Final Writeup

Kadriye Sude Almus

10/25/2020

Introduction

The Data

```r
data <- read.csv("data/data-train.csv")
```

Goals

Prediction: For a new parameter setting of (Re, F r, St), predict its particle cluster volume distribution in terms of its four raw moments.

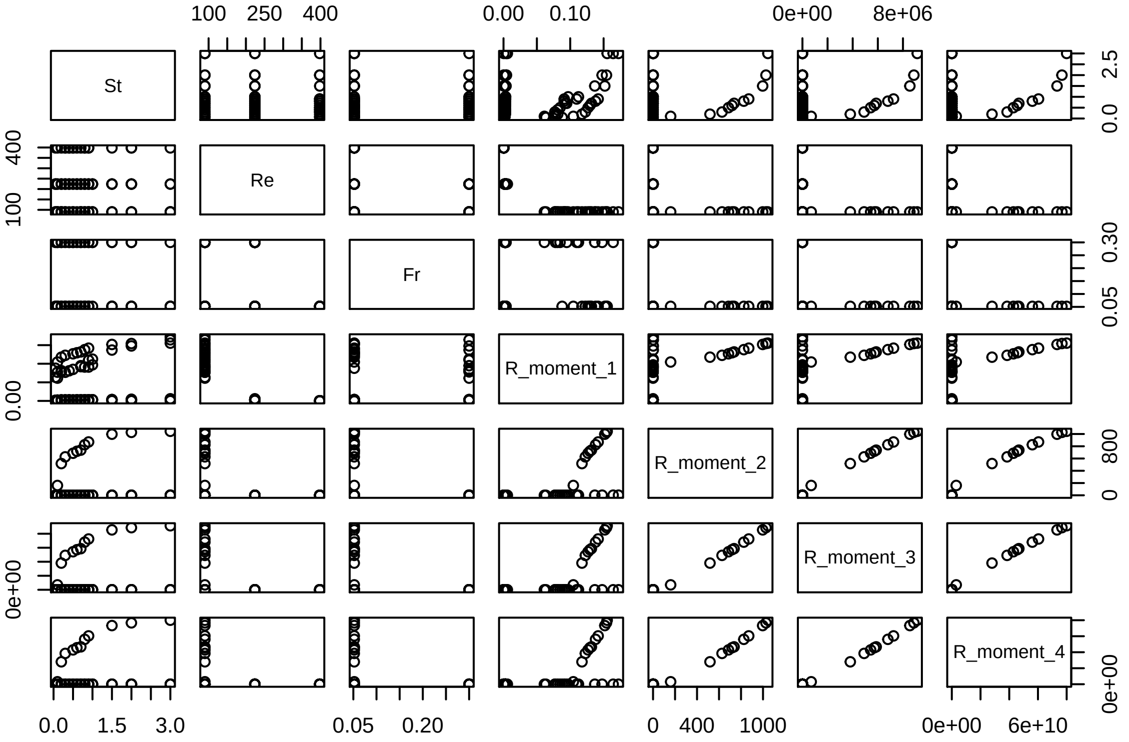
Inference: Investigate and interpret how each parameter affects the probability distribution for particle cluster volumes.

Methodology

Linear Modeling

Our univariate exploratory data analysis of Re, Fr, St, and each moment revealed that the R_moments are heavily right skewed, which poses a problem to potential linear analysis. We applied log transformations on each moment to obtain more normally distributed variables. The log-transformed R_moments are approximately normal, and it appears that each R_moment variable has somewhat of a linear relationship with St.

```r
pairs(data)
```



