01NAEX - Lecture 06
2^k Factorial Design

Exploring Experimental Designs for Multiple Factors

Jiri Franc

Czech Technical University
Faculty of Nuclear Sciences and Physical Engineering
Department of Mathematics

2^k Factorial Design

Last lesson: Introduction to 2^k Factorial Design

- 2^k factorial design is widely used in industrial experimentation, especially in the early stages of experimental work, when many factors are likely to be investigated.
- Special case of the general factorial design with k factors, all at 2 levels, usually called low (-) and high (+).
- Factors are fixed, the design is completely randomized, and the usual normality assumptions are satisfied.

Today's lesson:

- pyDOE2, pyDOE3 in Python, FrF2 package in R.
- Daniel Plot: Normal and Half-Normal Plot Effects.
- Pareto plot: Lenth's method.

2^k Factorial Design - Recap

General 2^k Factorial Design has

- k main effects
- $\binom{k}{2}$ two-factor interactions (first order interaction effects)
- $\binom{k}{3}$ three-factor interactions (second order interaction effects)
- $\vdots$
- 1 k-factor interaction

Unreplicated (single replicated) 2^k Factorial Design:

- If the factors are spaced too closely, it increases the chances that the noise will overwhelm the signal in the data.
- More aggressive spacing is usually best.
- Lack of replication causes potential problems in statistical testing:
  - With no replication, fitting the full model results in zero degrees of freedom for error.
  - Pooling high-order interactions to estimate error.
  - Normal probability plotting of effects (Daniels,1959)

2^3 Factorial Design in R - FrF2 package (Plasma Etch Rate Experiment)

Back to Plasma Etch Rate Experiment from the last lesson

```
> FrF2(2^3, 3, replications = 2, randomize = FALSE,
               factor.names = c("Gap", "Flow", "Power"))
creating full factorial with  8  runs ...
run.no run.no.std.rp         Gap Flow  Power
 1           1.1              -1   -1    -1
 2           2.1               1   -1    -1
 3           3.1              -1    1    -1
 4           4.1               1    1    -1
 5           5.1              -1   -1     1
 6           6.1               1   -1     1
 7           7.1              -1    1     1
 8           8.1               1    1     1
 9           1.2              -1   -1    -1
10           2.2               1   -1    -1
11           3.2              -1    1    -1
12           4.2               1    1    -1
13           5.2              -1   -1     1
14           6.2               1   -1     1
15           7.2              -1    1     1
16           8.2               1    1     1
class=design, type= full factorial
```

2^3 Factorial Design in Python - pyDOE3 package (Plasma Etch Rate Experiment)

Back to Plasma Etch Rate Experiment from the last lesson

```
import pandas as pd
from pyDOE3 import fullfact

# Generate the full factorial design
design = fullfact([2, 2, 2])
# Assign factor names
design_df = pd.DataFrame(design,
            columns=[’Gap’, ’Flow’, ’Power’])
# Map levels to actual values
level_mapping = {
    ’Gap’: {0: -1, 1: 1},
    ’Flow’: {0: -1, 1: 1},
    ’Power’: {0: -1, 1: 1}
}
for column in design_df.columns:
    design_df[column] = design_df[column].
        map(level_mapping[column])

print(design_df)
```

2^3 Factorial Design in R - FrF2 package (Plasma Etch Rate Experiment)

Care about settings: `randomize = TRUE` or `randomize = FALSE`

```
> FrF2(2^3, 3, replications = 2, randomize = T,
               factor.names = c("Gap", "Flow", "Power"))
creating full factorial with  8  runs ...
run.no  run.no.std.rp    Gap Flow  Power
 1           7.1         -1    1     1
 2           6.1          1   -1     1
 3           3.1         -1    1    -1
 4           5.1         -1   -1     1
 5           4.1          1    1    -1
 6           8.1          1    1     1
 7           2.1          1   -1    -1
 8           1.1         -1   -1    -1
 9           3.2         -1    1    -1
10           2.2          1   -1    -1
11           6.2          1   -1     1
12           5.2         -1   -1     1
13           8.2          1    1     1
14           7.2         -1    1     1
15           4.2          1    1    -1
16           1.2         -1   -1    -1
class=design, type= full factorial
```

2^k Factorial Design in R - FrF2 package

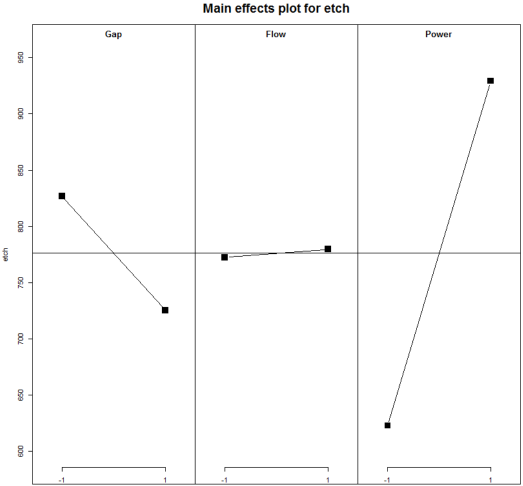
Back to Plasma Etch Rate Experiment

```
k = 3
plan <-  FrF2(2^k, k, replications = 2, randomize = FALSE,
factor.names = c("Gap", "Flow", "Power"))
plan <- add.response(plan, etch)
> plan
Gap Flow Power etch          Gap Flow Power etch
1   -1   -1     -1   550      9    -1   -1     -1   604
2    1   -1     -1   669      10    1   -1     -1   650
3   -1    1     -1   633      11   -1    1     -1   601
4    1    1     -1   642      12    1    1     -1   635
5   -1   -1      1 1037      13   -1   -1      1 1052
6    1   -1      1   749      14    1   -1      1   868
7   -1    1      1 1075      15   -1    1      1 1063
8    1    1      1   729      16    1    1      1   860
```

Analysis - same as in the last lesson.

