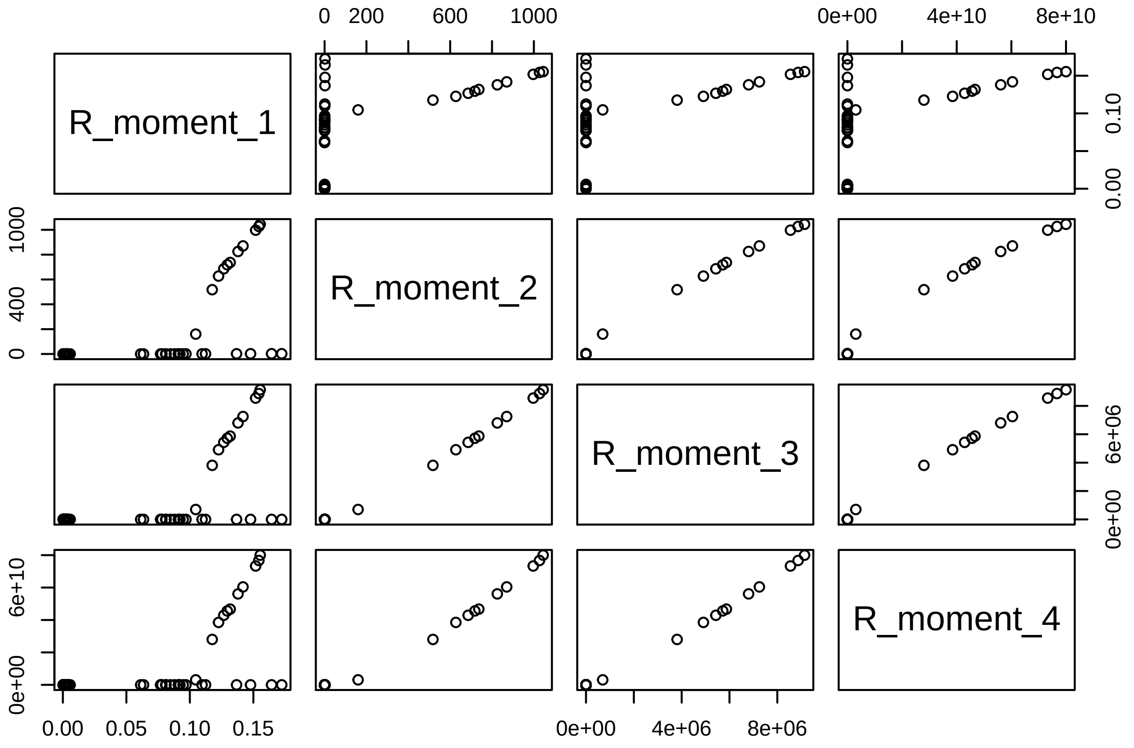
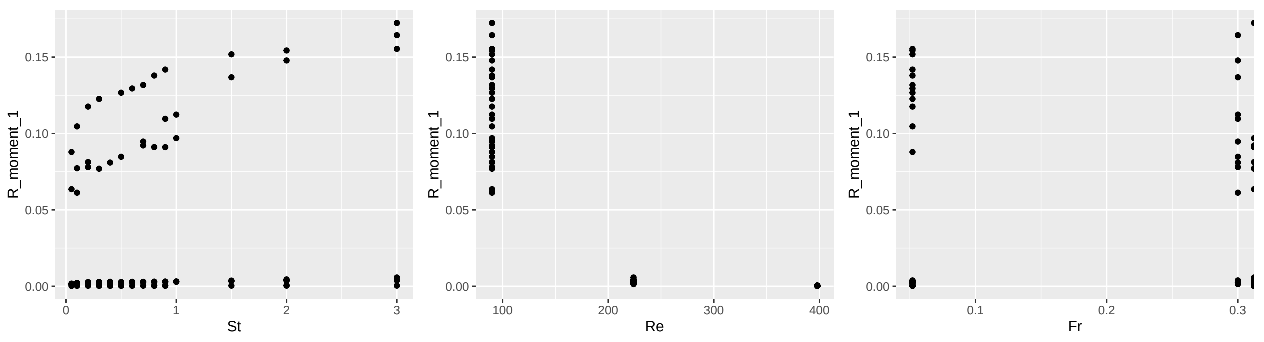
case_study_report

