FTC Short Course - Design and Analysis of Experiments with R

John Lawson

Department of Statistics - BYU

today

Outline of Part I

Outline of Part II

2. A Context for Discussing Experimental Designs
   - Introduction
   - Preliminary Exploration
   - Screening Factors
   - Effect Estimation
   - Optimization
   - Sequential Experimentation

Outline of Part III

3. Design and Analysis of Two-Level Factorials
   - Two-Level Factorials
   - The Justification for Two-Levels
   - Creating and Analyzing Two-Level Factorials with R
   - Blocking Two-Level Factorials
   - Restrictions on Randomization - Split-Plot Designs

Outline of Part IV

4. Preliminary Exploration
   - Introduction
   - One-Factor Designs
   - Two-Factor Designs
   - Staggered Nested Designs for Multiple Factors
   - Graphical Methods to Check Assumptions
   - Chemistry Example

Outline of Part V

5. Design and Analysis of Screening Experiments
   - Introduction
   - Half-Fractions of Two-Level Factorial Designs
   - One-Quarter and Higher Fractions of Two-Level Factorial Designs
   - Criteria for Choosing Generators for Fractional Factorial Designs
   - Augmenting Fractional Factorial Designs to Resolve Confounding
   - Plackett-Burman and Model Robust Screening Designs

Outline of Part VI

Part I

R - An Environment for Data Analysis and Graphics

Outline of Part I

A Short Description of R

- R is the language of choice for a large and growing proportion of people developing new statistical algorithms
- R is available under GNU General Public License for Windows, Mac OS X, and Linux
- R is extendable with user submitted packages
- The Comprehensive R Archive Network (CRAN) makes it easy to benefit from others work, and share your own work and get feedback for improvements
- There are many user written packages available for the Design and Analysis of Experiments

Websites for Help Getting Started with R

- The R Project for Statistical Computing
  https://www.r-project.org
- Getting Started with R
  http://data.princeton.edu/R/
- A Short Tutorial
  http://math.usask.ca/~longhai/doc/others/R-tutorial.pdf
- An Introductory pdf Manual can be Obtained Here
  https://cran.r-project.org/doc/manuals/R-intro.pdf

Websites for Help Getting Started with R

- Installing and using R packages
  http://math.usask.ca/~longhai/software/installrpkg.html
- R Packages for Design an Analysis of Experiments
  https://cran.r-project.org/web/views/ExperimentalDesign.html

Objects in R

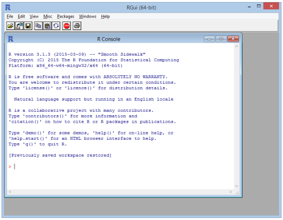
During an R session R Creates Entities known as Objects

- Variables
- Arrays of numbers
- Character strings
- Functions
- Data frames and other more complex elements built from earlier components

The R Console

Command line prompt >

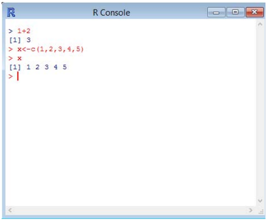
Type commands and see text results immediately



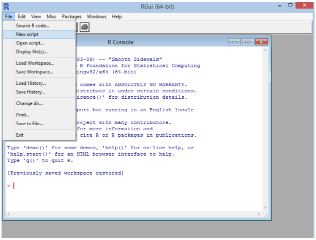
Command line Examples

Expressions and Assignments

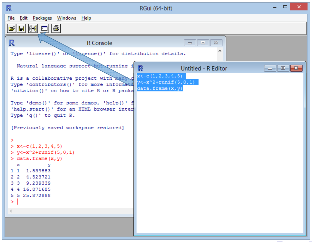
Do calculations or make assignments



The R Script

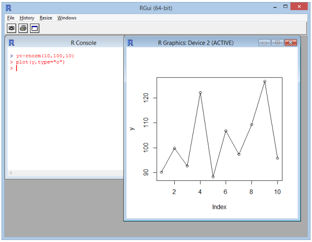


Running Commands from an RScript



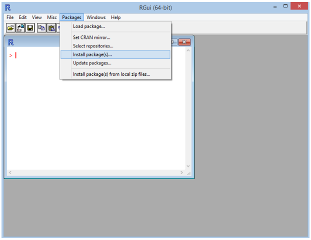
R Basics
Preliminaries
Program Interface
R packages
Code and Data from The Book

Making a Plot in R

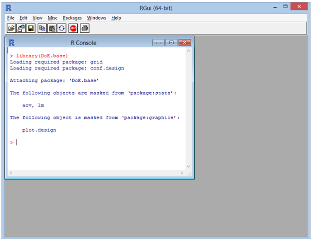


John Lawson
FTC Short Course - Design and Analysis of Experiments with R

Installing an R Package



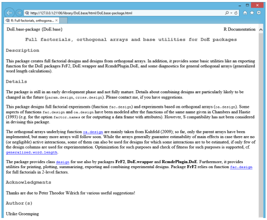
Loading an R Package



Documentation for an R Package

Package Documents

Document functions and data frames available in the package



http://127.0.0.1:21106/library/DoE.base/html/DoE.base-package.html

R: Full factorials, orthogona...

DoE.base-package {DoE.base} R Documentation

Full factorials, orthogonal arrays and base utilities for DoE packages

Description

This package creates full factorial designs and designs from orthogonal arrays. In addition, it provides some basic utilities like an exporting function for the DoE packages FrF2, DoE.wrapper and RcmdrPlugin.DoE, and some diagnostics for general orthogonal arrays (generalized word length calculations).

Details

The package is still in an early development phase and not fully mature. Details about combining designs are particularly likely to be changed in the future (`param.design`, `cross.design`). Please contact me, if you have suggestions.

This package designs full factorial experiments (function `fac.design`) and experiments based on orthogonal arrays (`oa.design`). Some aspects of functions `fac.design` and `oa.design` have been modeled after the functions of the same name given in Chambers and Hastie (1993) (e.g. for the option `factor.names` or for outputting a data frame with attributes). However, S compatibility has not been considered in devising this package.

The orthogonal arrays underlying function `oa.design` are mainly taken from Kuhfeld (2009); so far, only the parent arrays have been implemented, but many more arrays will follow soon. While the arrays generally guarantee estimability of main effects in case there are no (or negligible) active interactions, some of them can also be used for designs for which some interactions are to be estimated, if only few of the design columns are used for experimentation. Optimization for such purposes and check of fitness for such purposes is supported, cf. `generalized.word.length`.

The package provides class `design` for use also by packages **FrF2**, **DoE.wrapper** and **RcmdrPlugin.DoE**. Furthermore, it provides utilities for printing, plotting, summarizing, exporting and combining experimental designs. Package **FrF2** relies on function `fac.design` for full factorials in 2-level factors.

Acknowledgments

Thanks are due to Peter Theodor Wilrich for various useful suggestions!