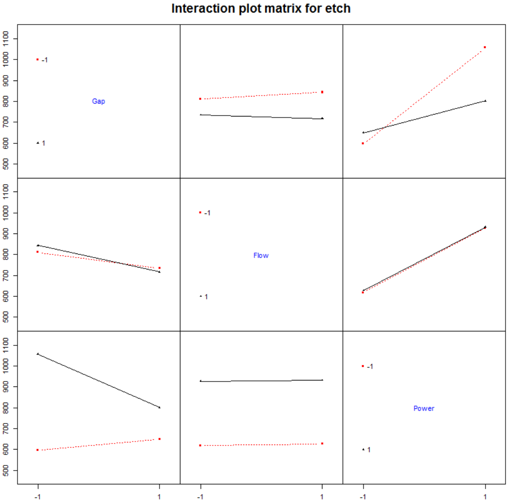
2^k Factorial Design in R - FrF2 package

```
MEPlot(plan)
```



2^k Factorial Design in R - FrF2 package

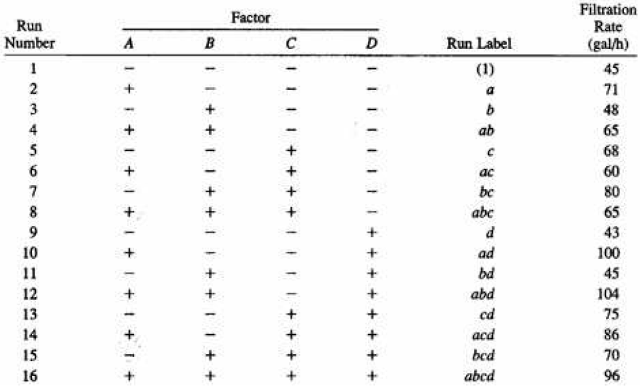
```
IAPlot(plan)
```



Unreplicated 2^k Factorial Designs

The 2^4 factorial design was used to investigate the effects of four factors on the filtration rate of a resin for a chemical process plant. The factors are:

A: temperature,
B: pressure,
C: concentration of chemical formaldehyde,
D: stirring rate.



Run Number	Factor				Run Label	Filtration Rate (gal/h)
	A	*B*	*C*	*D*		
1	−	−	−	−	(1)	45
2	+	−	−	−	*a*	71
3	−	+	−	−	*b*	48
4	+	+	−	−	*ab*	65
5	−	−	+	−	*c*	68
6	+	−	+	−	*ac*	60
7	−	+	+	−	*bc*	80
8	+	+	+	−	*abc*	65
9	−	−	−	+	*d*	43
10	+	−	−	+	*ad*	100
11	−	+	−	+	*bd*	45
12	+	+	−	+	*abd*	104
13	−	−	+	+	*cd*	75
14	+	−	+	+	*acd*	86
15	−	+	+	+	*bcd*	70
16	+	+	+	+	*abcd*	96

$2^4 = 16$ runs were made in random order.

Unreplicated 2^k Factorial Designs

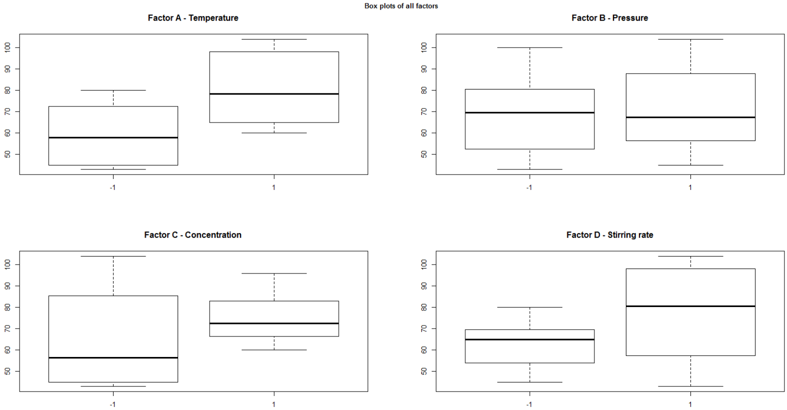
Pilot Plant Filtration Rate Experiment - FrF2 package

```
> rate=FrF2(2^4, 4, replications = 1,
       randomize = FALSE,factor.names = c("A", "B", "C", "D"
> Filtration = rate_experiment$Rate
> rate       = add.response(rate, Filtration)
> summary(rate)
Experimental design of type  full factorial 16  runs

Factor settings:      Responses:
A  B  C  D          [1] Filtration
1 -1 -1 -1 -1
2  1  1  1  1
The design itself:
   A  B  C  D Filtration    A  B  C  D Filtration
1  -1 -1 -1 -1         45   -1 -1 -1  1         43
2   1 -1 -1 -1         71    1 -1 -1  1        100
3  -1  1 -1 -1         48   -1  1 -1  1         45
4   1  1 -1 -1         65    1  1 -1  1        104
5  -1 -1  1 -1         68   -1 -1  1  1         75
6   1 -1  1 -1         60    1 -1  1  1         86
7  -1  1  1 -1         80   -1  1  1  1         70
8   1  1  1 -1         65    1  1  1  1         96
```