Connie Wu, Jason McEachin, Joe Choo, Scott Heng

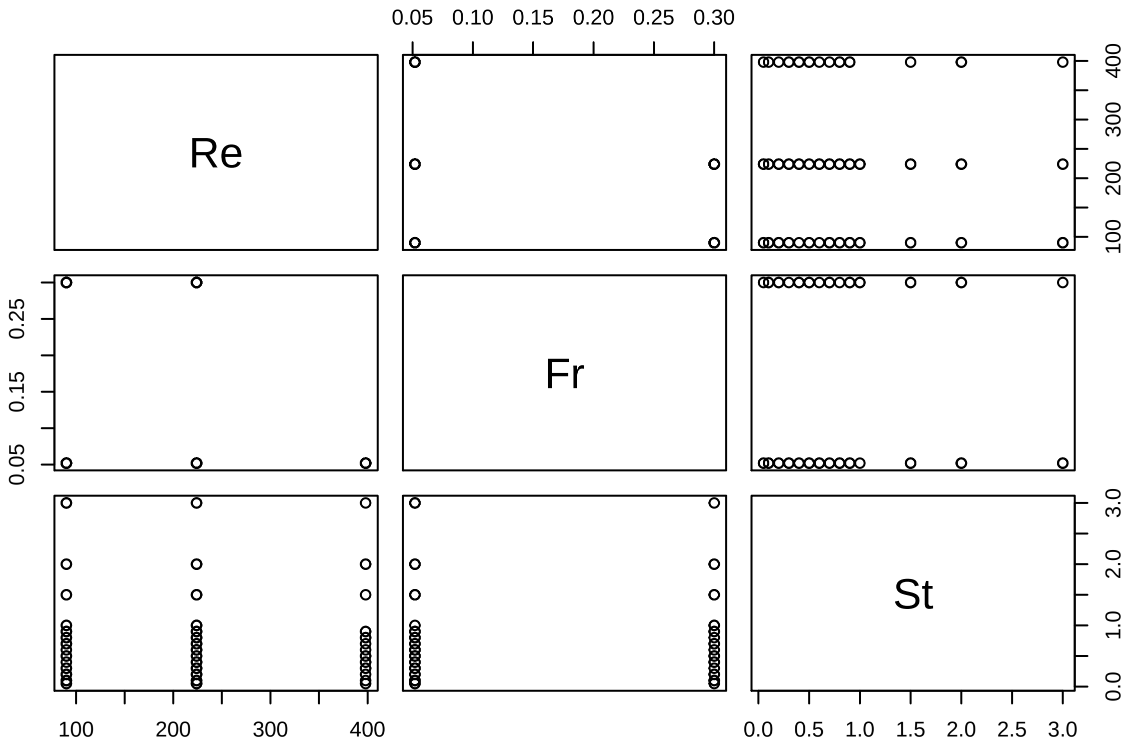
10/10/2020

```r
#pairs plot to show correlation between response variables
train_response <- data.frame(train[,c("R_moment_1","R_moment_2","R_moment_3","R_moment_4")])
pairs(train_response)
```





```r
#correlation between predictor variables (in case we need to include interaction effects)
train_predictors <- data.frame(train[,c("Re","Fr","St")])
pairs(train_predictors)
```



```
#linear regression with factored Re and Fr
train1 <- train
train1$Re <- as.factor(train$Re)
train1$Fr <- as.factor(train$Fr)
lm1_m1 <- lm(R_moment_1 ~ St + Fr + Re, data=train1)
lm2_m1 <- lm(R_moment_2 ~ St + Fr + Re, data=train1)
lm3_m1 <- lm(R_moment_3 ~ St + Fr + Re, data=train1)
lm4_m1 <- lm(R_moment_4 ~ St + Fr + Re, data=train1)
summary(lm1_m1)
```

```