Accordingly, we fit a basic linear model onto each log-transformed response variable. While the adjusted R^2 value for R_moment_1 was very high at 0.9949, subsequent moments exibited decreasing adjusted R^2 values, with R_moment_4 having an adjusted R^2 value of 0.6518. We explored multicollinearity through VIFs for each model, which were very low. We also explored the addition of interaction terms to the model. The only interaction term which was significant for all R_moments was the interaction between Re and Fr. Constructing a linear model as such:

```
glm.inter <- lm(cbind(log(R_moment_1), log(R_moment_2), log(R_moment_3), log(R_moment_4)) ~  (St + fact
summary(glm.inter)
```

```
## Response log(R_moment_1) :
##
## Call:
## lm(formula = `log(R_moment_1)` ~ (St + factor(Re) + factor(Fr) +
##     factor(Re) * factor(Fr)), data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.48592 -0.00915  0.03880  0.07277  0.17182
##
## Coefficients: (1 not defined because of singularities)
##                           Estimate Std. Error  t value Pr(>|t|)
## (Intercept)               -2.27306    0.04110  -55.299  < 2e-16 ***
## St                         0.24989    0.01803   13.863  < 2e-16 ***
## factor(Re)224             -3.81588    0.05160  -73.948  < 2e-16 ***
## factor(Re)398             -5.98854    0.05621 -106.548  < 2e-16 ***
## factor(Fr)0.3             -0.26297    0.05622   -4.678 1.16e-05 ***
## factor(Fr)Inf             -0.32944    0.05787   -5.693 1.99e-07 ***
## factor(Re)224:factor(Fr)0.3  0.22050    0.07574    2.911  0.00466 **
## factor(Re)398:factor(Fr)0.3       NA         NA       NA       NA
## factor(Re)224:factor(Fr)Inf  0.40185    0.07759    5.179 1.63e-06 ***
## factor(Re)398:factor(Fr)Inf  0.50151    0.08366    5.995 5.58e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1312 on 80 degrees of freedom
## Multiple R-squared:  0.9969, Adjusted R-squared:  0.9966
## F-statistic:  3193 on 8 and 80 DF,  p-value: < 2.2e-16
##
##
## Response log(R_moment_2) :
##
## Call:
## lm(formula = `log(R_moment_2)` ~ (St + factor(Re) + factor(Fr) +
##     factor(Re) * factor(Fr)), data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.8551 -0.0206  0.3104  0.5102  1.0043
##
## Coefficients: (1 not defined because of singularities)
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   5.1869     0.3843  13.498  < 2e-16 ***
## St                            0.8340     0.1685   4.949 4.06e-06 ***
## factor(Re)224                -7.4387     0.4824 -15.420  < 2e-16 ***
## factor(Re)398               -11.3837     0.5254 -21.665  < 2e-16 ***
## factor(Fr)0.3                -6.4163     0.5256 -12.208  < 2e-16 ***
## factor(Fr)Inf                -6.6523     0.5410 -12.297  < 2e-16 ***
## factor(Re)224:factor(Fr)0.3   4.3872     0.7081   6.196 2.37e-08 ***
## factor(Re)398:factor(Fr)0.3       NA         NA      NA       NA
## factor(Re)224:factor(Fr)Inf   4.7181     0.7254   6.504 6.25e-09 ***
## factor(Re)398:factor(Fr)Inf   7.0758     0.7821   9.047 7.09e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.226 on 80 degrees of freedom
## Multiple R-squared:  0.9008, Adjusted R-squared:  0.8909
## F-statistic: 90.79 on 8 and 80 DF,  p-value: < 2.2e-16
##
##
## Response log(R_moment_3) :
##
## Call:
## lm(formula = `log(R_moment_3)` ~ (St + factor(Re) + factor(Fr) +
##     factor(Re) * factor(Fr)), data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.3570  -0.0586   0.4564   0.8018   1.6559
##
## Coefficients: (1 not defined because of singularities)
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  13.3986     0.6241  21.469  < 2e-16 ***
## St                            1.1740     0.2737   4.290 4.97e-05 ***
## factor(Re)224               -11.1636     0.7835 -14.249  < 2e-16 ***
## factor(Re)398               -17.0302     0.8534 -19.957  < 2e-16 ***
## factor(Fr)0.3               -12.4781     0.8536 -14.618  < 2e-16 ***
## factor(Fr)Inf               -12.7719     0.8786 -14.536  < 2e-16 ***
## factor(Re)224:factor(Fr)0.3   8.3648     1.1500   7.274 2.10e-10 ***
## factor(Re)398:factor(Fr)0.3       NA         NA      NA       NA
## factor(Re)224:factor(Fr)Inf   8.7718     1.1781   7.446 9.76e-11 ***
## factor(Re)398:factor(Fr)Inf  13.3707     1.2702  10.527  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.991 on 80 degrees of freedom
## Multiple R-squared:  0.8882, Adjusted R-squared:  0.877
## F-statistic: 79.44 on 8 and 80 DF,  p-value: < 2.2e-16
##
##
## Response log(R_moment_4) :
##
## Call:
## lm(formula = `log(R_moment_4)` ~ (St + factor(Re) + factor(Fr) +
##     factor(Re) * factor(Fr)), data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.4109   0.0031   0.5741   1.0506   2.2382
##
## Coefficients: (1 not defined because of singularities)
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  21.6950     0.8354  25.971  < 2e-16 ***
## St                            1.4690     0.3663   4.010 0.000135 ***
## factor(Re)224               -14.9060     1.0487 -14.214  < 2e-16 ***
## factor(Re)398               -22.7148     1.1422 -19.886  < 2e-16 ***
## factor(Fr)0.3               -18.4708     1.1425 -16.166  < 2e-16 ***
## factor(Fr)Inf               -18.8106     1.1760 -15.995  < 2e-16 ***
## factor(Re)224:factor(Fr)0.3  12.2758     1.5393   7.975 9.06e-12 ***
## factor(Re)398:factor(Fr)0.3       NA         NA      NA       NA
## factor(Re)224:factor(Fr)Inf  12.7559     1.5769   8.089 5.40e-12 ***
## factor(Re)398:factor(Fr)Inf  19.5683     1.7001  11.510  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.666 on 80 degrees of freedom
## Multiple R-squared:  0.8917, Adjusted R-squared:  0.8809
## F-statistic: 82.34 on 8 and 80 DF,  p-value: < 2.2e-16
```

Adding the interaction term between Re and Fr improved the fit of the model according to the adjusted Rˆ2 values, which are higher for every moment.

###Predictive performance of linear modeling

We split data into training and testing sets to evaluate the predictive ability of the models we explored. The linear models with the interaction term for Re and Fr outperformed any other linear model, producing lower test MSEs for every moment of R.

Split data into training and test sets

```
attach(data)
set.seed(3)
train_ind <- sample(x = nrow(data), size = 0.8 * nrow(data))
test_ind_neg <- -train_ind
training <- data[train_ind, ]
testing <- data[test_ind_neg, ]
```