Author(s)

Ulrike Groemping

Documentation for a Function

Function Document

fact.design function in DoE.Base Package

`fac.design`	*Function for full factorial designs*

Description

Function for creating full factorial designs with arbitrary numbers of levels, and potentially with blocking

Usage

```
fac.design(nlevels=NULL, nfactors=NULL, factor.names = NULL,
        replications=1, repeat.only = FALSE, randomize=TRUE, seed=NULL,
        blocks=1, block.gen=NULL, block.name="Blocks", bbreps=replications,
        wbreps=1, block.old.behavior=FALSE)
```

Arguments

`nlevels` number(s) of levels, vector with `nfactors` entries or single number; can be omitted, if obvious from `factor.names`

`nfactors` number of factors, can be omitted if obvious from entries `nlevels` or `factor.names`

Example Code in Function Documentation

Function Examples

Examples of fact.design function

Examples

```
## only specify level combination
fac.design(nlevels=c(4,3,3,2))
## design requested via factor.names
fac.design(factor.names=list(one=c("a","b","c"), two=c(125,275),
   three=c("old","new"), four=c(-1,1), five=c("min","medium","max")))
## design requested via character factor.names and nlevels
##    (with a little German lesson for one two three)
fac.design(factor.names=c("eins","zwei","drei"),nlevels=c(2,3,2))

### blocking designs
fac.design(nlevels=c(2,2,3,3,6), blocks=6, seed=12345)
## the same design, now unnecessarily constructed via option block.gen
## preparation: look at the numbers of levels of pseudo factors
## (in this order)
unlist(factorize(c(2,2,3,3,6)))
## or, for more annotation, factorize the unblocked design
factorize(fac.design(nlevels=c(2,2,3,3,6)))
## positions 1 2 5 are 2-level pseudo factors
## positions 3 4 6 are 4-level pseudo factors
## blocking with highest possible interactions
G <- rbind(two=c(1,1,0,0,1,0),three=c(0,0,1,1,0,1))
plan.6blocks <- fac.design(nlevels=c(2,2,3,3,6), blocks=6, block.gen=G, seed=12345)
plan.6blocks
```


Running a function in a loaded package (DoE.Base)

R Console

```
> fac.design(factor.names=list(A=c(10,20),B=c("old","new"),C=c("min","med","max")))
creating full factorial with  12  runs ...
    A   B   C
1  10 new med
2  20 old med
3  10 new min
4  20 new med
5  20 new max
6  10 old max
7  20 new min
8  20 old min
9  10 old med
10 10 old min
11 10 new max
12 20 old max
class=design, type= full factorial
> |
```

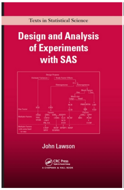
User written R packages illustrated in the book

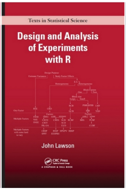
```
AlgDesign, agricolae
BsMD
car crossdes
daewr, DoE.base
effects
FrF2
GAD, gdata, gmodels
leaps, lme4
mixexp, multcomp
nlme
rsm
```

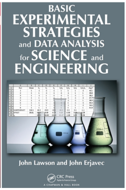

Website for the book

https://jlawson.byu.edu

Design and Analysis of Experiment Books written by Dr John Lawson



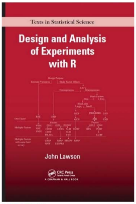




Code examples in the book

Code and Data

Code: Web page
Data: daewr package



R Code Examples
Errata for Design and Analysis of Experiments with R
R Commander Example

R Code Examples

- R Examples for Chapter 2
- R Examples for Chapter 3
- R Examples for Chapter 4
- R Examples for Chapter 5
- R Examples for Chapter 6
- R Examples for Chapter 7
- R Examples for Chapter 8
- R Examples for Chapter 9
- R Examples for Chapter 10
- R Examples for Chapter 11
- R Examples for Chapter 12
- R Examples for Chapter 13

R Basics
Preliminaries
Program Interface
R packages
Code and Data from The Book

R Code for Chapter 2

R Examples for Chapter 2

https://jlawson.byu.edu/RBOOK/RCode/Chapter2.R

```
# Example 1 p. 18
set.seed(7638)
f <- factor( rep ( c(35, 40, 45 ), each = 4))
fac <- sample ( f, 12 )
eu <- 1:12
plan <- data.frame ( loaf = eu, time = fac )
write.csv( plan, file = "Plan.csv", row.names = FALSE )

# Example 2 p. 23
bread <- read.csv("Plan.csv")

# Example 3 p. 24
rm(bread)
library(daewr)
mod0 <-lm( height ~ time, data = bread )
summary (mod0)

# Example 4 p. 25
library(gmodels)
fit.contrast (mod0, "time", c(1, -1, 0) )
```

Part II

A Context for Discussing Experimental Designs

Outline of Part II

Strategy for Experimentation

Present ⇓ Goal ⇓

0% **Knowledge** 100%

Objective:	Preliminary Exploration	Screening Factors	Effect Estimation	Optimization	Mechanistic Modeling
No. of Factors		5 - 20	3 - 6	2 - 4	1 - 5
Purpose:	Identify Sources of Variability	Identify Important Factors	Estimate Factor Effects + Interactions	Fit Empirical Model Interpolate	Estimate Parameters of Theory Extrapolate

R.D. Snee "Raise Your Batting Average" *Quality Progress* Dec. 2009

Preliminary Exploration

- Exploratory experiments to study repeatability of the process
- Identify process steps causing majority of the variability in results
- Identify factors that possibly affect the results

Screening

- Explores a large number of factors
- Objective is to identify smaller subset of most important factors
- Fit linear models to the data

Effect Estimation

- Explores the relationship between results and important factors
- Goal is to estimate linear effects and interactions and develop a prediction model
- Fit models including linear effects and interactions

Optimization

- Explores the relationship between results and a limited number of quantitative leveled factors
- Goal is to identify optimum operating conditions within the factor ranges studied
- Fit quadratic response surface models

Sequential Experimentation

- Plan Ahead – decide on a series of experiments that may be needed
- Consider All Possible Factors – majority of variation is caused by a subset of factors, but which ones?
- Don't Spend All Resources on a Single Experiment

Possible Sequences

- Preliminary Exploration – Effect Estimation
- Preliminary Exploration – Optimization
- Screening – Effect Estimation – Optimization

Part III

Design and Analysis of Two-Level Factorials

Outline of Part III

3. Design and Analysis of Two-Level Factorials
   - Two-Level Factorials
   - The Justification for Two-Levels
   - Creating and Analyzing Two-Level Factorials with R
   - Blocking Two-Level Factorials
   - Restrictions on Randomization - Split-Plot Designs

Why start discussion with two-level factorials?

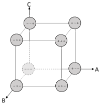
Present ⇓ → Goal ⇓

0% — **Knowledge** — 100%

Objective:	Preliminary Exploration	Screening Factors	Effect Estimation	Optimization	Mechanistic Modeling
No. of Factors		5 - 20	3 - 6	2 - 4	1 - 5
Purpose:	Identify Sources of Variability	Identify Important Factors	Estimate Factor Effects + Interactions	Fit Empirical Model Interpolate	Estimate Parameters of Theory Extrapolate

Why start discussion with two-level factorials?

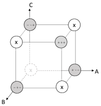
Screening Factors	Effect Estimation	Optimization

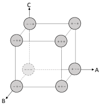


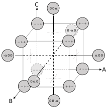
Two-Level Factorial

Why start discussion with two-level factorials?