##
## Call:
## lm(formula = R_moment_1 ~ St + Fr + Re, data = train1)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.038834 -0.008614  0.001702  0.009854  0.039423
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.106488   0.003971  26.818  < 2e-16 ***
## St           0.012213   0.002078   5.877 8.42e-08 ***
## Fr0.3       -0.007623   0.004245  -1.796  0.07618 .
## FrInf       -0.010210   0.003787  -2.696  0.00849 **
## Re224       -0.108091   0.003682 -29.353  < 2e-16 ***
## Re398       -0.111553   0.004632 -24.081  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01529 on 83 degrees of freedom
## Multiple R-squared:  0.9293, Adjusted R-squared:  0.9251
## F-statistic: 218.2 on 5 and 83 DF,  p-value: < 2.2e-16
```

```
x <- model.matrix(R_moment_1~St + Fr + Re,data=train_noinf)[,-1]
y <- train_noinf$R_moment_1

library(glmnet)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

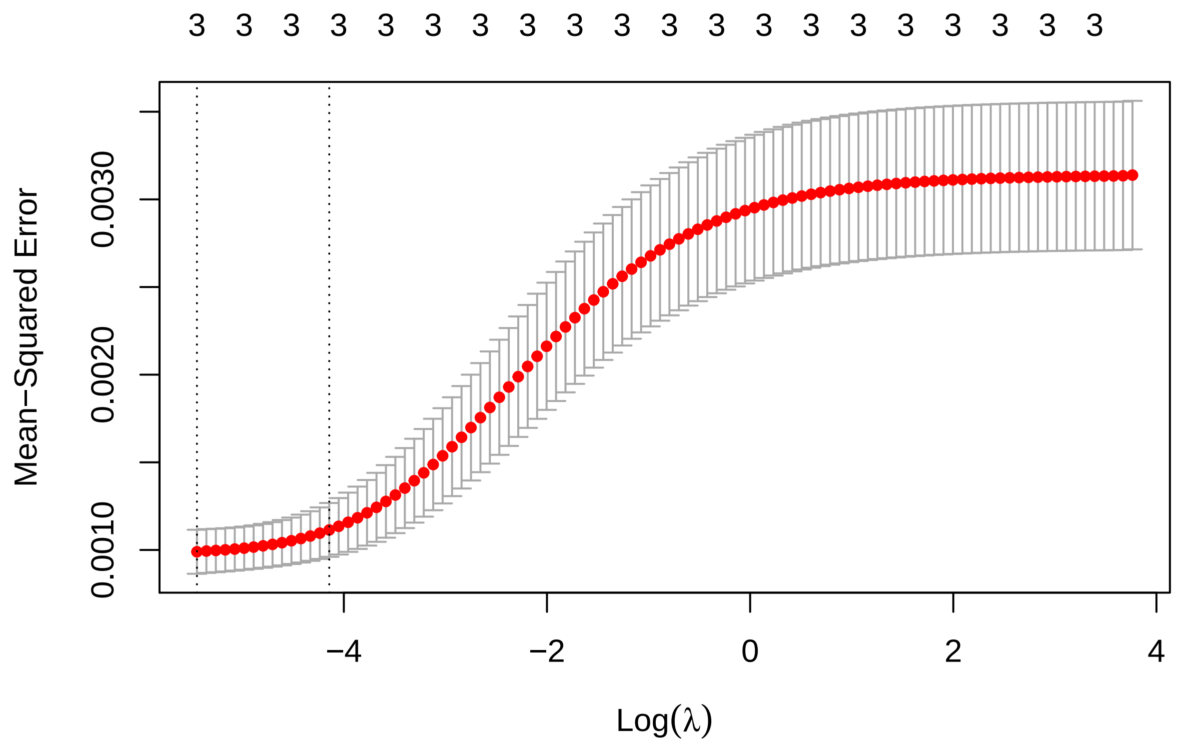
```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.0-2
```