Pilot Plant Filtration Rate Experiment

Box plot

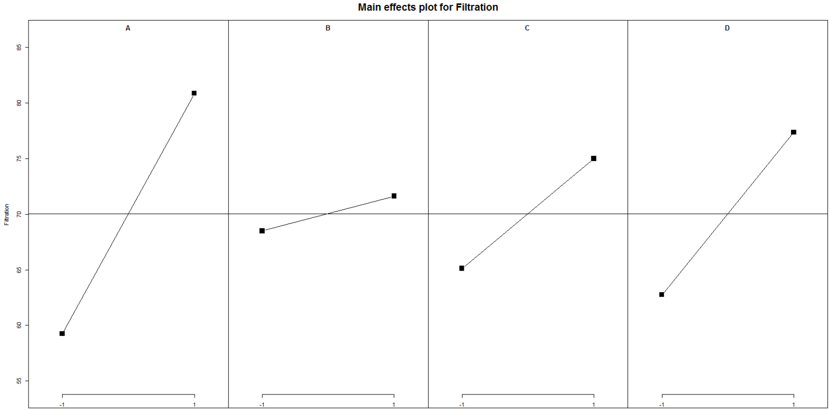
```
boxplot(Filtration ~ A , main = "Factor A - Temperature")
```



Pilot Plant Filtration Rate Experiment

Main Effects plotfor response variable

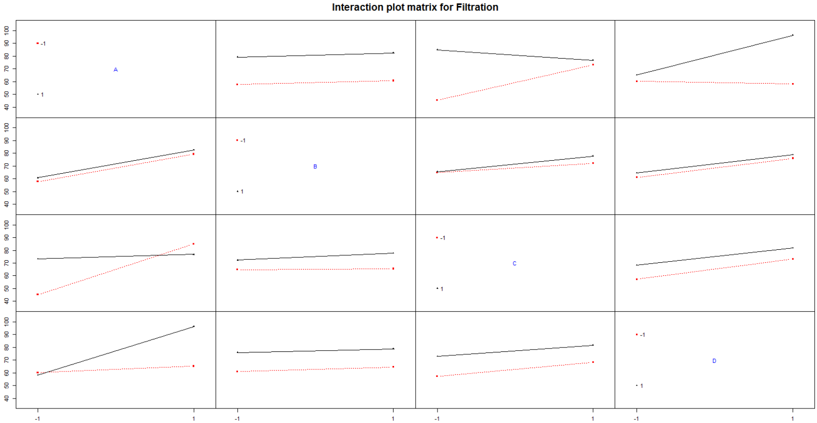
```
MEPlot(plan)
```



Pilot Plant Filtration Rate Experiment

Interaction Plot matrix for response variable

```
IAPlot(plan)
```



Pilot Plant Filtration Rate Experimene

ANOVA table - model with all factors and interactions

```
anova(aov(Filtration~A*B*C*D,  data=rate))
  Analysis of Variance Table, Response: Filtration
               Df  Sum Sq Mean Sq F value Pr(>F)
  A             1 1870.56 1870.56
  B             1   39.06   39.06
  C             1  390.06  390.06
  D             1  855.56  855.56
  A:B           1    0.06    0.06
  A:C           1 1314.06 1314.06
  B:C           1   22.56   22.56
  A:D           1 1105.56 1105.56
  B:D           1    0.56    0.56
  C:D           1    5.06    5.06
  A:B:C         1   14.06   14.06
  A:B:D         1   68.06   68.06
  A:C:D         1   10.56   10.56
  B:C:D         1   27.56   27.56
  A:B:C:D       1    7.56    7.56
  Residuals     0    0.00
  Warning message: ANOVA F-tests are unreliable
```

Pilot Plant Filtration Rate Experimene

ANOVA table - model with all factors and interactions in Python

```
import statsmodels.api as sm
from statsmodels.formula.api import ols

# Define the model
model = ols('EtchRate ~ Gap * Flow * Power', data=design_df).f

# Perform ANOVA
anova_table = sm.stats.anova_lm(model, typ=2)
print(anova_table)
```

Pilot Plant Filtration Rate Experimene: LM - all factors and interactions

```
> summary(lm(Filtration~A*B*C*D,  data=rate))
ALL 16 residuals are 0: no residual degrees of freedom!
             Estimate    Std. Error t value Pr(>|t|)
(Intercept)  70.0625           NA      NA       NA
A1           10.8125           NA      NA       NA
B1            1.5625           NA      NA       NA
C1            4.9375           NA      NA       NA
D1            7.3125           NA      NA       NA
A1:B1         0.0625           NA      NA       NA
A1:C1        -9.0625           NA      NA       NA
B1:C1         1.1875           NA      NA       NA
A1:D1         8.3125           NA      NA       NA
B1:D1        -0.1875           NA      NA       NA
C1:D1        -0.5625           NA      NA       NA
A1:B1:C1      0.9375           NA      NA       NA
A1:B1:D1      2.0625           NA      NA       NA
A1:C1:D1     -0.8125           NA      NA       NA
B1:C1:D1     -1.3125           NA      NA       NA
A1:B1:C1:D1   0.6875           NA      NA       NA
Residual standard error: NaN on 0 degrees of freedom
Multiple R-squared: 1, Adjusted R-squared: NaN
F-statistic: NaN on 15 and 0 DF, p-value: NA
```

Analysis of significant effects

We can’t use usual way because:

- ANOVA gives SS but no error estimate!
- LM gives effect estimates but no error estimate!
- We try to adjust a 2^k dimensional model to 2^k observations, leaving 0 dimensions (degrees of freedom) for the error estimate.