Screening Factors	Effect Estimation	Optimization





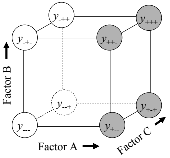


Two-Level Factorial

Effect estimation in two-level factorials

Figure 3.10 *Geometric Representation of* 2^3 *Design and Main Effect Calculation*

Run	X_A	X_B	X_C	Response
1	-	-	-	y_{---}
2	+	-	-	y_{+--}
3	-	+	-	y_{-+-}
4	+	+	-	y_{++-}
5	-	-	+	y_{--+}
6	+	-	+	y_{+-+}
7	-	+	+	y_{-++}
8	+	+	+	y_{+++}



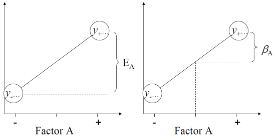
$$E_A = (y_{+--} + y_{++-} + y_{+-+} + y_{+++})/4 - (y_{---} + y_{-+-} + y_{--+} + y_{-++})/4$$

Relation between effect and regression coefficient

Figure 3.9 *Effect and Regression Coefficient for Two-Level Factorial*

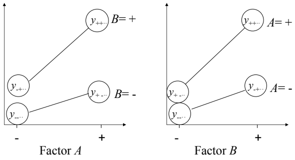
Coded Factor Levels
for factors with
quantitative levels

$$X_A = \frac{(\text{factor setting} - \text{mid setting})}{(\text{high setting} - \text{low setting})/2}$$



Definition of interaction effect

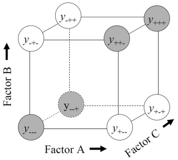
Figure 3.11 *Definition of an Interaction Effect for Two-Level Factorial*



Calculation of interaction effect

Figure 3.12 *Geometric Representation of 2^3 Design and Interaction Effect*

Run	X_A	X_B	X_C	$X_A{\cdot}X_B$	Resp.
1	-	-	-	+	y_{---}
2	+	-	-	-	y_{+--}
3	-	+	-	-	y_{-+-}
4	+	+	-	+	y_{++-}
5	-	-	+	+	y_{--+}
6	+	-	+	-	y_{+-+}
7	-	+	+	-	y_{-++}
8	+	+	+	+	y_{+++}



$$E_{AB}=(y_{---}+y_{++-}+y_{--+}+y_{+++})/4-(y_{+--}+y_{-+-}+y_{+-+}+y_{-++})/4$$

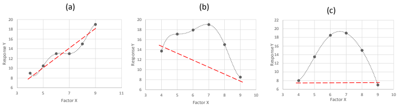
Number of experiments

Number of Experiments Required for a Full-Factorial

	Number of Levels		
Number of Factors	2	3	4
2	4	9	16
3	8	27	64
4	16	81	256
5	32	243	1024

Choice of Levels

- Factors with Qualitative Levels
- Factors with Quantitative Levels



Creating a two-level factorial design with R FrF2

Problem 9 Chapter 3 of "Design and Analysis with R"

9. Nyberg (1999) has shown that silicon nitride (SiNx) grown by Plasma Enhanced Chemical Vapor Deposition (PECVD) is a promising candidate for an antireflection coating (ARC) on commercial crystalline silicon solar cells. Silicon nitride was grown on polished (100)-oriented 4A silicon wafers using a parallel plate Plasma Technology PECVD reactor. The diameter of the electrodes of the PECVD is 24 cm and the diameter of the shower head (through which the gases enter) is 2A. The RF frequency was 13.56 MHz. The thickness of the silicon nitride was one-quarter of the wavelength of light in the nitride, the wavelength being 640 nm. This wavelength is expected to be close to optimal for silicon solar cell purposes. The process gases were ammonia and a mixture of 3% silane in argon. The experiments were carried out according to a 2^5 factorial design. The results are shown in the table on the next page.

Design and Analysis of Two-Level Factorials
Two-Level Factorials
The Justification for Two-Levels
Creating and Analyzing Two-Level Factorials with R
Blocking Two-Level Factorials
Restrictions on Randomization - Split-Plot Designs

Creating a two-level factorial design with R FrF2

Exercise data

Exp. No.	A Silane to Ammonia Flow Rate Ratio	B Total Gas Flow Rate (sccm)	C Press. (mtorr)	D Temp. (C°)	E Power (W)	y_1 Refract. Index	y_2 Growth Rate (nm/min)
1	0.1	40	300	300	10	1.92	1.79
2	0.9	40	300	300	10	3.06	10.1
3	0.1	220	300	300	10	1.96	3.02
4	0.9	220	300	300	10	3.33	15
5	0.1	40	1200	300	10	1.87	19.7
6	0.9	40	1200	300	10	2.62	11.2
7	0.1	220	1200	300	10	1.97	35.7
8	0.9	220	1200	300	10	2.96	36.2
9	0.1	40	300	460	10	1.94	2.31
10	0.9	40	300	460	10	3.53	5.58
11	0.1	220	300	460	10	2.06	2.75
12	0.9	220	300	460	10	3.75	14.5
13	0.1	40	1200	460	10	1.96	20.7
14	0.9	40	1200	460	10	3.14	11.7
15	0.1	220	1200	460	10	2.15	31
16	0.9	220	1200	460	10	3.43	39
17	0.1	40	300	300	60	1.95	3.93
18	0.9	40	300	300	60	3.16	12.4
19	0.1	220	300	300	60	2.01	6.33
20	0.9	220	300	300	60	3.43	23.7
21	0.1	40	1200	300	60	1.88	35.3
22	0.9	40	1200	300	60	2.14	15.1
23	0.1	220	1200	300	60	1.98	57.1
24	0.9	220	1200	300	60	2.81	45.9
25	0.1	40	300	460	60	1.97	5.27
26	0.9	40	300	460	60	3.67	12.3
27	0.1	220	300	460	60	2.09	6.39
28	0.9	220	300	460	60	3.73	30.5
29	0.1	40	1200	460	60	1.98	30.1
30	0.9	40	1200	460	60	2.99	14.5
31	0.1	220	1200	460	60	2.19	50.3
32	0.9	220	1200	460	60	3.39	47.1

Creating a two-level factorial design with R FrF2

```
> library(FrF2)
> Design.p9 <-FrF2(nruns=32, nfactors=5, blocks=1, ncenter=0, replications=1,
+ randomize=FALSE, factor.names=list(Ratio=c(0.1,0.9),Gas_flow=c(40,60),
+ Pressure=c(300,1200),Temperature=c(300,460), Power=c(10,60)))
creating full factorial with 32 runs ...

> y1<-c(1.92,3.06,1.96,3.33,1.87,2.62,1.97,2.96,1.94,3.53,2.06,3.75,1.96,3.14,2.15,
+ 3.43,1.95,3.16,2.01,3.43,1.88,2.14,1.98,2.81,1.97,3.67,2.09,3.73,1.98,2.99,2.19,
+ 3.39)
> y2<-c(1.79,10.10,3.02,15.00,19.70,11.20,35.70,36.20,2.31,5.58,2.75,14.50,20.70,
+ 11.70,31.00,39.00,3.93,12.40,6.33,23.70,35.30,15.10,57.10,45.90,5.27,12.30,6.39,
+ 30.50,30.10,14.50,50.30,47.10)
> Design.p9 <- add.response(Design.p9, y1, replace=FALSE)
> Design.p9 <- add.response(Design.p9, y2, replace=FALSE)
```