```
grid <- 10^seq(10, -2, length = 100) # grid of values for lambda param
ridge.mod <- glmnet(x, y, alpha = 0, lambda = grid)


train <- sample(1:nrow(x), nrow(x)/2)
test <- (-train)
y.test <- y[test]

ridge.mod <- glmnet(x[train,], y[train], alpha = 0, lambda = grid, thresh = 1e-12)
ridge.pred <- predict(ridge.mod, s = 4, newx = x[test,])
mean((ridge.pred - y.test)^2) ## calculate MSE
```

```
## [1] 0.003083567
```

```
set.seed(1)
cv.out <- cv.glmnet(x[train,], y[train], alpha = 0)
plot(cv.out)
```



```
bestlam <- cv.out$lambda.min
bestlam
```

```
## [1] 0.004312608
```

```
ridge.pred <- predict(ridge.mod, s = bestlam, newx = x[test,])
mean((ridge.pred - y.test)^2)
```

```
## [1] 0.001489969
```

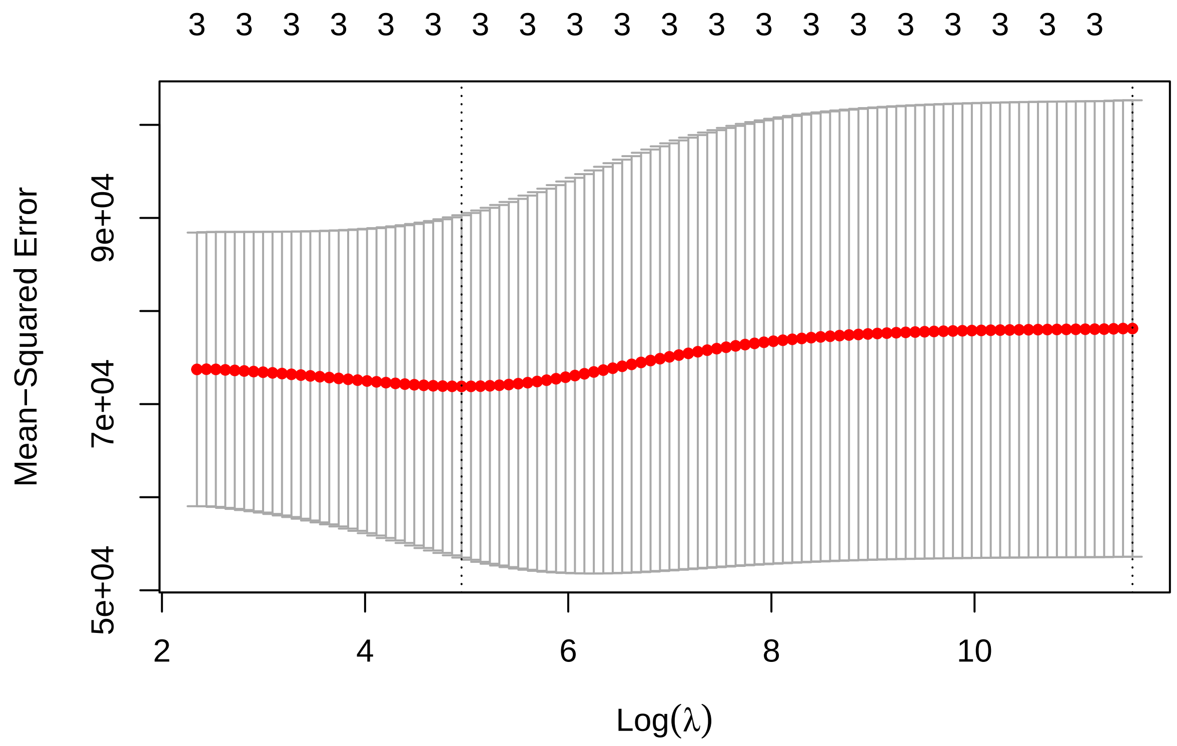
improvment in the mse

```
x2 <- model.matrix(R_moment_2~ St + Fr + Re,data=train_noinf)[,-1]
y2 <- train_noinf$R_moment_2

library(glmnet)
set.seed(1)
grid <- 10^seq(10, -2, length = 100) # grid of values for lambda param
ridge.mod2 <- glmnet(x2, y2, alpha = 0, lambda = grid)


train2 <- sample(1:nrow(x2), nrow(x2)/2)
test2 <- (-train2)
y.test2 <- y2[test2]

ridge.mod2 <- glmnet(x2[train,], y2[train], alpha = 0, lambda = grid, thresh = 1e-12)
ridge.pred <- predict(ridge.mod, s = 4, newx = x2[test,])
mean((ridge.pred - y.test2)^2) ## calculate MSE
```

```
## [1] 66957.52
```

```
set.seed(1)
cv.out2 <- cv.glmnet(x2[train2,], y2[train2], alpha = 0)
```

```r
plot(cv.out2)
```



```r
bestlam2 <- cv.out2$lambda.min
bestlam2
```

```
## [1] 141.1089
```

```r
ridge.pred2 <- predict(ridge.mod2, s = bestlam2, newx = x2[test2,])
mean((ridge.pred2 - y.test2)^2)
```

