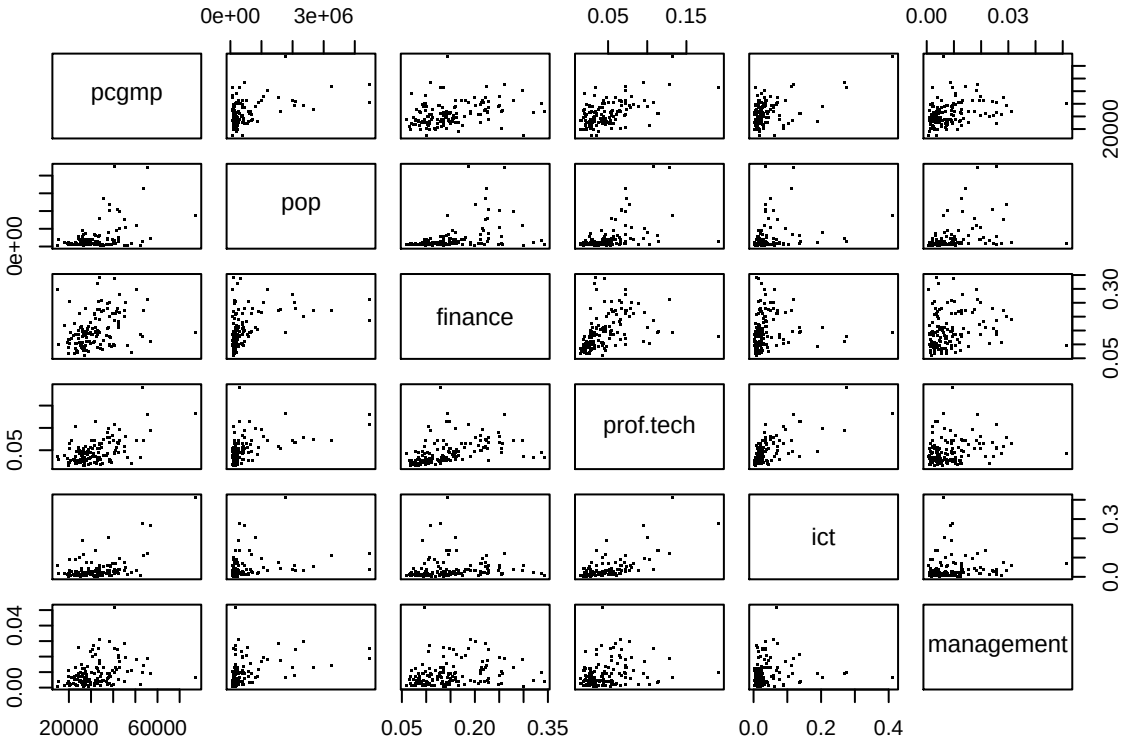
HW10

Shaojie Zhang (shaojiez)

04/18/2019

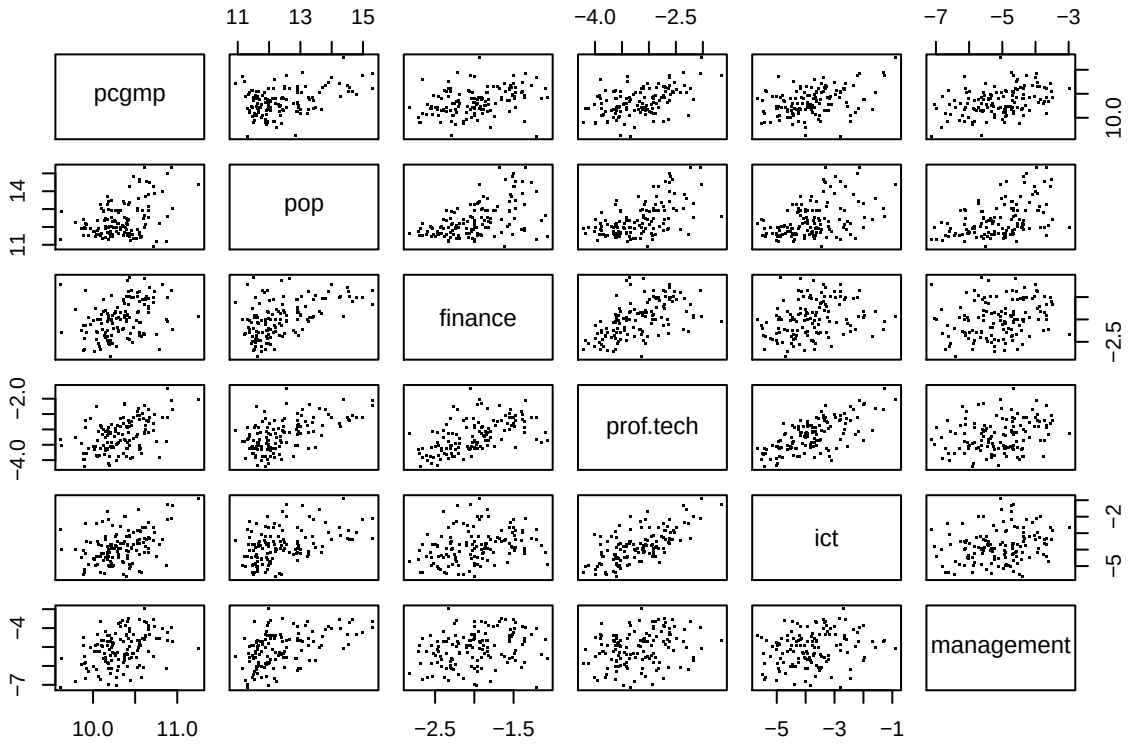
Part a

```
gmp = read.csv("gmp.csv",header=T)

# Exploratory Analysis
pairs(gmp[, 3:8], pch=".")
```



