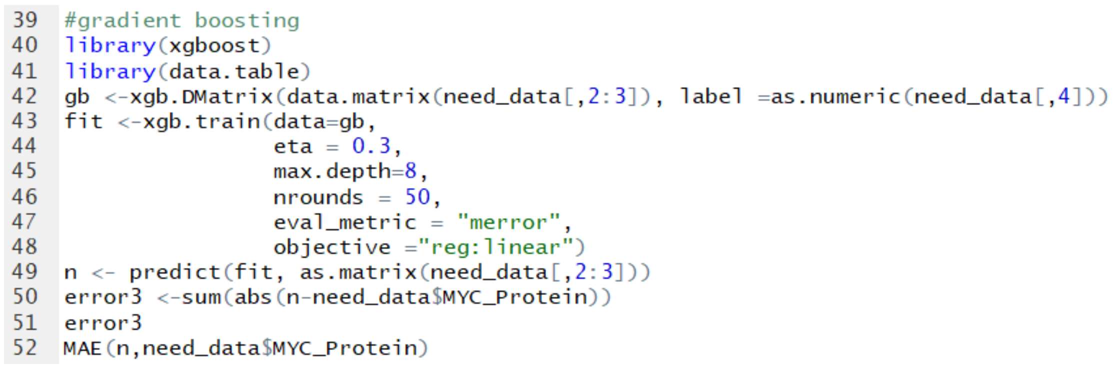


```r
#random forecast
library(randomForest)
library(MASS)
mod1 <- randomForest(MYC_Protein~MYC_TRUE_CN+MYC_RNA, need_data,ntree=500)
m <- predict(mod1, need_data)
error2 <-sum(abs(m-need_data$MYC_Protein))
MAE(m,need_data$MYC_Protein)
```

Here I define the amount of tree inside the forest is 500 and then I make a prediction basing on the new data. I also calculate the total error and mean absolute error here, it can help to check the accuracy as well as compare the random forest and simple classification tree.

5. Gradient Boosting

Gradient boosting is another technique for regression and classification problems, it can also be used to predict models. The difference is that it can generalize the model by allowing optimization of an arbitrary differentiable loss function, which means minimizing the loss. The reason why I choose this model is to predict the protein expression again and see which model is the best one regarding to classification and prediction issues.



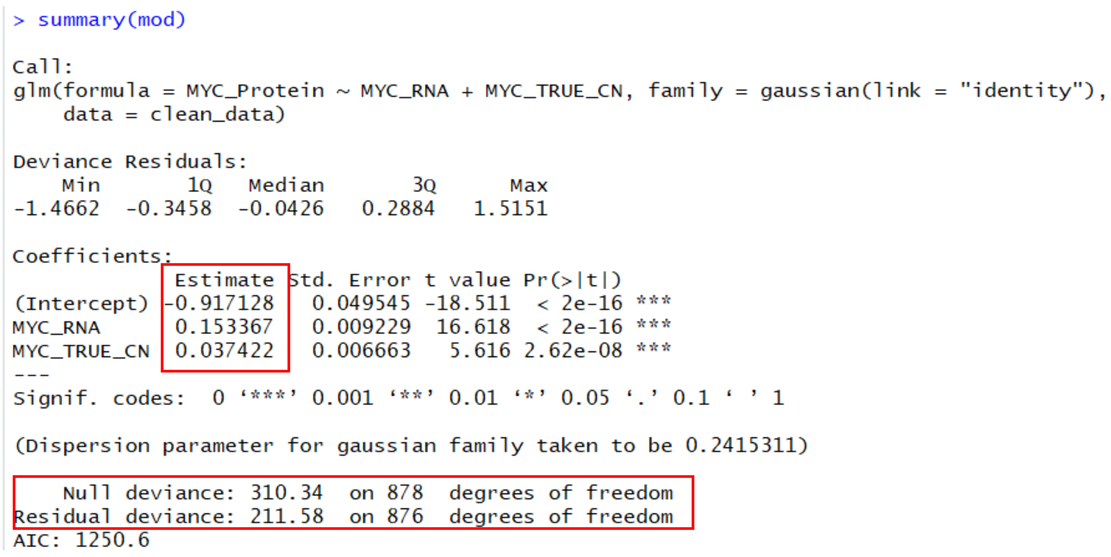
```
#gradient boosting
library(xgboost)
library(data.table)
gb <-xgb.DMatrix(data.matrix(need_data[,2:3]), label =as.numeric(need_data[,4]))
fit <-xgb.train(data=gb,
                eta = 0.3,
                max.depth=8,
                nrounds = 50,
                eval_metric = "merror",
                objective ="reg:linear")
n <- predict(fit, as.matrix(need_data[,2:3]))
error3 <-sum(abs(n-need_data$MYC_Protein))
error3
MAE(n,need_data$MYC_Protein)
```