```
## [1] 46952.88
```

```r
x3 <- model.matrix(R_moment_3~St + Fr + Re,data=train_noinf)[,-1]
y3 <- train_noinf$R_moment_3


ridge.mod3 <- glmnet(x3, y3, alpha = 0, lambda = grid)


train3 <- sample(1:nrow(x3), nrow(x3)/2)
test3 <- (-train3)
y.test3 <- y[test3]

ridge.mod3 <- glmnet(x3[train3,], y3[train3], alpha = 0, lambda = grid, thresh = 1e-12)
ridge.pred3 <- predict(ridge.mod3, s = 4, newx = x3[test3,])
mean((ridge.pred3 - y.test3)^2) ## calculate MSE
```

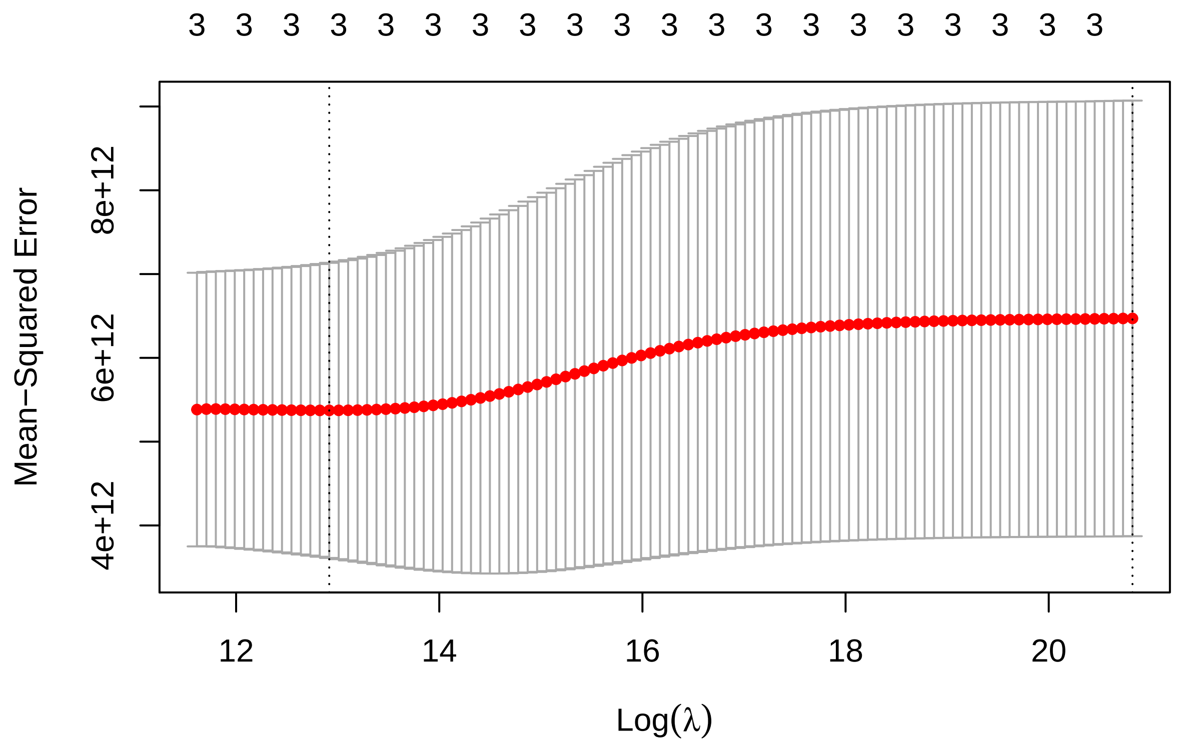
```
## [1] 2.924727e+12
```

```r
x4 <- model.matrix(R_moment_4~St + Fr + Re,data=train_noinf)[,-1]
y4 <- train_noinf$R_moment_4

ridge.mod4 <- glmnet(x4, y4, alpha = 0, lambda = grid)
```

```r
train4 <- sample(1:nrow(x4), nrow(x4)/2)
test4 <- (-train4)
y.test4 <- y[test4]

ridge.mod4 <- glmnet(x4[train4,], y4[train4], alpha = 0, lambda = grid, thresh = 1e-12)
ridge.pred4 <- predict(ridge.mod4, s = 4, newx = x4[test4,])
mean((ridge.pred4 - y.test4)^2) ## calculate MSE
```