We don’t know σ and we can’t estimate it. We know that

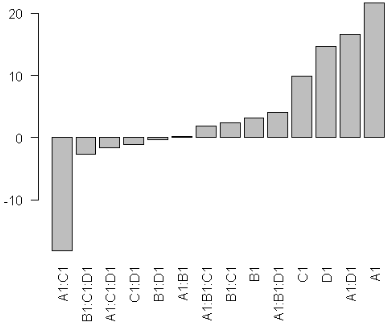
- the effects estimates are normally distributed with mean 0 and common standard deviation σ.
- the effect estimates are independent.

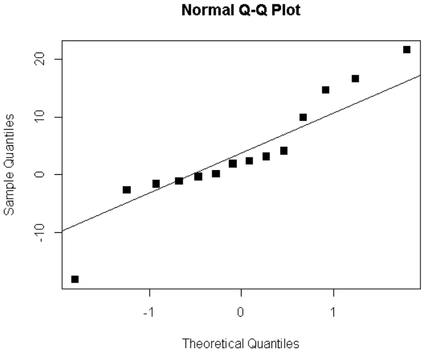
Lets make *QQ*-plot and look for the outliers - suspected significant effects.

Analysis of significant effects

Make QQ plot from Linear Model coefficients (without intercept) manually:

```
> model0 = lm(2*Filtration~A*B*C*D,  data=rate)
> barplot(sort(model0$coeff[2:(2^4-1)]),las = 2)
> qqnorm(model0$coeff[2:(2^4-1)],cex = 1.3,pch = 15)
> qqline(model0$coeff[2:(2^4-1)],cex = 1.3,pch = 15)
```





Analysis of significant effects

In unreplicated 2^k factorial design with high-order interactions we can use R build-in functions:

- Daniel normal probability plot;
- The Half-Normal Plot Effects;
- Lenth's method - Pareto plot.

Required packages in R: FrF2, qualityTools

Daniel plot

Daniel plot is the normal probability plot of the estimates of the effects.

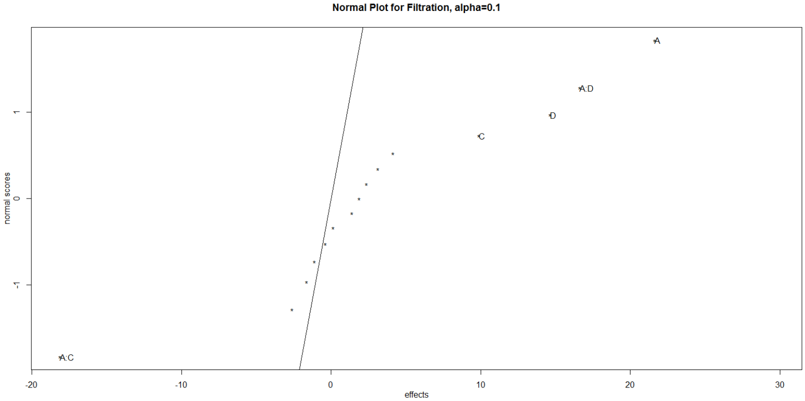
The effects that are negligible are normally distributed, with mean zero and variance σ^2 and will tend to fall along a straight line on this plot, whereas significant effects will have nonzero means and will not lie along straight line.

1. **Negligible effects**: lie along the qqline.
2. **Important effects**: lie far from the qqline.

Pilot Plant Filtration Rate Experiment

Daniel Plot (Classical effects qqplot) with alpha = 0.1, qqline, and only significant factors

```
qqplot(DanielPlot(rate,alpha=0.1)$x,DanielPlot(rate)$y)
qqline(DanielPlot(rate,alpha=0.1)$y)
```



The Half-Normal Plot Effects

Alternative plot to the Normal probability plot:

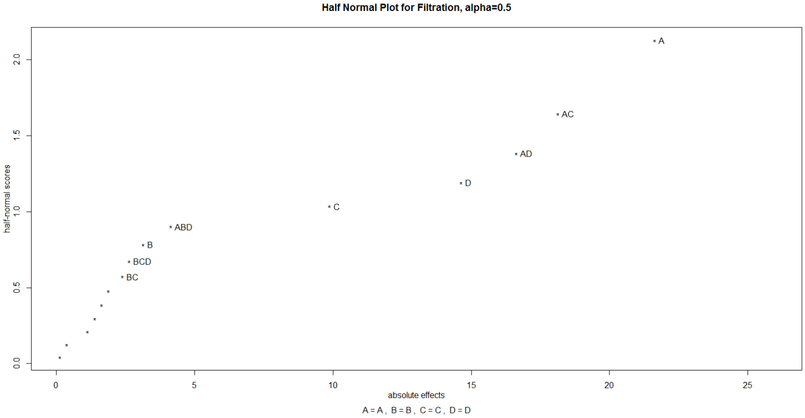
It is a plot of absolute value of the effects estimates against their cumulative normal probabilities.