Linear model using least squares & interaction term

```
fit.lm1 <- lm(log(R_moment_1) ~ (St + factor(Re) + factor(Fr) + factor(Re)*factor(Fr)), data = training
pred.lm1 <- predict(fit.lm1, testing)
```

```
## Warning in predict.lm(fit.lm1, testing): prediction from a rank-deficient fit
## may be misleading
```

```
mse_test1 <- mean((pred.lm1 - log(testing$R_moment_1))^2)

fit.lm2 <- lm(log(R_moment_2) ~ (St + factor(Re) + factor(Fr) + factor(Re)*factor(Fr)), data = training
pred.lm2 <- predict(fit.lm2, testing)
```

```
## Warning in predict.lm(fit.lm2, testing): prediction from a rank-deficient fit
## may be misleading
```

```
mse_test2 <- mean((pred.lm2 - log(testing$R_moment_2))^2)

fit.lm3 <- lm(log(R_moment_3) ~ (St + factor(Re) + factor(Fr) + factor(Re)*factor(Fr)), data = training
pred.lm3 <- predict(fit.lm3, testing)
```

```
## Warning in predict.lm(fit.lm3, testing): prediction from a rank-deficient fit
## may be misleading
```

```
mse_test3 <- mean((pred.lm3 - log(testing$R_moment_3))^2)

fit.lm4 <- lm(log(R_moment_4) ~ (St + factor(Re) + factor(Fr) + factor(Re)*factor(Fr)), data = training
pred.lm4 <- predict(fit.lm4, testing)
```

```
## Warning in predict.lm(fit.lm4, testing): prediction from a rank-deficient fit
## may be misleading
```

```
mse_test4 <- mean((pred.lm4 - log(testing$R_moment_4))^2)

mse_test1
```

```
## [1] 0.008822464
```

```
mse_test2
```

```
## [1] 1.396723
```

```
mse_test3
```

```
## [1] 3.184988
```

```
mse_test4
```

```
## [1] 5.272393
```

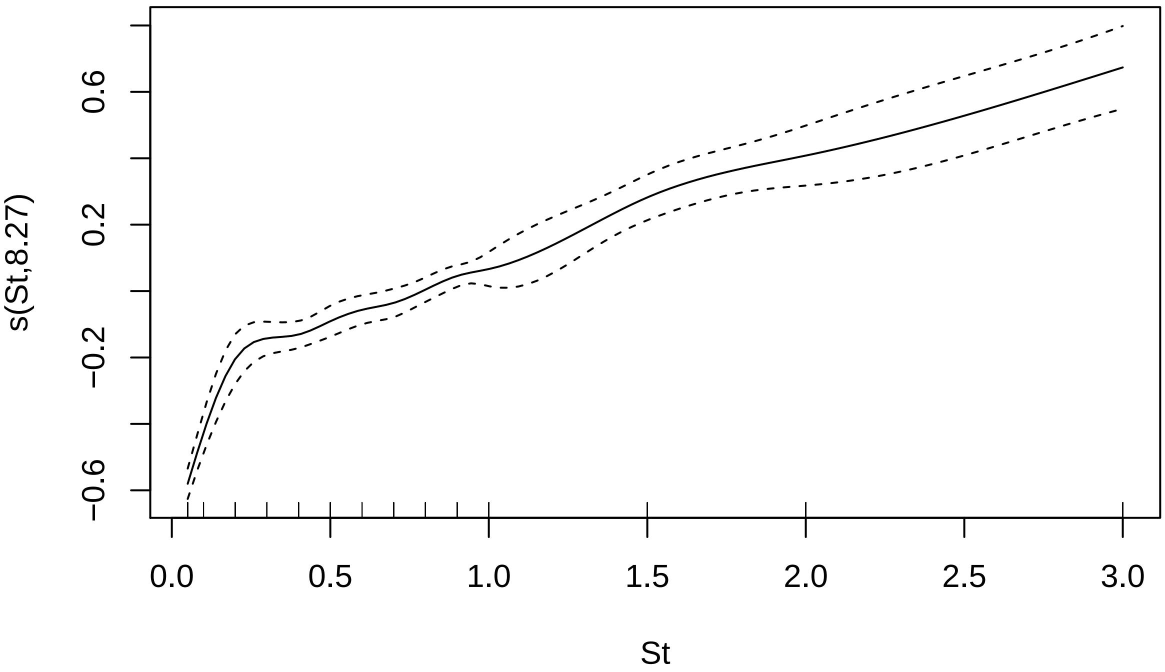
Having an interaction term significantly improved the test MSEs of the linear model.

Other model selection methods

We applied other model selection methods and nonlinear models such as principle components regression, partial least squares, regression tree, random forest, box cox transformations, and polynomial regressions. The model fits and predictive performances of these models were poor or the same as the linear regression model.