We notice from our basic exploratory analysis, we see that lots of the graphs in this pair plots have outliers, so we need to do some transformations to the data.

```
pairs(log(gmp[, 3:8]), pch=".")
```



We apply log transformation and now the relationships are more clear.

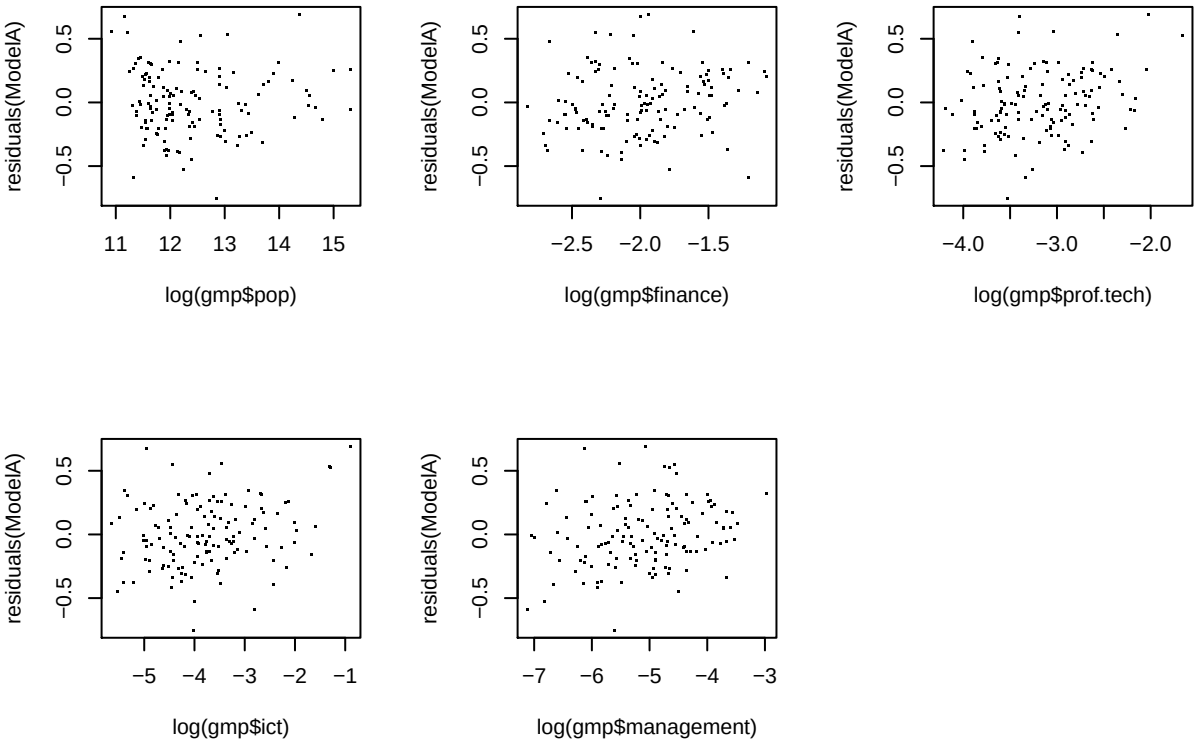
```
library(mgcv)
```

```
## Loading required package: nlme
```

```
## This is mgcv 1.8-17. For overview type 'help("mgcv-package")'.
```