The straight line on the half-normal plot always passes through the origin and should pass close to the fiftieth percentile data value. Bettet for interpretation with a few effects estimates (for example 8-run design).

Pilot Plant Filtration Rate Experimene

Half normal plot of effects

```
DanielPlot(rate,code=TRUE,alpha=0.1,half=TRUE)
```



Lenth's method - Pareto plot

Method proposed by Lenth (1989), sometime called Pareto plot has good power to detect significant effects.

Suppose we have m contrast of interest $c_1, c_2, \ldots c_m$. For 2^k unreplicated factor design $m = 2^k - 1$. Lenth's method estimates the variance of a contrast from the smallest contrast estimate.

$$s_0 = 1.5\text{median}(|c_j|) \quad PSE = 1.5\text{median}(|c_j|\,,\ |c_j| < 2.5s_0)$$

PSE is called pseudostandard error and it should be reasonable estimator of the contrast variance. An individual contrast is compared to the **margin of error**:

$$ME = t_{0.025,d}PSE, \quad d = \frac{m}{3}$$

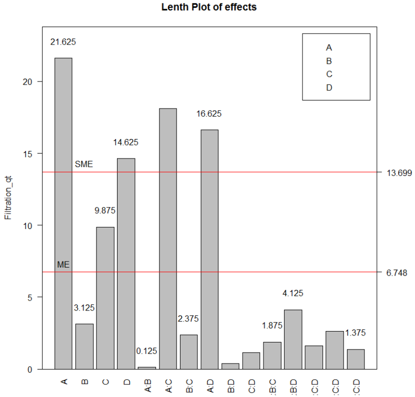
or to the **simultaneous margin of error**

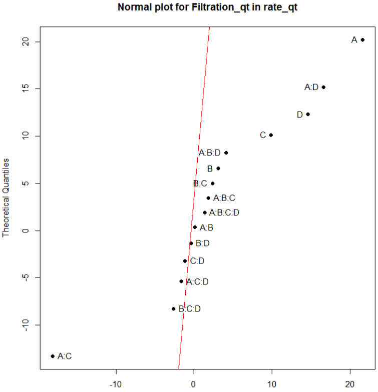
$$SME = t_{\gamma,d}PSE, \quad \gamma = 1 - \frac{(1 + 0.95^{\frac{1}{m}})}{2}$$

Pilot Plant Filtration Rate Experimene - qualityTools library

```
rate_pareto = fracDesign(k = 4, replicates = 1)
response(rate_pareto) = rate$Filtration[rate_pareto[,1]]
paretoPlot(rate_pareto) normalPlot(rate_pareto)
```

$PSE = 1.5 \times |1.75| = 2.625$, $ME = 2.571 \times 2.625 = 6.748$, $SME = 5.219 \times 2.625 = 13.699$





Pilot Plant Filtration Rate Experimene

ANOVA table - model without factor B

By having dropped B totally, we obtain a 2^3 design with 2 replicates per cell.

```
> anova(aov(Filtration~A*C*D,  data=rate))
Analysis of Variance Table

Response: Filtration
Df   Sum Sq Mean Sq F value    Pr(>F)
A          1 1870.56 1870.56 83.3677 1.667e-05 ***
C          1  390.06  390.06 17.3844 0.0031244 **
D          1  855.56  855.56 38.1309 0.0002666 ***
A:C        1 1314.06 1314.06 58.5655 6.001e-05 ***
A:D        1 1105.56 1105.56 49.2730 0.0001105 ***
C:D        1    5.06    5.06  0.2256 0.6474830
A:C:D      1   10.56   10.56  0.4708 0.5120321
Residuals  8  179.50   22.44
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1  1
>
```

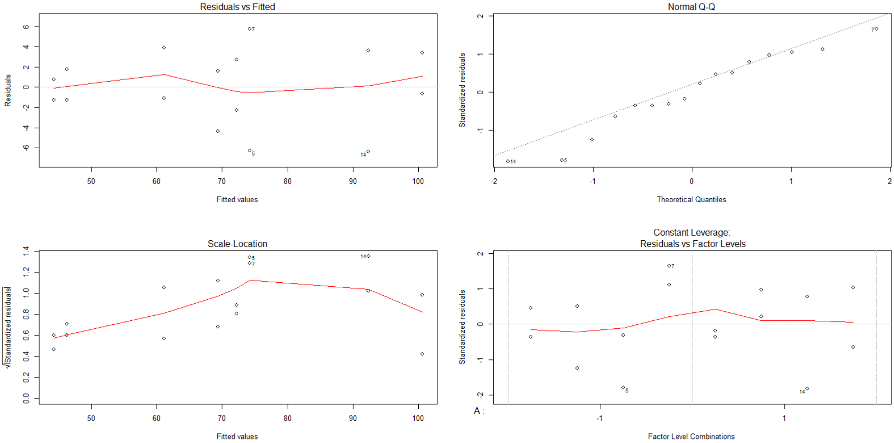
Pilot Plant Filtration Rate Experimene

ANOVA table - final model

```
> anova(aov(Filtration~A*C+A*D,  data=rate))
Analysis of Variance Table

Response: Filtration
Df  Sum Sq Mean Sq F value    Pr(>F)
A          1 1870.56 1870.56  95.865 1.928e-06 ***
C          1  390.06  390.06  19.990  0.001195 **
D          1  855.56  855.56  43.847 5.915e-05 ***
A:C        1 1314.06 1314.06  67.345 9.414e-06 ***
A:D        1 1105.56 1105.56  56.659 1.999e-05 ***
Residuals 10  195.12   19.51
--
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
>
```

Pilot Plant Filtration Rate Experiment

Model validation



+ perform all relevant statistical hypothesis testing.