Splines and GAM

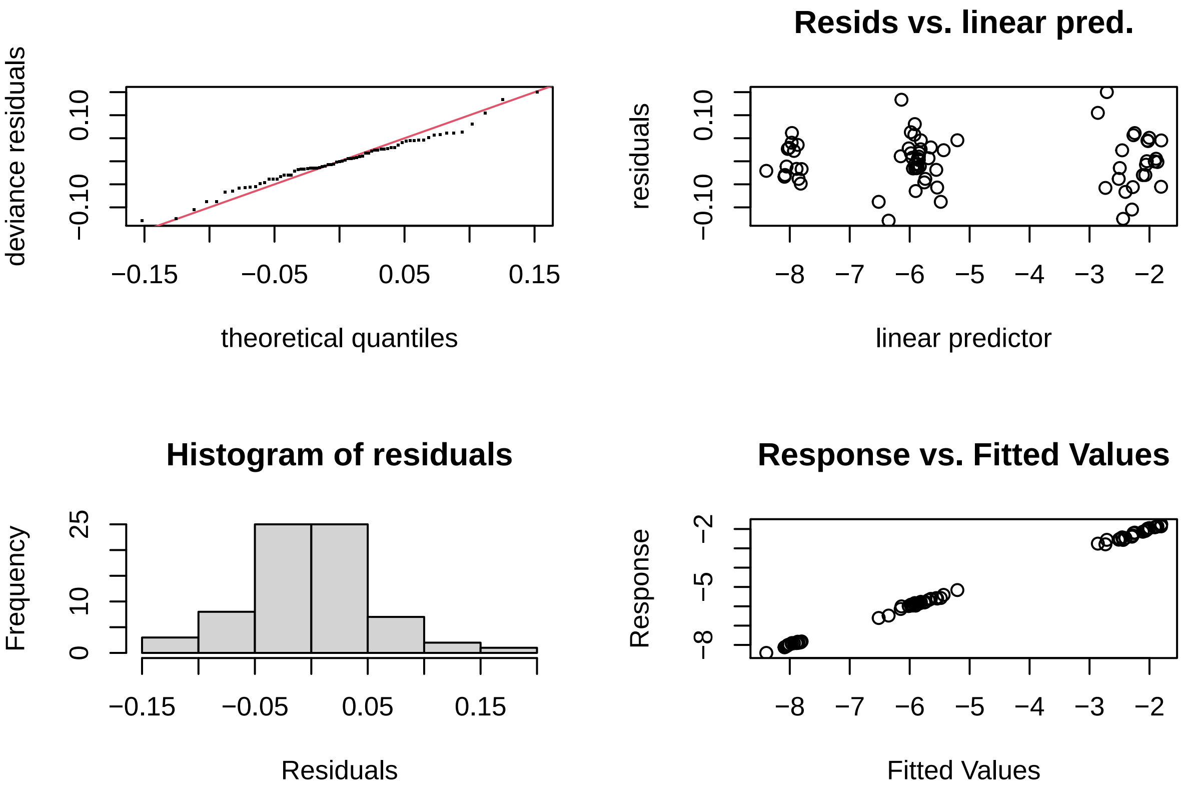
```
ftraining <- training
ftesting <- testing
ftraining$Fr <- factor(ftraining$Fr, levels = c(0.052, 0.300, Inf))
ftraining$Re <- factor(ftraining$Re, levels = c(90, 224, 398))
ftesting$Fr <- factor(ftesting$Fr, levels = c(0.052, 0.300, Inf))
ftesting$Re <- factor(ftesting$Re, levels = c(90, 224, 398))

gam.m1 = gam(log(R_moment_1) ~ s(St) + Re + Fr + St:Re + St: Fr + Re:Fr, data = ftraining)
plot(gam.m1)
```



```
summary(gam.m1)
```

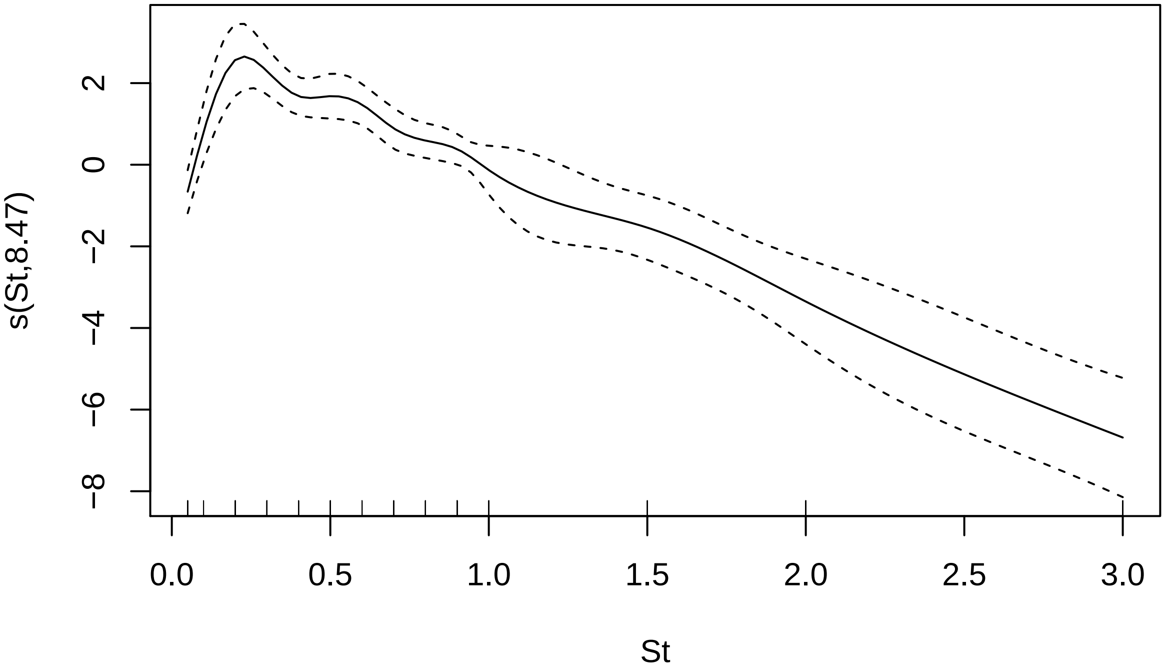
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## log(R_moment_1) ~ s(St) + Re + Fr + St:Re + St:Fr + Re:Fr
##
## Parametric coefficients:
##             Estimate Std. Error  t value Pr(>|t|)
## (Intercept) -1.86515    0.03215  -58.013  < 2e-16 ***
## Re224       -3.84039    0.03306 -116.168  < 2e-16 ***
## Re398       -6.01255    0.04587 -131.089  < 2e-16 ***
## Fr0.3       -0.44233    0.04321  -10.236 6.36e-14 ***
## FrInf       -0.41249    0.03755  -10.986 5.28e-15 ***
## Re90:St     -0.20504    0.03014   -6.804 1.15e-08 ***
## Re224:St    -0.17496    0.03148   -5.557 1.03e-06 ***
## Re398:St    -0.19949    0.03318   -6.012 2.02e-07 ***
## Fr0.3:St     0.19896    0.03391    5.868 3.38e-07 ***
## FrInf:St     0.13852    0.02226    6.224 9.41e-08 ***
## Re224:Fr0.3  0.21018    0.04158    5.054 6.02e-06 ***
## Re398:Fr0.3  0.00000    0.00000       NA       NA
## Re224:FrInf  0.34898    0.04056    8.603 1.79e-11 ***
## Re398:FrInf  0.48160    0.04884    9.861 2.27e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##         edf Ref.df     F p-value
## s(St) 8.271  8.727 181.7  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Rank: 21/23
## R-sq.(adj) =  0.999   Deviance explained = 99.9%
## GCV = 0.0053712  Scale est. = 0.003827  n = 71
```