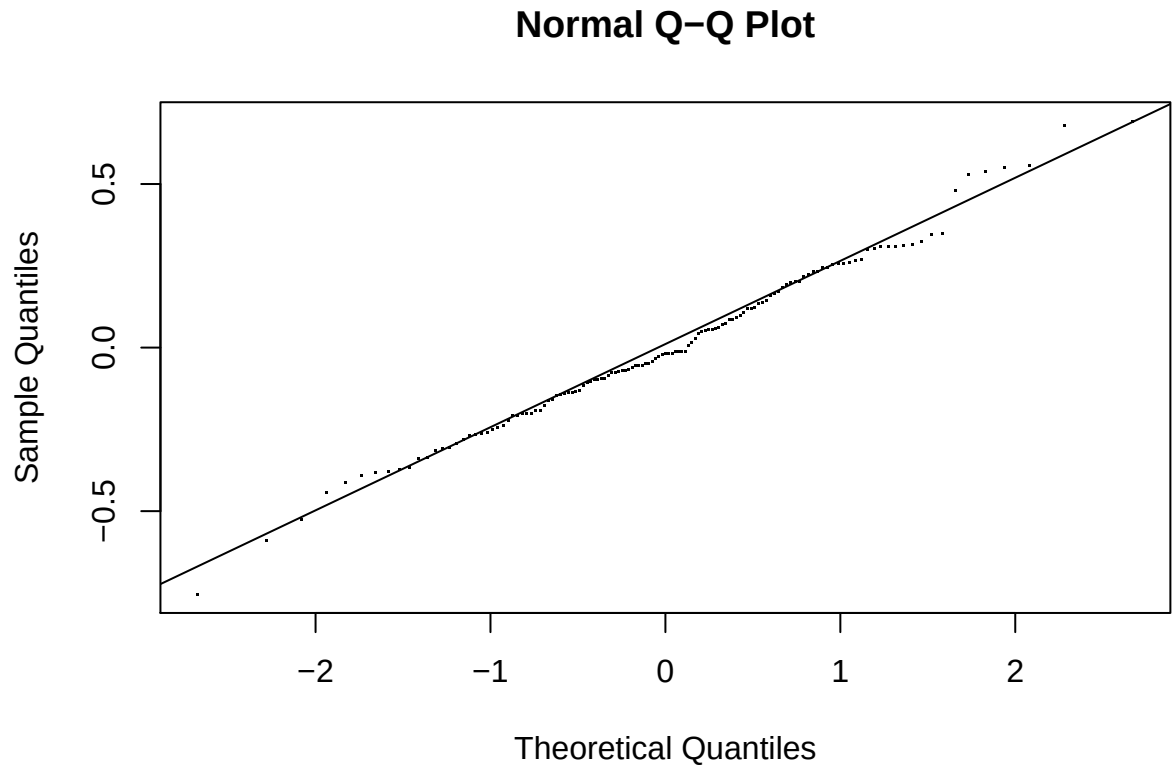
```
ModelA = gam(log(pcgmp) ~ log(pop), data=gmp)

# Residual plots
par(mfrow=c(2,3))
plot(log(gmp$pop), residuals(ModelA),pch=".")
title(main = "Residuals Plots ModelA")
plot(log(gmp$finance), residuals(ModelA),pch=".")
plot(log(gmp$prof.tech), residuals(ModelA),pch=".")
plot(log(gmp$ict), residuals(ModelA),pch=".")
plot(log(gmp$management),residuals(ModelA),pch=".")
```

Residuals Plots ModelA



We observe some similar patterns across all the residual plots. Now we want to look at qq plot.