Pilot Plant Filtration Rate Experimene

Another approach - omitting the highest interaction

```
>anova(aov(Filtration~(.)^3,  data=rate))
Analysis of Variance Table, Response: Filtration
Df   Sum Sq Mean Sq  F value  Pr(>F)
A          1 1870.56 1870.56 247.3471 0.04042 *
B          1   39.06   39.06   5.1653 0.26388
C          1  390.06  390.06  51.5785 0.08808 .
D          1  855.56  855.56 113.1322 0.05968 .
A:B        1    0.06    0.06   0.0083 0.94228
A:C        1 1314.06 1314.06 173.7603 0.04820 *
A:D        1 1105.56 1105.56 146.1901 0.05253 .
B:C        1   22.56   22.56   2.9835 0.33410
B:D        1    0.56    0.56   0.0744 0.83050
C:D        1    5.06    5.06   0.6694 0.56345
A:B:C      1   14.06   14.06   1.8595 0.40282
A:B:D      1   68.06   68.06   9.0000 0.20483
A:C:D      1   10.56   10.56   1.3967 0.44707
B:C:D      1   27.56   27.56   3.6446 0.30718
Residuals  1    7.56    7.56
```

Continue only with significant variables.

Pilot Plant Filtration Rate Experiment - Regression analysis

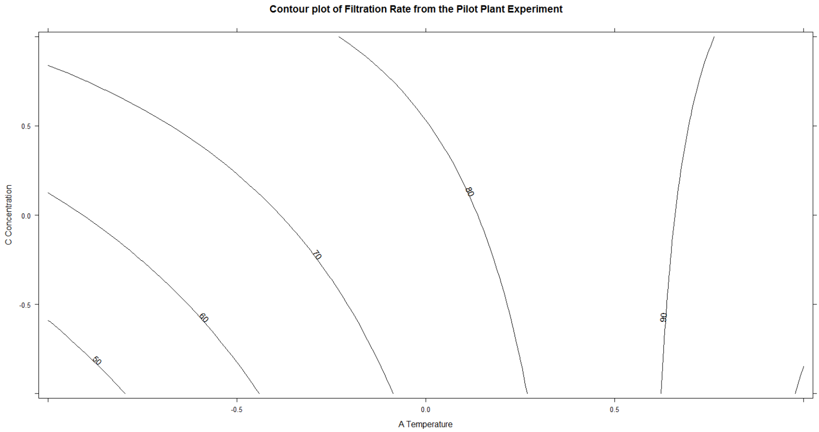
Preparation for countour plots (Swap factors for numerics)

```
> rate$A.num <- 2*(as.numeric(rate$A)-1.5)
> rate$C.num <- 2*(as.numeric(rate$C)-1.5)
> rate$D.num <- 2*(as.numeric(rate$D)-1.5)
> rate.lm      <- lm(Filtration~A.num*C.num+A.num*D.num,data=rate)
> summary(rate.lm)

Coefficients: Estimate Std. Error t value Pr(>|t|)
(Intercept)   70.062      1.104  63.444 2.30e-14 ***
A.num         10.812      1.104   9.791 1.93e-06 ***
C.num          4.938      1.104   4.471   0.0012 **
D.num          7.313      1.104   6.622 5.92e-05 ***
A.num:C.num   -9.063      1.104  -8.206 9.41e-06 ***
A.num:D.num    8.312      1.104   7.527 2.00e-05 ***
Residual standard error: 4.417 on 10 degrees of freedom
Multiple R-squared: 0.966,  Adjusted R-squared: 0.9489
F-statistic: 56.74 on 5 and 10 DF,  p-value: 5.14e-07

> tmp          <- list(A.num=seq(-1,1,by=.05),C.num=seq(-1,1,by=0.05
D.num=seq(-1,1,by=0.05),data=rate)
> new.data     <- expand.grid(tmp)
> new.data$fit<- predict(rate.lm,new.data)
```

Pilot Plant Filtration Rate Experiment

Countour plot: A(temperature) and C(concentration) interaction only

```
contourplot(fit~A.num*C.num,new.data,xlab="A Temperature",
ylab="C Concentration",
main="Contour plot of Filtration Rate from the Pilot Plant Experime
```



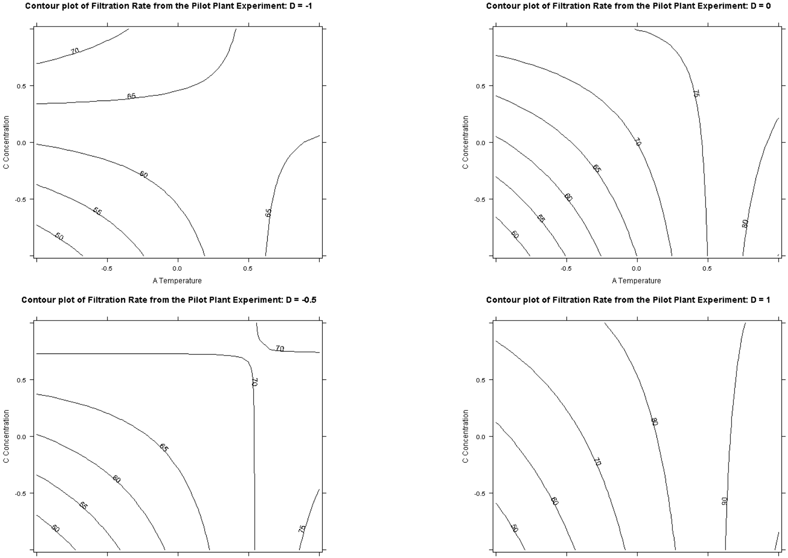
Contour plot of Filtration Rate from the Pilot Plant Experiment