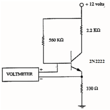

Design and Analysis of Two-Level Factorials
Two-Level Factorials
The Justification for Two-Levels
Creating and Analyzing Two-Level Factorials with R
Blocking Two-Level Factorials
Restrictions on Randomization - Split-Plot Designs

Creating a two-level factorial design with R FrF2

```
> print( Design.p9, std.order=TRUE)
   run.no.in.std.order run.no Ratio Gas_flow Pressure Temperature Power   y1    y2
1                    1      1   0.1       40      300         300    10 1.92  1.79
2                    2      2   0.9       40      300         300    10 3.06 10.10
3                    3      3   0.1       60      300         300    10 1.96  3.02
4                    4      4   0.9       60      300         300    10 3.33 15.00
5                    5      5   0.1       40     1200         300    10 1.87 19.70
6                    6      6   0.9       40     1200         300    10 2.62 11.20
7                    7      7   0.1       60     1200         300    10 1.97 35.70
8                    8      8   0.9       60     1200         300    10 2.96 36.20
9                    9      9   0.1       40      300         460    10 1.94  2.31
10                  10     10   0.9       40      300         460    10 3.53  5.58
11                  11     11   0.1       60      300         460    10 2.06  2.75
12                  12     12   0.9       60      300         460    10 3.75 14.50
13                  13     13   0.1       40     1200         460    10 1.96 20.70
14                  14     14   0.9       40     1200         460    10 3.14 11.70
15                  15     15   0.1       60     1200         460    10 2.15 31.00
16                  16     16   0.9       60     1200         460    10 3.43 39.00
17                  17     17   0.1       40      300         300    60 1.95  3.93
18                  18     18   0.9       40      300         300    60 3.16 12.40
19                  19     19   0.1       60      300         300    60 2.01  6.33
20                  20     20   0.9       60      300         300    60 3.43 23.70
21                  21     21   0.1       40     1200         300    60 1.88 35.30
22                  22     22   0.9       40     1200         300    60 2.14 15.10
                        ...
30                  30     30   0.9       40     1200         460    60 2.99 14.50
31                  31     31   0.1       60     1200         460    60 2.19 50.30
32                  32     32   0.9       60     1200         460    60 3.39 47.10
NOTE: columns run.no.in.std.order and run.no are annotation, not part of the data frame
>
```

Example analysis of a replicated 2^3 factorial

Factor	Levels −	Levels +
A=Ambient temperature, °C	22	32
B=Voltmeter warmup time, minutes	0.5	5.0
C=Time power is connected, minutes	0.5	5.0
Y=measured voltage, millivolts		



Example analysis of a replicated 2^3 factorial

Table 3.6 *Factor Settings and Response for Voltmeter Experiment*

	Factor Levels			Coded Factors					
Run	A	B	C	X_A	X_B	X_C	Rep	Order	y
1	22	0.5	0.5	−	−	−	1	5	705
2	32	0.5	0.5	+	−	−	1	14	620
3	22	5.0	0.5	−	+	−	1	15	700
4	32	5.0	0.5	+	+	−	1	1	629
5	22	0.5	5.0	−	−	+	1	8	672
6	32	0.5	5.0	+	−	+	1	12	668
7	22	5.0	5.0	−	+	+	1	10	715
8	32	5.0	5.0	+	+	+	1	9	647
1	22	0.5	0.5	−	−	−	1	4	680
2	32	0.5	0.5	+	−	−	1	7	651
3	22	5.0	0.5	−	+	−	1	2	685
4	32	5.0	0.5	+	+	−	1	3	635
5	22	0.5	5.0	−	−	+	1	11	654
6	32	0.5	5.0	+	−	+	1	16	691
7	22	5.0	5.0	−	+	+	1	6	672
8	32	5.0	5.0	+	+	+	1	13	673

Example analysis of a replicated 2^3 factorial

Note

volt is a data frame in daewr package

```
> library(daewr)
Warning message:
package ‘daewr’ was built under R version
3.2.2
> volt
    A   B   C   y
1  22 0.5 0.5 705
2  32 0.5 0.5 620
3  22   5 0.5 700
4  32   5 0.5 629
5  22 0.5   5 672
6  32 0.5   5 668
7  22   5   5 715
8  32   5   5 647
9  22 0.5 0.5 680
10 32 0.5 0.5 651
11 22   5 0.5 685
12 32   5 0.5 635
13 22 0.5   5 654
14 32 0.5   5 691
15 22   5   5 672
16 32   5   5 673
> class(volt$A)
[1] "factor"
```


Example analysis of a replicated 2^3 factorial

Code was cut and pasted from
R examples for Chapter 2

https://jlawson.byu.edu/RBOOK/REXamples.html

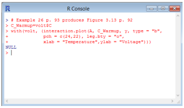
the statement
`contrast=list(A=contr.FrF2,…`

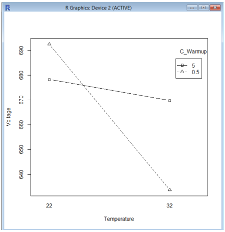
Converts actual factor levels for A stored as factors in data frame `volt` to coded factor level contrasts `A1` etc. This would not be necessary if the design was Created by R package FrF2

The estimates are the regression coefficients or ½ of the Effects.

```
> library(FrF2)
> modv<-lm(y ~ A*B*C, data=volt, contrast=list(A=contr.FrF2,
+ B=contr.FrF2, C=contr.FrF2))
> summary(modv)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 668.5625     4.5178 147.985 4.86e-15 ***
A1          -16.8125     4.5178  -3.721  0.00586 **
B1            0.9375     4.5178   0.208  0.84079
C1            5.4375     4.5178   1.204  0.26315
A1:B1        -6.6875     4.5178  -1.480  0.17707
A1:C1        12.5625     4.5178   2.781  0.02390 *
B1:C1         1.8125     4.5178   0.401  0.69878
A1:B1:C1     -5.8125     4.5178  -1.287  0.23422
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '

Residual standard error: 18.07 on 8 degrees of freedom
Multiple R-squared:  0.772,     Adjusted R-squared:  0.5724
F-statistic: 3.869 on 7 and 8 DF,  p-value: 0.0385
```

Example analysis of a replicated 2^3 factorial





Design and Analysis of Two-Level Factorials
Two-Level Factorials
The Justification for Two-Levels
Creating and Analyzing Two-Level Factorials with R
Blocking Two-Level Factorials
Restrictions on Randomization - Split-Plot Designs

Example analysis of a replicated 2^3 factorial

Note

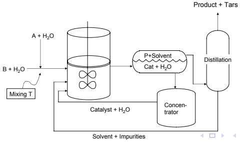
Since the design is orthogonal insignificant terms dropped without refitting to get a prediction equation

$$y = 668.56 - 16.81\left(\frac{Temp-27}{5}\right) + 6.27\left(\frac{CWarm-2.75}{2.25}\right)\left(\frac{Temp-27}{5}\right)$$