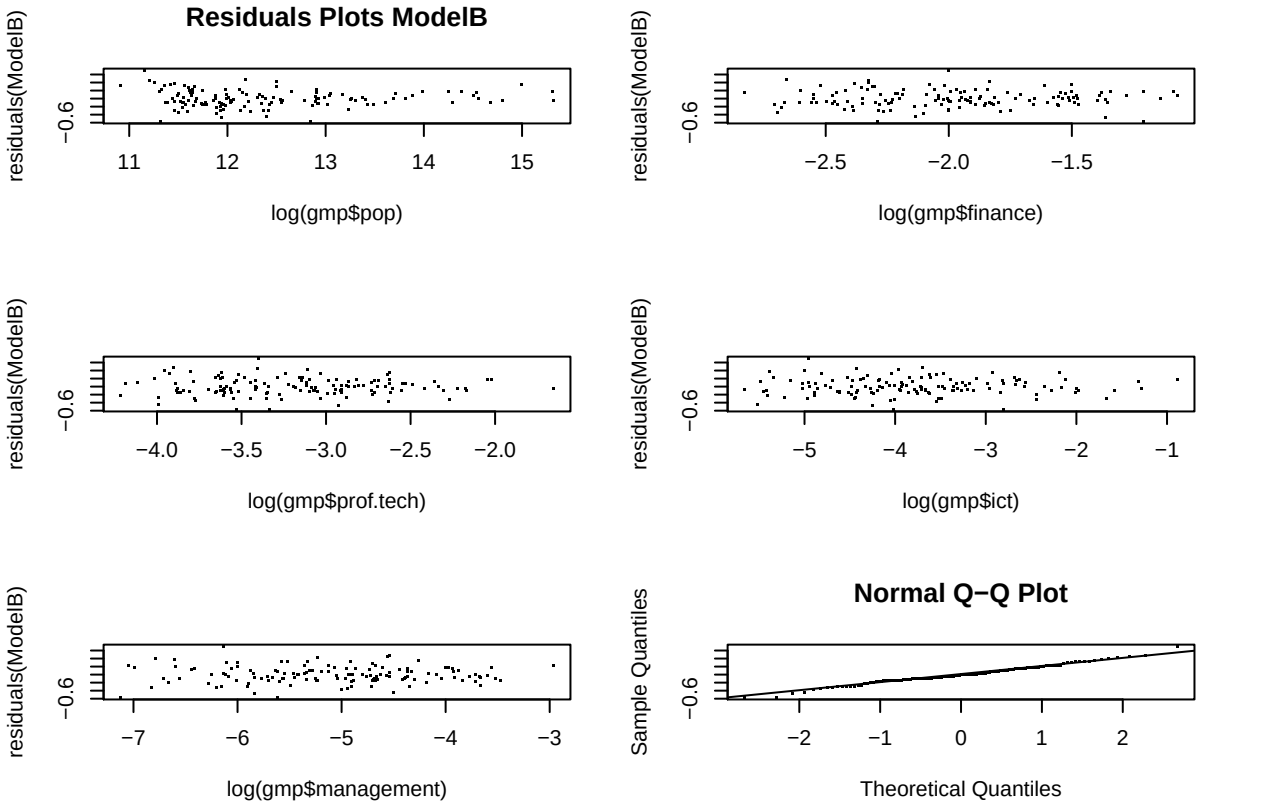
```
qqnorm(residuals(ModelA), pch=".")
qqline(residuals(ModelA))
```



We see the line is very straight, so we conclude that there is no assumption violation.

Part b

```
ModelB = gam(log(pcgmp) ~ log(pop)+ s(log(finance),k=5,fx=T) + s(log(prof.tech),k=5,fx=T) + s(log(ict),k

# Residual Plots
par(mfrow=c(3, 2))
plot(log(gmp$pop), residuals(ModelB),pch=".")
title(main="Residuals Plots ModelB")
plot(log(gmp$finance), residuals(ModelB),pch=".")
plot(log(gmp$prof.tech), residuals(ModelB),pch=".")
plot(log(gmp$ict), residuals(ModelB),pch=".")
plot(log(gmp$management), residuals(ModelB),pch=".")

# QQ plots
qqnorm(residuals(ModelB), pch=".")
qqline(residuals(ModelB))
```



From our residual plots and QQ plot, we can't really tell the relationship between the parameters. But the QQ plot looks good so no assumption is violated.

```
summary(ModelB)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## log(pcgmp) ~ log(pop) + s(log(finance), k = 5, fx = T) + s(log(prof.tech),
##     k = 5, fx = T) + s(log(ict), k = 5, fx = T) + s(log(management),
##     k = 5, fx = T)
```

```
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 10.75356    0.39448  27.260   <2e-16 ***
## log(pop)    -0.03383    0.03177  -1.065    0.289
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                     edf Ref.df     F p-value
## s(log(finance))      4      4 2.356 0.05780 .
## s(log(prof.tech))    4      4 1.255 0.29175
## s(log(ict))          4      4 3.030 0.02032 *
## s(log(management))   4      4 4.395 0.00242 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.346   Deviance explained =   43%
## GCV = 0.058945  Scale est. = 0.050968  n = 133
```

From the model summary, we see that the log(pop), log(finance) and log(prof.tech) does not contribute in the model with their relatively large p value.