```
gam.check(gam.m1)
```



```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 6 iterations.
## The RMS GCV score gradient at convergence was 1.297042e-06 .
## The Hessian was positive definite.
## Model rank =  21 / 23
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##          k'  edf k-index p-value
## s(St) 9.00 8.27    1.15    0.88
```

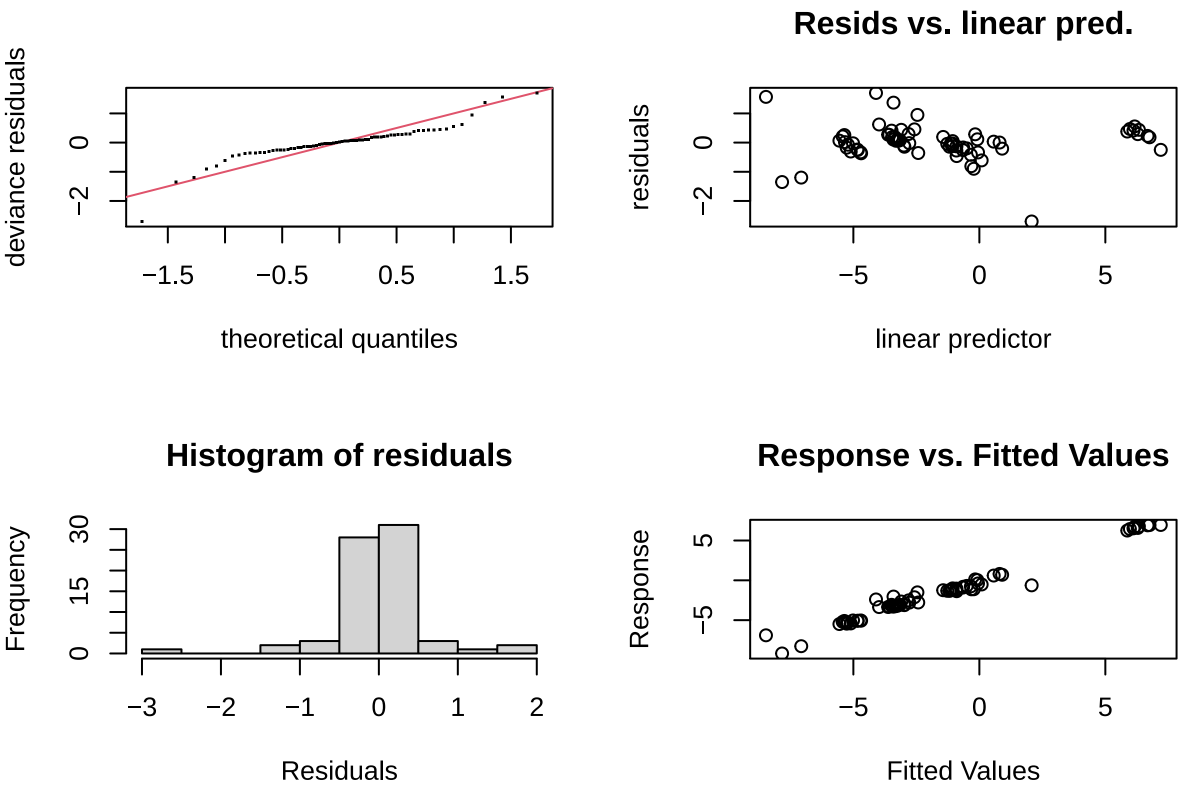
```
gam.m2 = gam(log(R_moment_2) ~ s(St) + Re + Fr + St:Re + St:Fr + Re:Fr, data = ftraining)
plot(gam.m2)
```