Example analysis of an unreplicated 2^4 design

Symbol	Factor Name
A	Excess of Reactant A (over molar amount)
B	Catalyst Concentration
C	Pressure in the Reactor
D	Temperature of the Coated Mixing-T

Figure 3.14 *Diagram of a Chemical Process*

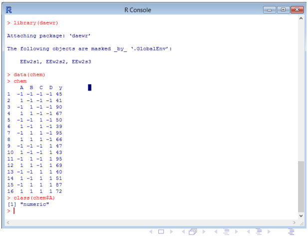


Design and Analysis of Two-Level Factorials
Two-Level Factorials
The Justification for Two-Levels
Creating and Analyzing Two-Level Factorials with R
Blocking Two-Level Factorials
Restrictions on Randomization - Split-Plot Designs

Example analysis of an unreplicated 2^4 design

Note

chem is a data frame in daewr package



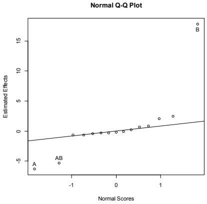

```
> library(daewr)

Attaching package: 'daewr'

The following objects are masked _by_ '.GlobalEnv':

    EEw2s1, EEw2s2, EEw2s3

> data(chem)
> chem
    A  B  C  D  y
1  -1 -1 -1 -1 45
2   1 -1 -1 -1 41
3  -1  1 -1 -1 90
4   1  1 -1 -1 67
5  -1 -1  1 -1 50
6   1 -1  1 -1 39
7  -1  1  1 -1 95
8   1  1  1 -1 66
9  -1 -1 -1  1 47
10  1 -1 -1  1 43
11 -1  1 -1  1 95
12  1  1 -1  1 69
13 -1 -1  1  1 40
14  1 -1  1  1 51
15 -1  1  1  1 87
16  1  1  1  1 72
> class(chem$A)
[1] "numeric"
>
```

Example analysis of an unreplicated 2^4 design

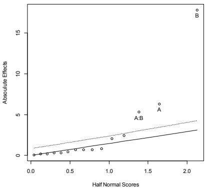
```
> modf <-lm( y ~ A*B*C*D, data = chem)
> summary(modf)

Call:
lm(formula = y ~ A * B * C * D, data = chem)

Residuals:
ALL 16 residuals are 0: no residual degrees of freedom!

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  62.3125         NA      NA       NA
A            -6.3125         NA      NA       NA
B            17.8125         NA      NA       NA
C             0.1875         NA      NA       NA
D             0.6875         NA      NA       NA
A:B          -5.3125         NA      NA       NA
A:C           0.8125         NA      NA       NA
B:C          -0.3125         NA      NA       NA
A:D           2.0625         NA      NA       NA
B:D          -0.0625         NA      NA       NA
C:D          -0.6875         NA      NA       NA
A:B:C        -0.1875         NA      NA       NA
A:B:D        -0.6875         NA      NA       NA
A:C:D         2.4375         NA      NA       NA
B:C:D        -0.4375         NA      NA       NA
A:B:C:D      -0.3125         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
```


Example analysis of an unreplicated 2^4 design

```
> fullnormal(coef(modf)[-1],alpha=.025)
```

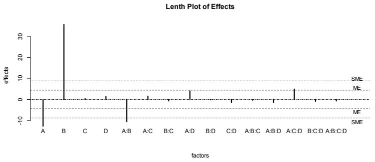



```
> LGB( coef(modf)[-1], rpt = FALSE)
```



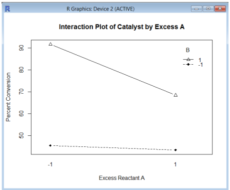
Example analysis of an unreplicated 2^4 design

```
> LenthPlot(modf, main = "Lenth Plot of Effects")
```



Example analysis of an unreplicated 2^4 design

```
> with(chem, (interaction.plot( A, B, y, type = "b", pch = c(18,24),
          main = "Interaction Plot of Catalyst by Excess A",
          xlab = "Excess Reactant A", ylab = "Percent Conversion")))
```



B= Catalyst concentration

Example analysis of an unreplicated design with an outlier

$$E_i = \left(\left(\sum_{\{X_i=+\}} Y_i \right) - \left(\sum_{\{X_i=-\}} Y_i \right) \right) \Big/ \left(\frac{n}{2} \right)$$

Daniel (1960) proposed a manual method for detecting and correcting an outlier or atypical value in an unreplicated 2^k design. This method consists of three steps. First, the presence of an outlier is detected by a gap in the center of a normal plot of effects. Second, the outlier is identified by matching the signs of the insignificant effects with the signs of the coded factor levels and interactions of each observation. The third step is to estimate the magnitude of the discrepancy and correct the atypical value.

Example analysis of an unreplicated design with an outlier

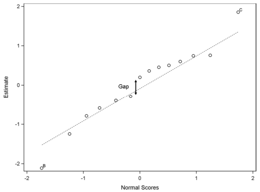
Note

BoxM is a data frame in daewr package taken from Box(1991)

```
> library(daewr)
> data(BoxM)
> BoxM
    A  B  C  D     y
1  -1 -1 -1 -1 47.46
2   1 -1 -1 -1 49.62
3  -1  1 -1 -1 43.13
4   1  1 -1 -1 46.31
5  -1 -1  1 -1 51.47
6   1 -1  1 -1 48.49
7  -1  1  1 -1 49.34
8   1  1  1 -1 46.10
9  -1 -1 -1  1 46.76
10  1 -1 -1  1 48.56
11 -1  1 -1  1 44.83
12  1  1 -1  1 44.45
13 -1 -1  1  1 59.15
14  1 -1  1  1 51.33
15 -1  1  1  1 47.02
16  1  1  1  1 47.90
```

Example analysis of an unreplicated design with an outlier

```
> fullnormal(coef(modB)[-1],alpha=.2)
```



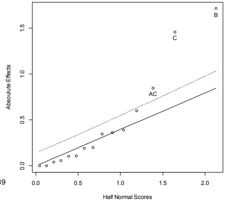
Design and Analysis of Two-Level Factorials
Two-Level Factorials
The Justification for Two-Levels
Creating and Analyzing Two-Level Factorials with R
Blocking Two-Level Factorials
Restrictions on Randomization - Split-Plot Designs

Example analysis of an unreplicated design with an outlier

```
> Gaptest(BoxM)
Effect Report
                                                  Corrrected Data Report
Label     Half Effect     Sig(.05)                Response  Corrected Response    Detect Outlier
A           -0.400        no                        47.46        47.46               no
B           -2.110        no                        49.62        49.62               no
C            1.855        no                        43.13        43.13               no
D            0.505        no                        46.31        46.31               no
AB           0.455        no                        51.47        51.47               no
AC          -1.245        no                        48.49        48.49               no
AD          -0.290        no                        49.34        49.34               no
BC          -0.400        no                        46.10        46.10               no
BD          -0.590        no                        46.76        46.76               no
CD           0.745        no                        48.56        48.56               no
ABC          0.600        no                        44.83        44.83               no
ABD          0.360        no                        44.45        44.45               no
ACD          0.200        no                        59.15        52.75               yes
BCD         -0.790        no                        51.33        51.33               no
ABCD         0.760        no                        47.02        47.02               no
                                                    47.90        47.90               no
Lawson, Grimshaw & Burt Rn Statistic =  1
95th percentile of Rn =  1.201
Initial Outlier Report
Standardized-Gap =  3.353227 Significant at 50th percentile
Final Outlier Report
Standardized-Gap =  13.18936 Significant at 99th percentile
```