Part c

```
anova(ModelA, ModelB, test="F")
```

```
## Analysis of Deviance Table
##
## Model 1: log(pcgmp) ~ log(pop)
## Model 2: log(pcgmp) ~ log(pop) + s(log(finance), k = 5, fx = T) + s(log(prof.tech),
##     k = 5, fx = T) + s(log(ict), k = 5, fx = T) + s(log(management),
##     k = 5, fx = T)
##   Resid. Df Resid. Dev Df Deviance      F    Pr(>F)
## 1       131     8.6450
## 2       115     5.8613 16   2.7837 3.4136 6.137e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From ANOVA test, we see that p value is really small, which means that ModelB is better than ModelA. From our previous plots and observations, log(pop), log(finance) and log(prof.tech) are probably not needed.

Part d

```
# parametric bootstrap
B=1000
n=133
rssA=sum(residuals(ModelA)^2)
rssB=sum(residuals(ModelB)^2)
dfA=sum(ModelA$edf)
dfB=sum(ModelB$edf)
f=((rssA-rssB)/(dfB-dfA))/(rssB/(n-dfB))

fs=NULL
datarep=gmp
```

```
sig=sd(residuals(ModelA))
for(b in 1:B){
  datarep$pcgmp = exp(rnorm(n)*sig + fitted.values(ModelA))
  m1 = gam(log(pcgmp)~log(pop), data=datarep)
  m2 = gam(log(pcgmp)~log(pop)+s(log(finance),k=5,fx=T) + s(log(prof.tech),k=5,fx=T) + s(log(ict),k=5,f
  rssA1=sum(residuals(m1)^2)
  rssB1=sum(residuals(m2)^2)
  fs[b]=((rssA1-rssB1)/(dfB-dfA))/(rssB1/(n-dfB))
}
```

What we are supposed to look at after the boostrap is the comparison with mean and variance. We see that dfA and dfB are 2 and 18 in our case. And mean is 1.125 and variance is 1.63 under f distribution. So now look at:

```
mean(fs)
```

```
## [1] 1.016208
```

```
var(fs)
```

```
## [1] 0.1473636
```

So this bootstrap is a little off.

```
# resample residuals bootstrap
fss=NULL
datareps=gmp
sig=sd(residuals(ModelA))
for(b in 1:B){
  samp=sample(n,replace=T)
  datareps$pcgmp=exp(residuals(ModelA)[samp] + fitted.values(ModelA))
  m1=gam(log(pcgmp)~log(pop), data=datareps)
  m2=gam(log(pcgmp)~log(pop) + s(log(finance),k=5,fx=T) + s(log(prof.tech),k=5,fx=T) + s(log(ict),k=5,f
  rssA1=sum(residuals(m1)^2)
  rssB1=sum(residuals(m2)^2)
  fss[b]=((rssA1-rssB1)/(dfB-dfA))/(rssB1/(n-dfB))
}
```

Again, compare with the numbers:

```
mean(fss)
```

```
## [1] 1.031813
```

```
var(fss)
```

```
## [1] 0.1506601
```

Again, the model is off. So it is some bad estimation for c

Part e

The reason is that resampling cases can't give the same sample as the NULL. The distribution of the sampled data could be messed up.

Part g

```
ModelC = gam(log(pcgmp) ~ s(log(ict),k=5,fx=T) + s(log(management),k=5,fx=T), data=gmp)
```

```r
testdata=gmp[c(10,34,70),]
cpred=predict(ModelC, testdata, se.fit=T)

cbind(cpred$fit + qt(0.05,9)*cpred$se.fit, cpred$fit + qt(0.95,9)*cpred$se.fit)
```

```
##        [,1]     [,2]
## 10 10.50025 10.75058
## 34 10.51224 10.76679
## 70 10.11076 10.33394
```

From the comparison, we see the predict intervals. Now it's time to do bootstrap:

```r
B=1000
means=NULL

for(b in 1:B){
  samp=sample(n,replace=T)
  sampdata=gmp[samp,]
  mod=gam(log(pcgmp) ~ s(log(ict),k = 5, fx = T) + s(log(management), k = 5, fx = T), data = sampdata)
  means=rbind(means,predict(mod, testdata))
}

quantile=apply(means, 2, quantile, prob=c(0.05,0.95))

cbind(2*cpred$fit-quantile[2,], 2*cpred$fit-quantile[1,])
```

```
##        [,1]     [,2]
## 10 10.52405 10.79185
## 34 10.53066 10.76858
## 70 10.07364 10.32365
```

Here's our intervals. We observe that they are pretty close to the t distribution.