Here I choose eta as 0.3 which means how far a vector needs to be negatively moved, nrounds as 50, representing the number of trees generated, and objective as "reg: linear" because my case is a linear regression. Total error and mean absolute error is used to make comparison after wards.

Data Analysis

1. Generalized Linear Regression

I use glm() function to describe a linear relationship between these three variables, the summary of my model can be like this

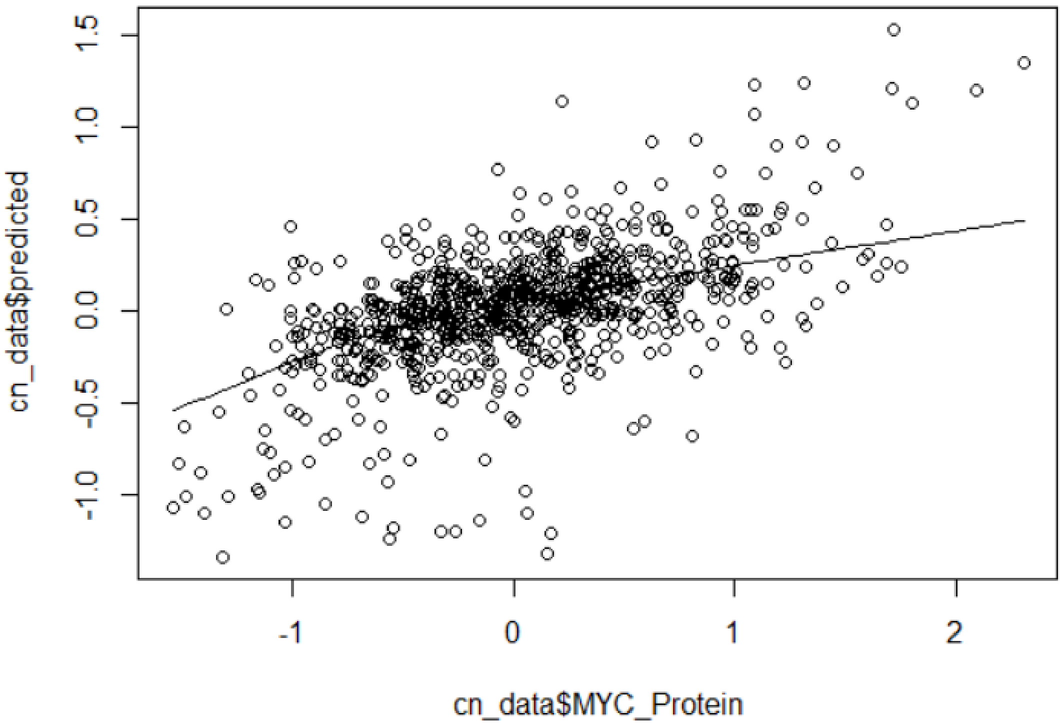


```
> summary(mod)

Call:
glm(formula = MYC_Protein ~ MYC_RNA + MYC_TRUE_CN, family = gaussian(link = "identity"),
    data = clean_data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.4662  -0.3458  -0.0426   0.2884   1.5151

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.917128   0.049545 -18.511  < 2e-16 ***
MYC_RNA      0.153367   0.009229  16.618  < 2e-16 ***
MYC_TRUE_CN  0.037422   0.006663   5.616 2.62e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 0.2415311)

    Null deviance: 310.34  on 878  degrees of freedom
Residual deviance: 211.58  on 876  degrees of freedom
AIC: 1250.6
```