```
summary(gam.m2)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## log(R_moment_2) ~ s(St) + Re + Fr + St:Re + St:Fr + Re:Fr
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.54657    0.37518   6.788 1.24e-08 ***
## Re224        -7.11591    0.37585 -18.933  < 2e-16 ***
## Re398       -11.08527    0.52216 -21.230  < 2e-16 ***
## Fr0.3        -6.87397    0.49181 -13.977  < 2e-16 ***
## FrInf        -6.17375    0.42687 -14.463  < 2e-16 ***
## Re90:St       3.77928    0.35437  10.665 1.60e-14 ***
## Re224:St      3.63936    0.36826   9.883 2.19e-13 ***
## Re398:St      3.26669    0.38455   8.495 2.71e-11 ***
## Fr0.3:St      0.51261    0.38567   1.329    0.190
## FrInf:St     -0.07419    0.25302  -0.293    0.771
## Re224:Fr0.3   4.06642    0.47277   8.601 1.86e-11 ***
## Re398:Fr0.3   0.00000    0.00000      NA       NA
## Re224:FrInf   4.15555    0.46108   9.013 4.39e-12 ***
## Re398:FrInf   6.74397    0.55523  12.146  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##         edf Ref.df     F p-value
## s(St) 8.467  8.799 17.03  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Rank: 21/23
## R-sq.(adj) =  0.965   Deviance explained = 97.5%
## GCV = 0.69646  Scale est. = 0.49431   n = 71
```

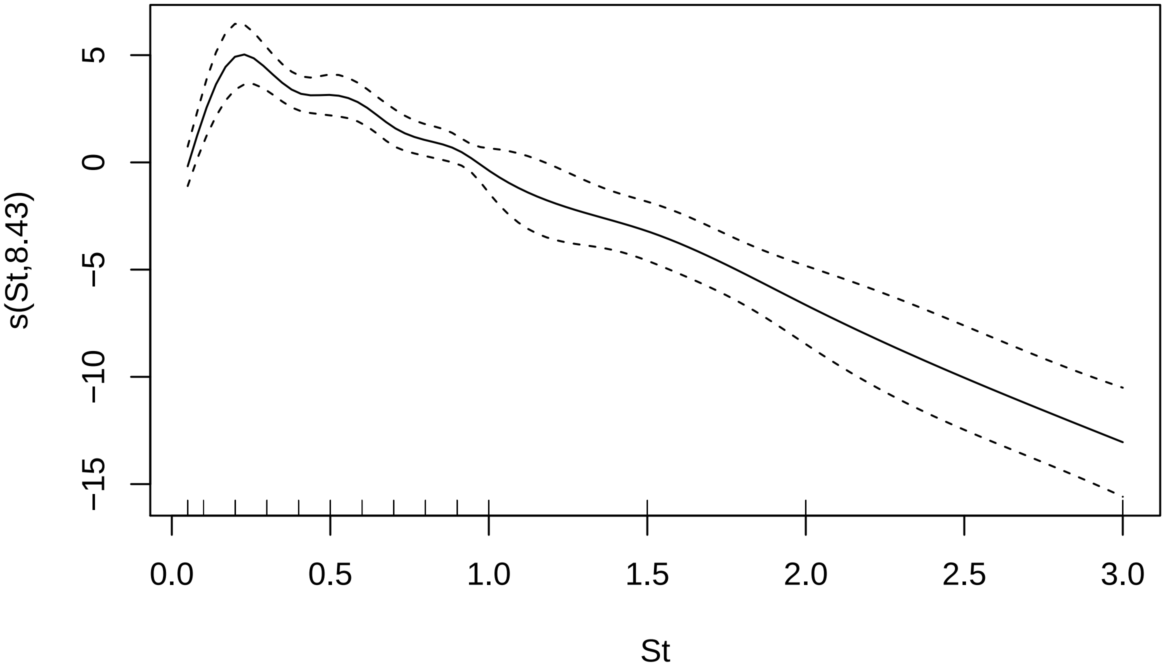
```r
gam.check(gam.m2)
```



```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 8 iterations.
## The RMS GCV score gradient at convergence was 3.715048e-07 .
## The Hessian was positive definite.
## Model rank =  21 / 23
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
```

```
##          k'  edf k-index p-value
## s(St) 9.00 8.47    1.08    0.78
```

```
gam.m3 = gam(log(R_moment_3) ~ s(St) + Re + Fr + St:Re + St:Fr + Re:Fr, data = ftraining)
plot(gam.m3)
```

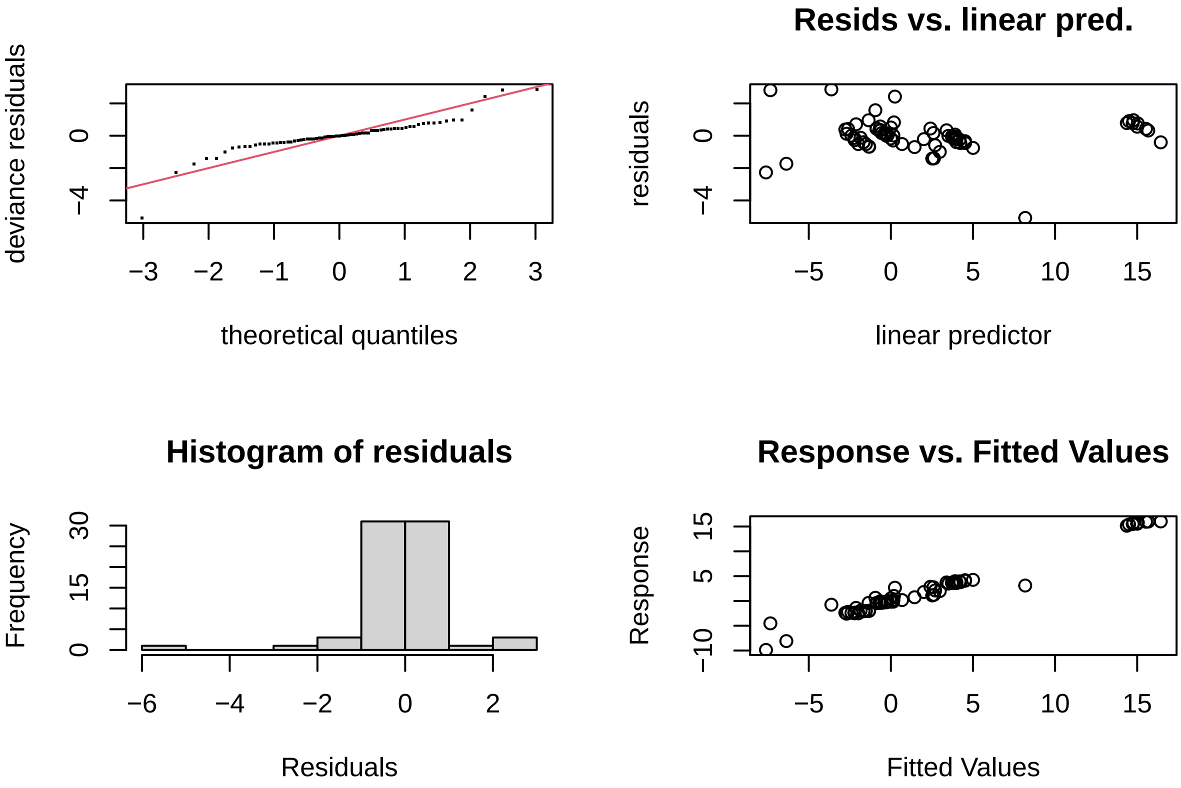