Example analysis of an unreplicated design with an outlier

```
Effect Report

Label     Half Effect    Sig(.05)
A        -4.514306e-15     no
B        -1.710000e+00     yes
C         1.455000e+00     yes
D         1.050000e-01     no
AB        5.500000e-02     no
AC       -8.450000e-01     yes
AD        1.100000e-01     no
BC        2.170070e-15     no
BD       -1.900000e-01     no
CD        3.450000e-01     no
ABC       2.000000e-01     no
ABD      -4.000000e-02     no
ACD       6.000000e-01     no
BCD      -3.900000e-01     no
ABCD      3.600000e-01     no

Lawson, Grimshaw & Burt Rn Statistic =   1.626089
95th percentile of Rn =   1.201
```



Design and Analysis of Two-Level Factorials
Two-Level Factorials
The Justification for Two-Levels
Creating and Analyzing Two-Level Factorials with R

Restrictions on Randomization - Split-Plot Designs

Blocking a 2^4

Dish Soaking Experiment

Experimental Unit:

Response: Number of Clean grid squares

Factors:

A=Water Temperature

B=Soap Amount

C=Soap Brand

D=Soaking Time

Table 7.4 *Factors for Dishwashing Experiment*

	Levels	
Factor	(−)	(+)
A-Water Temperature	60 Deg F	115 Deg F
B-Soap Amount	1 tbs	2tbs
C-Soaking Time	3 min	5 min
D-Soap Brand	WF	UP

John Lawson
FTC Short Course - Design and Analysis of Experiments with R

Blocking a 2^4

Blocking factor:

Block 1 = W.F., 1:30
Block 2 = W.F., 1:00
Block 3 = Prego, 1:30
Block 4 = Prego, 1:00

4 E.U's per block

Confound AC, ABD

AC(ABD)=BCD gets confounded

Table 7.5 *Blocks for Dishwashing Experiment*

Block	Type Sauce	Microwave Time
1	Store Brand	1 min
2	Premium Brand	1 min
3	Store Brand	1:30 min
4	Premium Brand	1:30 min

Design and Analysis of Two-Level Factorials
Two-Level Factorials
The Justification for Two-Levels
Creating and Analyzing Two-Level Factorials with R
Blocking Two-Level Factorials
Restrictions on Randomization - Split-Plot Designs

Create the design with FrF2

```
> library(FrF2)
> Bdish <- FrF2(16, 4, blocks=c("ABD", "BCD"), alias.block.2fis=TRUE, randomize=FALSE)
> Bdish
  run.no run.no.std.rp Blocks  A  B  C  D
1      1         1.1.1      1 -1 -1 -1 -1
2      2         6.1.2      1 -1  1 -1  1
3      3        12.1.3      1  1 -1  1  1
4      4        15.1.4      1  1  1  1 -1
  run.no run.no.std.rp Blocks  A  B  C  D
5      5         3.2.1      2 -1 -1  1 -1
6      6         8.2.2      2 -1  1  1  1
7      7        10.2.3      2  1 -1 -1  1
8      8        13.2.4      2  1  1 -1 -1
   run.no run.no.std.rp Blocks  A  B  C  D
9       9         4.3.1      3 -1 -1  1  1
10     10         7.3.2      3 -1  1  1 -1
11     11         9.3.3      3  1 -1 -1 -1
12     12        14.3.4      3  1  1 -1  1
   run.no run.no.std.rp Blocks  A  B  C  D
13     13         2.4.1      4 -1 -1 -1  1
14     14         5.4.2      4 -1  1 -1 -1
15     15        11.4.3      4  1 -1  1 -1
16     16        16.4.4      4  1  1  1  1
class=design, type= FrF2.blocked
NOTE: columns run.no and run.no.std.rp are annotation, not part of the data frame
```

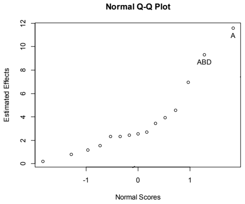
Create the design with FrF2

```
> y<-c(0, 0, 12, 14, 1, 0, 1, 11, 10, 2, 33, 24, 3, 5, 41, 70)
> Bdish<-add.response(Bdish, response=y)
> Bdish
  run.no run.no.std.rp Blocks  A  B  C  D  y
1      1         1.1.1      1 -1 -1 -1 -1  0
2      2         6.1.2      1 -1  1 -1  1  0
3      3        12.1.3      1  1 -1  1  1 12
4      4        15.1.4      1  1  1  1 -1 14
  run.no run.no.std.rp Blocks  A  B  C  D  y
5      5         3.2.1      2 -1 -1  1 -1  1
6      6         8.2.2      2 -1  1  1  1  0
7      7        10.2.3      2  1 -1 -1  1  1
8      8        13.2.4      2  1  1 -1 -1 11
   run.no run.no.std.rp Blocks  A  B  C  D  y
9       9         4.3.1      3 -1 -1  1  1 10
10     10         7.3.2      3 -1  1  1 -1  2
11     11         9.3.3      3  1 -1 -1 -1 33
12     12        14.3.4      3  1  1 -1  1 24
   run.no run.no.std.rp Blocks  A  B  C  D  y
13     13         2.4.1      4 -1 -1 -1  1  3
14     14         5.4.2      4 -1  1 -1 -1  5
15     15        11.4.3      4  1 -1  1 -1 41
16     16        16.4.4      4  1  1  1  1 70
class=design, type= FrF2.blocked
NOTE: columns run.no and run.no.std.rp are annotation, not part of
the data frame
```


Design and Analysis of Two-Level Factorials
Two-Level Factorials
The Justification for Two-Levels
Creating and Analyzing Two-Level Factorials with R
Blocking Two-Level Factorials
Restrictions on Randomization - Split-Plot Designs

Analyze the design ignoring blocks

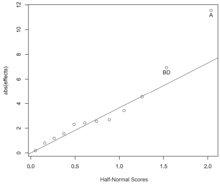
```
> mudu<-lm(y ~ A*B*C*D, data=Bdish)
> fullnormal(coef(mudu)[-1],alpha=.1)
```



John Lawson FTC Short Course - Design and Analysis of Experiments with R

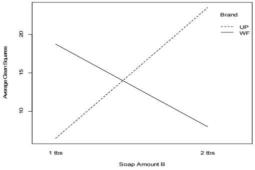
Analyze the design accounting for blocks

```
dish <- lm( y ~ Blocks + A * B * C * D, data = Bdish)
effects <- coef(dish)
effects <- effects[5:19]
effects <- effects[ !is.na(effects) ]
library(daewr)
halfnorm(effects, names(effects), alpha=.25)
```



An unlikely interaction

```
> x <- as.numeric(Bdish$B)
> x[x=="1"] <- "1 tbs"
> x[x=="2"] <- "2 tbs"
> Brand <- as.numeric(Bdish$D)
> Brand[Brand==1] <- "WF"
> Brand[Brand=="2"] <- "UP"
> interaction.plot(x, Brand, Bdish$y, type="l" ,xlab="Soap Amount B",ylab="Average Clean Squares")
```



Criteria for choosing block defining contrasts

Confounding a 2^k in blocks of size 2^q

1. Choose k-q block defining contrasts
2. Block defining contrasts plus their generalized interactions are confounded with blocks

Example: Confounding a 2^5 factorial in blocks of size 2^2=4 ⇒ $2^5/2^2 = 2^3$ = 8 blocks, 7 df
5-2 = 3 Choose ABC, CDE, ABCDE as block defining contrasts
then the generalized interactions ABDE, DE, AB, and **C** are also confounded with blocks.