The linear model can be like this

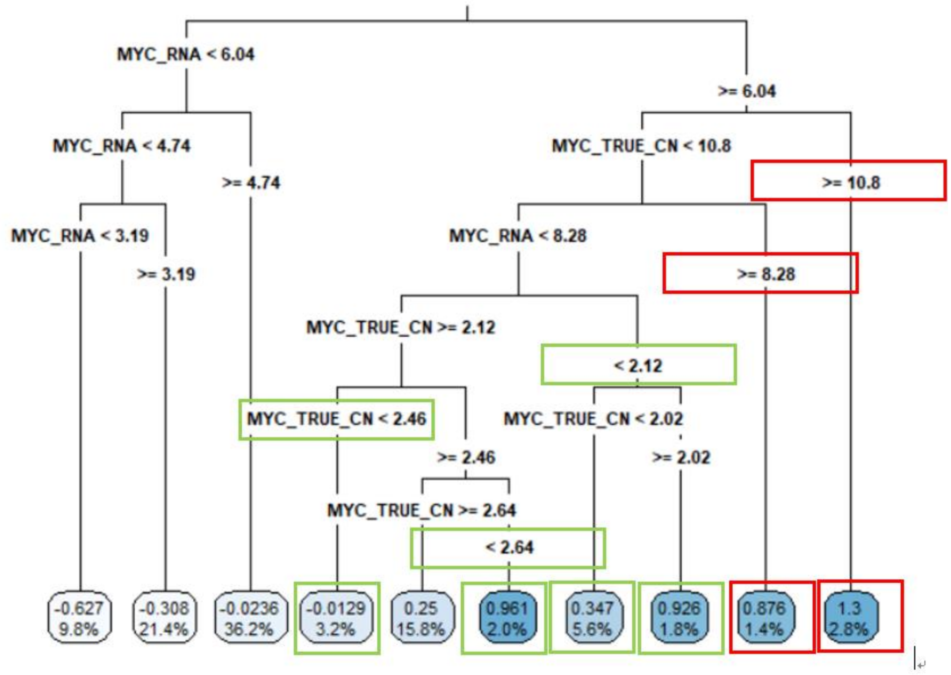
Protein = 0.1533 * RNA_expression +0.0374 * copy_number -0.9171

Deviance is similar to the variance and it describes the degree of bias of this model. Obviously, residual deviance is 211.58 which is a high value. Just as mentioned before, scatter smoothing is still needed to check whether this line fits well.



The fit result is flexible than a straight line and it bends downward from right to left. So, in this case, a linear model won't be a great fit especially for the values on the lower left part.

2. Classification and Regression Tree



My classification tree has 10 terminal nodes in total. Out of all predictors the RNA expression is the most predictive one. Model calculated that protein expression will be split by RNA less then 6.04 and not.

Just as I mentioned before, copy number around 2 is a relatively safe level, in this case, the protein level inside the green box is the safe level which means a low probability of having tumor or the tumor is still under control, while the values inside the red box have exceed the safe region.

```
> head(predict_data)
          4           5          11          12 
 0.34694089 -0.02362645 -0.02362645 -0.02362645 
         13          15 
 0.96094581 -0.02362645 
```

Above is the head result of our prediction

```
> MAE(test$MYC_Protein,predict_data)
[1] 0.4067715
```

Mean absolute error is only 0.407 which is not a high value. So our model is relatively accurate.

3. Random Forecast

Random forecast can be taken as the cluster of classification tree. The head result of prediction is

```
> head(m)
            4             5             8 
 0.2396226506 -0.3787588350 -0.3537263211 
           10            11            12 
 0.4342285234 -0.1864499884 -0.0008457467 
```

Then the mean absolute error is

```
> MAE(m,need_data$MYC_Protein)
[1] 0.1967665
```

The mean absolute error is only 0.197 which is much lower than simple classification tree does.