```
summary(gam.m3)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## log(R_moment_3) ~ s(St) + Re + Fr + St:Re + St:Fr + Re:Fr
##
## Parametric coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)     8.0050     0.6534  12.251  < 2e-16 ***
## Re224         -10.6005     0.6575 -16.124  < 2e-16 ***
## Re398         -16.4306     0.9132 -17.993  < 2e-16 ***
## Fr0.3         -13.0638     0.8601 -15.188  < 2e-16 ***
## FrInf         -11.7892     0.7467 -15.788  < 2e-16 ***
## Re90:St         7.1596     0.6164  11.614 7.29e-16 ***
## Re224:St        6.8820     0.6412  10.734 1.26e-14 ***
## Re398:St        6.2334     0.6705   9.296 1.63e-12 ***
## Fr0.3:St        0.6865     0.6746   1.018    0.314
## FrInf:St       -0.3003     0.4426  -0.678    0.501
## Re224:Fr0.3     7.8560     0.8270   9.499 8.08e-13 ***
## Re398:Fr0.3     0.0000     0.0000      NA       NA
## Re224:FrInf     7.8729     0.8066   9.761 3.30e-13 ***
## Re398:FrInf    12.7630     0.9712  13.141  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

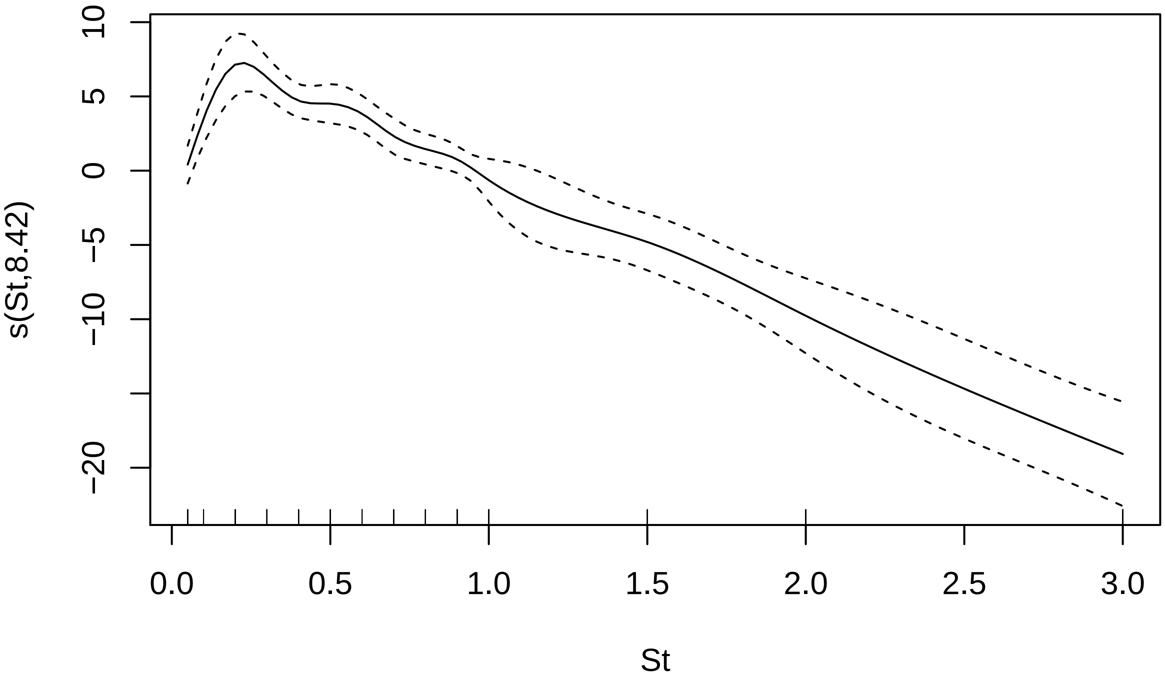
```
## Approximate significance of smooth terms:
##         edf Ref.df     F p-value
## s(St) 8.435  8.788 19.87  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Rank: 21/23
## R-sq.(adj) =  0.955   Deviance explained = 96.8%
## GCV =   2.13  Scale est. = 1.5127    n = 71
```

```
gam.check(gam.m3)
```



```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 7 iterations.
## The RMS GCV score gradient at convergence was 8.319095e-05 .
## The Hessian was positive definite.
## Model rank =  21 / 23
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##         k'  edf k-index p-value
## s(St) 9.00 8.43    1.08    0.72
```