To find the best generators and block defining contrasts for a particular design problem is not a simple task. Fortunately, statisticians have provided tables that show choices that are optimal in certain respects. Box et al. (1978) provide tables for block defining contrasts that will result in a minimal number of low-order interactions being confounded with blocks in a blocked 2^k design. Sun et al.(1997) provide an extensive catalog of block defining contrasts for 2^k designs and generators for 2^{k-p} designs along with the corresponding block defining contrasts that will result in best designs with regard to one of several quality criteria such as *estimability* order.

When not specied by the user, the function FrF2 in the R package FrF2 uses the block defining contrasts from Sun et al.'s (1997) catalog to create blocked 2^k designs.

Create design with Default FrF2 block contrasts

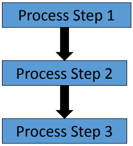
```
> Blocked25<-FrF2(32, 5, blocks=8, alias.block.2fis=TRUE, randomize=FALSE)
> summary(Blocked25)
Call:
FrF2(32, 5, blocks = 8, alias.block.2fis = TRUE, randomize = FALSE)

Experimental design of type  FrF2.blocked
32  runs
blocked design with  8  blocks of size  4

Factor settings (scale ends):
   A  B  C  D  E
1 -1 -1 -1 -1 -1
2  1  1  1  1  1

Design generating information:
$legend
[1] A=A B=B C=C D=D E=E

$`generators for design itself`
[1] full factorial

$`block generators`
[1] ABCD ACE  BCE


no aliasing of main effects or 2fis among experimental factors

Aliased with block main effects:
[1] AB CD
```

Create design with Default FrF2 block contrasts

```
The design itself:
  run.no run.no.std.rp Blocks  A  B  C  D  E
1      1         3.1.1      1 -1 -1 -1  1 -1
2      2         6.1.2      1 -1 -1  1 -1  1
3      3        28.1.3      1  1  1 -1  1  1
4      4        29.1.4      1  1  1  1 -1 -1
  run.no run.no.std.rp Blocks  A  B  C  D  E
5      5         9.2.1      2 -1  1 -1 -1 -1
6      6        16.2.2      2 -1  1  1  1  1
7      7        18.2.3      2  1 -1 -1 -1  1
8      8        23.2.4      2  1 -1  1  1 -1
   run.no run.no.std.rp Blocks  A  B  C  D  E
9       9        10.3.1      3 -1  1 -1 -1  1
10     10        15.3.2      3 -1  1  1  1 -1
11     11        17.3.3      3  1 -1 -1 -1 -1
12     12        24.3.4      3  1 -1  1  1  1
   run.no run.no.std.rp Blocks  A  B  C  D  E
13     13         4.4.1      4 -1 -1 -1  1  1
14     14         5.4.2      4 -1 -1  1 -1 -1
15     15        27.4.3      4  1  1 -1  1 -1
16     16        30.4.4      4  1  1  1 -1  1
   run.no run.no.std.rp Blocks  A  B  C  D  E
17     17         1.5.1      5 -1 -1 -1 -1 -1
18     18         8.5.2      5 -1 -1  1  1  1
19     19        26.5.3      5  1  1 -1 -1  1
20     20        31.5.4      5  1  1  1  1 -1

              . . .
class=design, type= FrF2.blocked
NOTE: columns run.no and run.no.std.rp are annotation, not part of the data frame
```

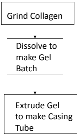

Multiple process steps make complete randomization very time consuming

Process Experiments



- Factor in Earlier Step become Whole Plot Factor
- Factors in Later Steps can be varied within and become subplot factors

Example - Process for making sausage casing



Test all 4 combinations of C and D in each batch

Sausages can be cooked in many ways from steaming to deep-fat frying, and the casing must be able to handle the stress and temperature changes without bursting. Experiments were run to determine how the combination of levels of two factors A and B in the gel making process, and the combination of levels of two factors C and D in the gel extrusion step affected the bursting strength of the final casing.

Table 8.4 *First Four Batches for Sausage-Casing Experiment*

Gel Batch	A	B	C D	- -	+ -	- +	+ +
1	-	-		2.07	2.07	2.10	2.12
2	+	-		2.02	1.98	2.00	1.95
3	-	+		2.09	2.05	2.08	2.05
4	+	+		1.98	1.96	1.97	1.97

Repeat with another lot of raw material (collagen)

Table 8.5 *Second Block of Four Batches for Sausage-Casing Experiment*

Gel			C	-	+	-	+
Batch	A	B	D	-	-	+	+
1	-	-		2.08	2.05	2.07	2.05
2	+	-		2.03	1.97	1.99	1.97
3	-	+		2.05	2.02	2.02	2.01
4	+	+		2.01	2.01	1.99	1.97

Design and Analysis of Two-Level Factorials
Two-Level Factorials
The Justification for Two-Levels
Creating and Analyzing Two-Level Factorials with R
Blocking Two-Level Factorials
Restrictions on Randomization - Split-Plot Designs

Whole plot model is like a blocked two-factor factorial

$$y_{ijk} = \mu + b_i + \alpha_j + \beta_k + \alpha\beta_{jk} + w_{ijk}$$

b_i is the random block or collagen shipment effect

α_j is the fixed effect of factor A

β_k is the fixed effect of factor B

John Lawson FTC Short Course - Design and Analysis of Experiments with R

Split-plot model has two error terms

The model for the complete split-plot experiment is obtained by adding the split-plot factors C and D and all their interactions with the other factors as shown

Block (Collagen Lot)

Block interactions (variability in gel batches)

$$
\begin{aligned}
y_{ijklm} = \mu + b_i + \alpha_j + \beta_k + \alpha\beta_{jk} + w_{ijk} \\
+ \gamma_l + \delta_m + \gamma\delta_{lm} + \alpha\gamma_{jl} + \alpha\delta_{jm} \\
+ \beta\gamma_{kl} + \beta\delta_{km} + \alpha\beta\gamma_{jkl} + \alpha\beta\delta_{jkm} \\
+ \alpha\gamma\delta_{jkl} + \beta\gamma\delta_{klm} + \alpha\beta\gamma\delta_{jklm} + \epsilon_{ijklm}
\end{aligned}
$$