No specification in the code means last value of D, i.e. D = 1.

Pilot Plant Filtration Rate Experiment

Countour plot: A(temperature) and C(concentration) for different D's.

```
contourplot(fit~A.num*C.num,new.data[new.data$D.num == -1,])
```



Contour plot of Filtration Rate from the Pilot Plant Experiment: D = -1

Contour plot of Filtration Rate from the Pilot Plant Experiment: D = 0

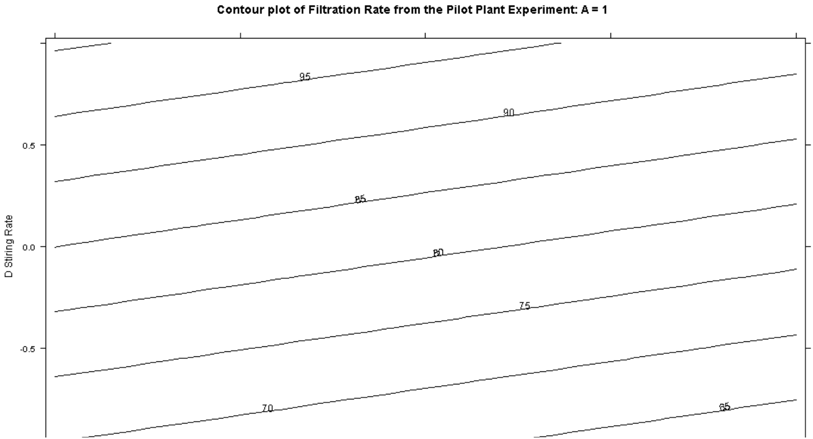
Contour plot of Filtration Rate from the Pilot Plant Experiment: D = -0.5

Contour plot of Filtration Rate from the Pilot Plant Experiment: D = 1

Pilot Plant Filtration Rate Experiment

Countour plot: C(concentration) and D(stirring rate) interaction only

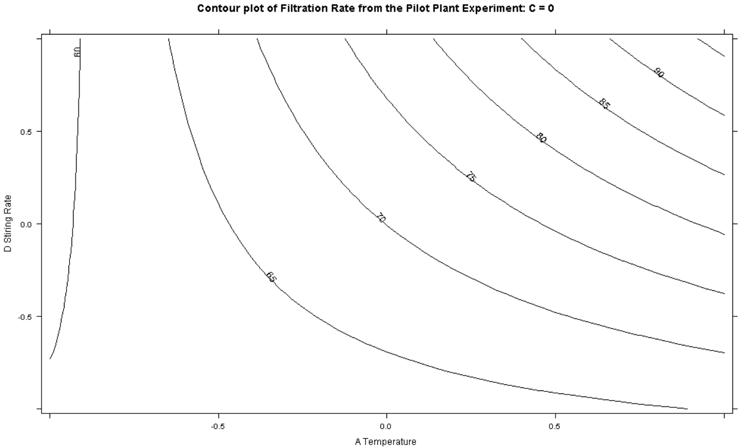
```
> contourplot(fit~C.num*D.num,new.data[new.data$A.num == 1,],
 xlab="C Concentration",ylab="D Stiring Rate",
 main = "Contour plot of Filtration Rate from the Pilot Plant Exper
 A = 1")
```



Pilot Plant Filtration Rate Experiment

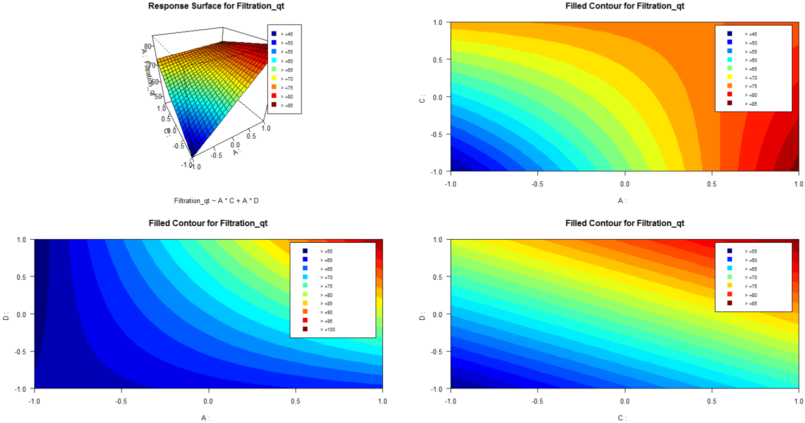
Countour plot: A(temperature) and D(stirring rate) interaction only

```
contourplot(fit~A.num*D.num,new.data[new.data$C.num == 0,],
xlab="A Temperature",ylab="D Stiring Rate",
main="Contour plot of Filtration Rate from the Pilot Plant Experiment
:C = 0")
```