```
## [1] 1.702767e+20
```

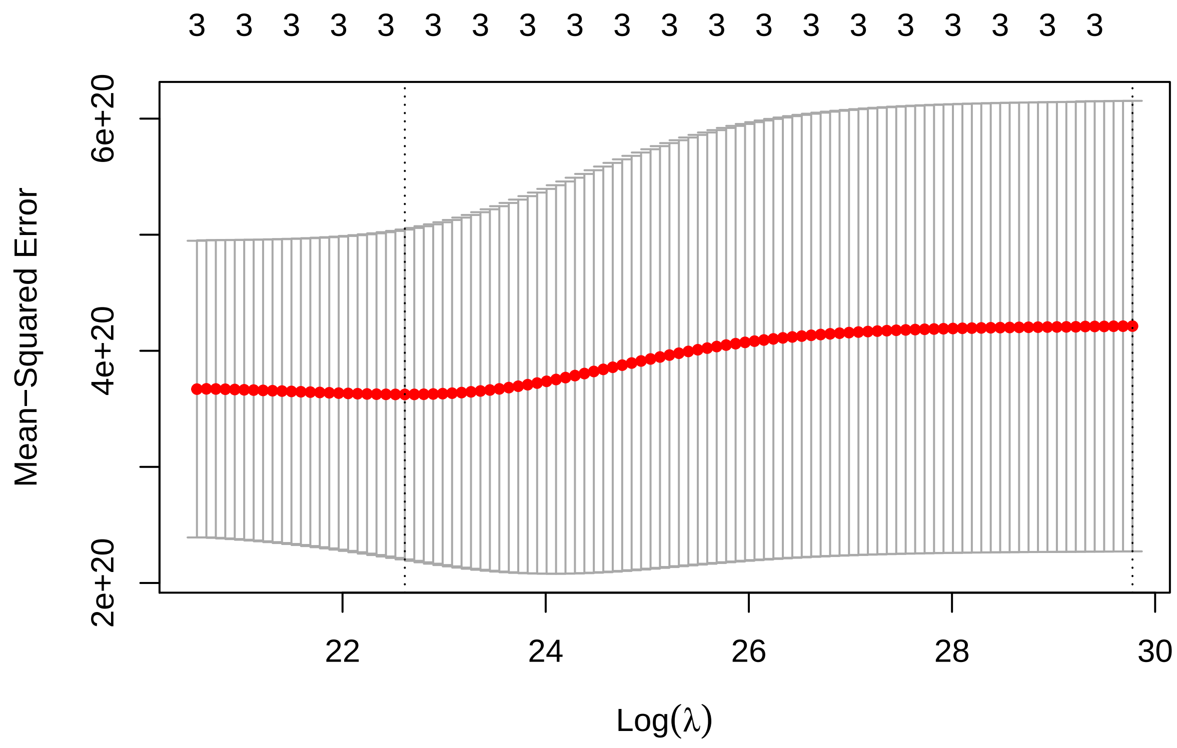
```r
set.seed(1)
cv.out3 <- cv.glmnet(x3[train3,], y3[train3], alpha = 0)
plot(cv.out3)
```



```r
bestlam3 <- cv.out3$lambda.min
bestlam3
```

```
## [1] 407052.8
```

```r
cv.out4 <- cv.glmnet(x4[train4,], y4[train4], alpha = 0)
plot(cv.out4)
```



```r
bestlam4 <- cv.out4$lambda.min
bestlam4
```

```
## [1] 6620092655
```

```r
ridge.pred3 <- predict(ridge.mod3, s = bestlam3, newx = x3[test3,])
mean((ridge.pred3 - y.test3)^2)
```

```
## [1] 2.26338e+12
```

```r
ridge.pred4 <- predict(ridge.mod4, s = bestlam4, newx = x4[test4,])
mean((ridge.pred4 - y.test4)^2)
```

```
## [1] 1.20962e+20
```

improvement? still large

Introduction

Turbulence is highly versatile motion that is often times difficult to predict and understanding fluid motion and its effect on natural problems poses a great challenge. Interpreting turbulence data is an incredibly important task in the engineering from understanding the cosmos to the cosmic cycle.

Parametric modeling is effective when we want to compactly represent features as model parameters. Unlike certain "black box" machine learning techniques, it offers a higher level of interpretability, which is especially useful for a practical setting.Rather than simply outputting a classification result, a parametric model allows us to investigate in more detail which aspects of turbulence differ between high and low particle cluster volumes.

In our project, we use linear and (. . .) and apply it to interpret the difference in model parameters in order to achieve the the following objectives:

1. Build a model that predict its particle cluster volume distribution in terms of the moments.

2. Investigate and interpret how model parameters affects the probability distribution for particle cluster volumes

Methodology

Results

Conclusion