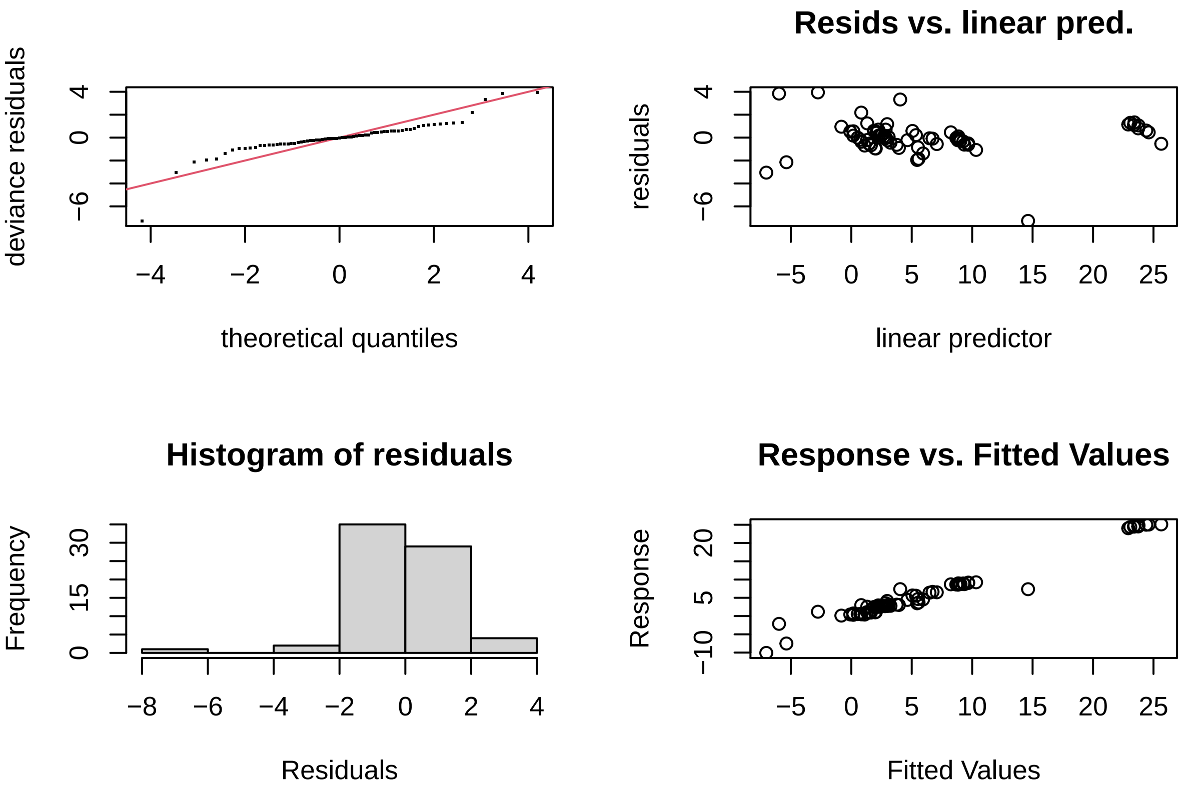
```
gam.m4 = gam(log(R_moment_4) ~ s(St) + Re + Fr + St:Re + St:Fr + Re:Fr, data = ftraining)
plot(gam.m4)
```



```r
summary(gam.m4)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## log(R_moment_4) ~ s(St) + Re + Fr + St:Re + St:Fr + Re:Fr
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  13.6971     0.9028  15.171  < 2e-16 ***
## Re224       -14.1498     0.9106 -15.538  < 2e-16 ***
## Re398       -21.8583     1.2647 -17.284  < 2e-16 ***
## Fr0.3       -19.1353     1.1913 -16.063  < 2e-16 ***
## FrInf       -17.3592     1.0342 -16.784  < 2e-16 ***
## Re90:St      10.3389     0.8511  12.148  < 2e-16 ***
## Re224:St      9.9495     0.8857  11.234 2.46e-15 ***
## Re398:St      9.0499     0.9271   9.762 3.27e-13 ***
## Fr0.3:St      0.8019     0.9343   0.858    0.395
## FrInf:St     -0.5199     0.6131  -0.848    0.400
## Re224:Fr0.3  11.6078     1.1455  10.134 9.28e-14 ***
## Re398:Fr0.3   0.0000     0.0000      NA       NA
## Re224:FrInf  11.5630     1.1172  10.350 4.49e-14 ***
## Re398:FrInf  18.7047     1.3453  13.904  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##         edf Ref.df     F p-value
## s(St) 8.416  8.782 22.52  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Rank: 21/23
## R-sq.(adj) =  0.954   Deviance explained = 96.7%
## GCV = 4.0851  Scale est. = 2.9023    n = 71
```

```
gam.check(gam.m4)
```



```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 7 iterations.
## The RMS GCV score gradient at convergence was 0.0001068624 .
## The Hessian was positive definite.
## Model rank =  21 / 23
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##         k'  edf k-index p-value
## s(St) 9.00 8.42    1.08    0.72
```

```
pred.gam1 <- predict(gam.m1, ftesting)
pred.gam2 <- predict(gam.m2, ftesting)
pred.gam3 <- predict(gam.m3, ftesting)
pred.gam4 <- predict(gam.m4, ftesting)

mse_gam1 <- mean((pred.gam1 - log(ftesting$R_moment_1))^2)
mse_gam2 <- mean((pred.gam2 - log(ftesting$R_moment_2))^2)
mse_gam3 <- mean((pred.gam3 - log(ftesting$R_moment_3))^2)
mse_gam4 <- mean((pred.gam4 - log(ftesting$R_moment_4))^2)
```

```
#mse_gam1 <- mean((exp(pred.gam1) - ftesting$R_moment_1)^2)
#mse_gam2 <- mean((exp(pred.gam2) - ftesting$R_moment_2)^2)
#mse_gam3 <- mean((exp(pred.gam3) - ftesting$R_moment_3)^2)
#mse_gam4 <- mean((exp(pred.gam4) - ftesting$R_moment_4)^2)

mse_gam1
```

```
## [1] 0.008102973
```

```
mse_gam2
```

```
## [1] 0.8611115
```

```
mse_gam3
```

```
## [1] 2.292613
```

```
mse_gam4
```

```
## [1] 4.100509
```

Final Model

Linear model vs gam with splines

Results

Predictive results of the final model + uncertainties and trade-offs

Scientific insight

Conclusion