Pilot Plant Filtration Rate Experiment - qualityTools library

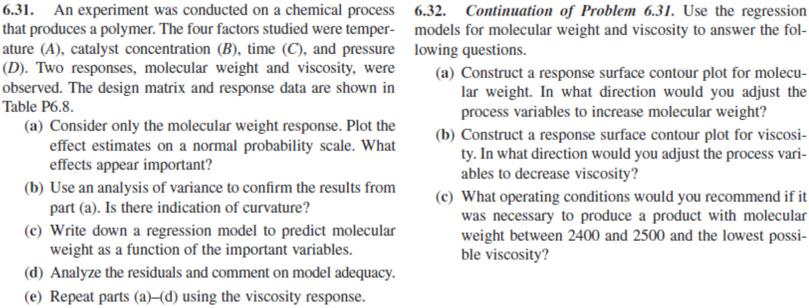
```
wirePlot(A, C, Filtration_qt, data = rate_qt)
contourPlot(A, C, Filtration_qt, data = rate_qt)
```



Missing factor is equal to 0 in all three cases!

Today Exercise

Solve problems 6.31 and 6.32 from the chapter 6,
D. C. Montgomery DAoE - 8. edition.



6.31. An experiment was conducted on a chemical process that produces a polymer. The four factors studied were temperature (A), catalyst concentration (B), time (C), and pressure (D). Two responses, molecular weight and viscosity, were observed. The design matrix and response data are shown in Table P6.8.

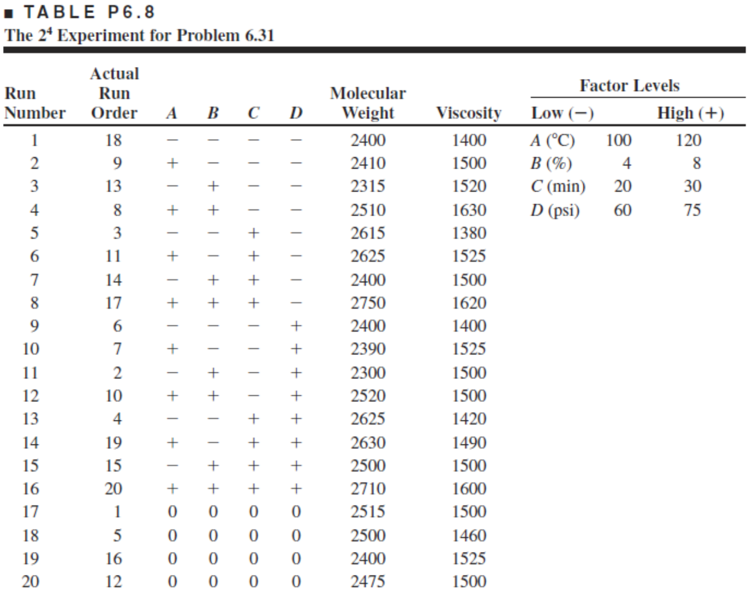
- **(a)** Consider only the molecular weight response. Plot the effect estimates on a normal probability scale. What effects appear important?
- **(b)** Use an analysis of variance to confirm the results from part (a). Is there indication of curvature?
- **(c)** Write down a regression model to predict molecular weight as a function of the important variables.
- **(d)** Analyze the residuals and comment on model adequacy.
- **(e)** Repeat parts (a)–(d) using the viscosity response.

6.32. *Continuation of Problem 6.31.* Use the regression models for molecular weight and viscosity to answer the following questions.

- **(a)** Construct a response surface contour plot for molecular weight. In what direction would you adjust the process variables to increase molecular weight?
- **(b)** Construct a response surface contour plot for viscosity. In what direction would you adjust the process variables to decrease viscosity?
- **(c)** What operating conditions would you recommend if it was necessary to produce a product with molecular weight between 2400 and 2500 and the lowest possible viscosity?

Don’t use center points, skip the question about curvature term (will be covered in the next lesson).

Today Exercise



■ **TABLE P6.8**

The 2^4 Experiment for Problem 6.31

Run Number	Actual Run Order	*A*	*B*	*C*	*D*	Molecular Weight	Viscosity	Factor Levels		
									Low (−)	High (+)
1	18	−	−	−	−	2400	1400	*A* (°C)	100	120
2	9	+	−	−	−	2410	1500	*B* (%)	4	8
3	13	−	+	−	−	2315	1520	*C* (min)	20	30
4	8	+	+	−	−	2510	1630	*D* (psi)	60	75
5	3	−	−	+	−	2615	1380			
6	11	+	−	+	−	2625	1525			
7	14	−	+	+	−	2400	1500			
8	17	+	+	+	−	2750	1620			
9	6	−	−	−	+	2400	1400			
10	7	+	−	−	+	2390	1525			
11	2	−	+	−	+	2300	1500			
12	10	+	+	−	+	2520	1500			
13	4	−	−	+	+	2625	1420			
14	19	+	−	+	+	2630	1490			
15	15	−	+	+	+	2500	1500			
16	20	+	+	+	+	2710	1600			
17	1	0	0	0	0	2515	1500			
18	5	0	0	0	0	2500	1460			
19	16	0	0	0	0	2400	1525			
20	12	0	0	0	0	2475	1500			