4. Gradient Boosting

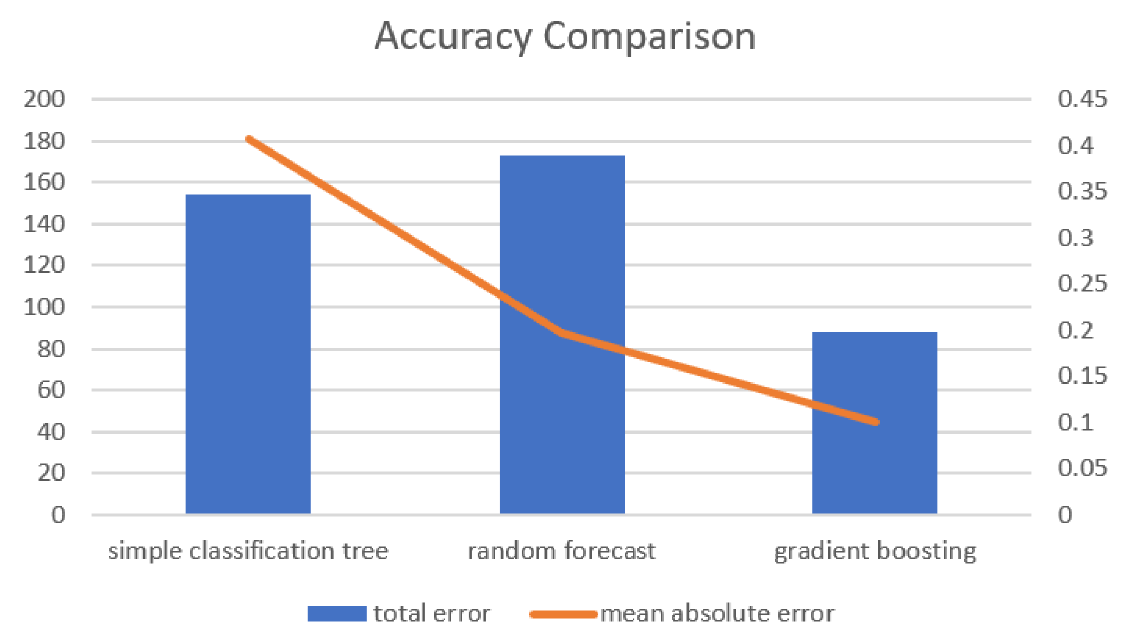
Gradient boosting is another classification and predicting method which focus on loss function. Here is the head result of prediction

```
> head(n)
[1]  0.2125828 -0.5050354 -0.3663711  0.6065171
[5] -0.2472127  0.1430299
```

Then the mean absolute error is

```
> MAE(n,need_data$MYC_Protein)
[1] 0.1004393
```

The mean absolutely error is only 0.100 which further lower than random forecast does.



Contemplating all the three classification and predicting methods, gradient boosting has the least mean absolute error which is 0.100, so in this case, gradient boosting model is the most accurate one.

Discussion

From the perspective of analytics, the project is interesting because it challenges the traditional point that the relationship between copy number, RNA expression and protein expression is linear. Additionally, it offers an analytical way to predict the protein number basing on copy number and RNA, which are easy to observe, then the conclusion can be taken to consider the probability of having tumor. Last but not least, it selected out the best model when it comes to this kind of issue.

Techniques used in this project such as generalized linear model, classification trees, random forecast and gradient boosting are highly overlapped with what were covered in class. This project put all the techniques we got into practical use.

As for future analysis, I think the exact relationship between the three variables should be figured out which can improve the accuracy. What's more, we can still further predict what the specific organ will the cell line come from and which part has the highest possibility to be infected with cancer.

Reference

1. J.Blancato, B.Singh, A.Liu, D.J.Liao and R.B.Dickson.Correlation of amplification and overexpression of the c-myc oncogene in high-grade breast cancer: FISH, in situ hybridisation and immunohistochemical analyses. (2004, March 30). Retrieved March 28, 2018, from https://www.nature.com/articles/6601703

2. GEOG 414/514: Advanced Geographic Data Analysis Scatter-diagram smoothing (Nonparametric regression). (n.d.). Retrieved March 28, 2018, from http://geog.uoregon.edu/bartlein/old_courses/geog414f03/lectures/lec05.htm