Create the design with FrF2

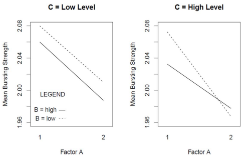
```
> FrF2(32, 4, WPs = 8, nfac.WP = 2, factor.names = (c("A","B","C","D")))
  run.no run.no.std.rp  A  B WP3  C  D
1      1         4.1.4 -1 -1  -1  1  1
2      2         1.1.1 -1 -1  -1 -1 -1
3      3         3.1.3 -1 -1  -1  1 -1
4      4         2.1.2 -1 -1  -1 -1  1
  run.no run.no.std.rp A B WP3  C  D
5      5        29.8.1 1 1   1 -1 -1
6      6        30.8.2 1 1   1 -1  1
7      7        32.8.4 1 1   1  1  1
8      8        31.8.3 1 1   1  1 -1
   run.no run.no.std.rp A  B WP3  C  D
9       9        20.5.4 1 -1  -1  1  1
10     10        18.5.2 1 -1  -1 -1  1
11     11        17.5.1 1 -1  -1 -1 -1
12     12        19.5.3 1 -1  -1  1 -1
              ...
run.no run.no.std.rp  A B WP3  C  D
29     29        15.4.3 -1 1   1  1 -1
30     30        16.4.4 -1 1   1  1  1
31     31        13.4.1 -1 1   1 -1 -1
32     32        14.4.2 -1 1   1 -1  1
class=design, type= FrF2.splitplot
NOTE: columns run.no and run.no.std.rp are annotation, not part of the data frame
```

The data frame sausage is in the daewr package

```
> library(daewr)
> library(lme4)
Loading required package: Matrix
Loading required package: Rcpp

Attaching package: 'lme4'

The following object is masked from 'package:daewr':

    cake

> rmod2<-lmer(ys~ A + B + A:B + (1|Block) + (1|A:B:Block) + C + D + C:D + A:C + A:D +
+ B:C + B:D + A:B:C + A:B:D + A:C:D + B:C:D + A:B:C:D, data=sausage)
> summary(rmod2)
Linear mixed model fit by REML ['lmerMod']
Formula: ys ~ A + B + A:B + (1 | Block) + (1 | A:B:Block) + C + D + C:D +
    A:C + A:D + B:C + B:D + A:B:C + A:B:D + A:C:D + B:C:D + A:B:C:D
   Data: sausage

REML criterion at convergence: -69.4

Scaled residuals:
    Min      1Q  Median      3Q     Max
-1.5089 -0.3102  0.0000  0.3102  1.5089

Random effects:
 Groups    Name        Variance  Std.Dev.
 A:B:Block (Intercept) 0.0003396 0.01843
 Block     (Intercept) 0.0000000 0.00000
 Residual              0.0002385 0.01544
Number of obs: 32, groups:  A:B:Block, 8; Block, 2
```


Analysis of the fixed Effects

```
> anova(rmod2)
Analysis of Variance Table
        Df    Sum Sq   Mean Sq F value
A        1 0.0068346 0.0068346 28.6517
B        1 0.0003926 0.0003926  1.6458
C        1 0.0038281 0.0038281 16.0480
D        1 0.0005281 0.0005281  2.2140
A:B      1 0.0001685 0.0001685  0.7065
C:D      1 0.0002531 0.0002531  1.0611
A:C      1 0.0001531 0.0001531  0.6419
A:D      1 0.0009031 0.0009031  3.7860
B:C      1 0.0000781 0.0000781  0.3275
B:D      1 0.0002531 0.0002531  1.0611
A:B:C    1 0.0013781 0.0013781  5.7773
A:B:D    1 0.0007031 0.0007031  2.9476
A:C:D    1 0.0000281 0.0000281  0.1179
B:C:D    1 0.0000281 0.0000281  0.1179
A:B:C:D  1 0.0000281 0.0000281  0.1179
```



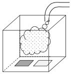
Effect of factor A depends upon the combination of levels of factors B and C

An unreplicated split-plot design

Bisgaard *et al.*(1996) described an experiment that was performed to study the plasma treatment of paper, between electrodes in a low vacuum chamber reactor, to make it more susceptible to ink.

The factors are shown below.

Factor	Levels –	Levels +	Difficulty in Changing Levels
A - pressure	Low	High	
B - Power Level	Low	High	difficult requires a new set up to change
C - Gas Flow Rate	Low	High	difficult requires a new set up to change
D - Type Gas	Oxygen	$SiCl_4$	difficult requires a new set up to change
E - Paper Type	A	B	easy both types can be treated in the same run after setup is complete



The data frame plasma is in the daewr package

Whole-Plot Effects
A, B, AB, C, AC, BC, ABC, D, AD, BD, ABD, CD, ACD, BCD, ABCD

Split-Plot Effects
E and interactions with E

```
> library(daewr)
> sol <- lm(y ~ A*B*C*D*E, data = plasma)
> effects <- coef(sol)
> effects <- effects[c(2:32)]
> Wpeffects <- effects[c(1:4, 6:11, 16:19, 26)]
> Speffects <- effects[c(5,12:15,20:25,27:31)]
```

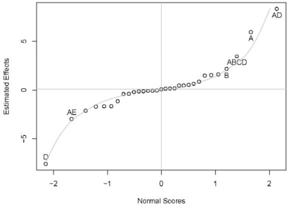
Table 8.6 *Plasma Experiment Factor Levels and Response*

A	B	C	D	E –	E +
–	–	–	–	48.6	57.0
+	–	–	–	41.2	38.2
–	+	–	–	55.8	62.9
+	+	–	–	53.5	51.3
–	–	+	–	37.6	43.5
+	–	+	–	47.2	44.8
–	+	+	–	47.2	54.6
+	+	+	–	48.7	44.4
–	–	–	+	5.0	18.1
+	–	–	+	56.8	56.2
–	+	–	+	25.6	33.0
+	+	–	+	41.8	37.8
–	–	+	+	13.3	23.7
+	–	+	+	47.5	43.2
–	+	+	+	11.3	23.9
+	+	+	+	49.5	48.2

Analysis by normal plot of all effects is misleading

```
> fullnormal(effects, names(Wpeffects), alpha = .10)
```

Figure 8.6 *Normal Plot of All Effects—Plasma Experiment*

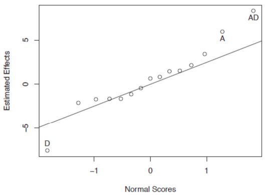


Design and Analysis of Two-Level Factorials
Two-Level Factorials
The Justification for Two-Levels
Creating and Analyzing Two-Level Factorials with R
Blocking Two-Level Factorials
Restrictions on Randomization - Split-Plot Designs

Normal plot of whole-plot effects

```
> fullnormal(Wpeffects, names(Wpeffects), alpha = .10)
```

Figure 8.4 *Normal Plot of Whole-Plot Effects—Plasma Experiment*

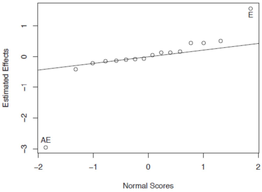


John Lawson
FTC Short Course - Design and Analysis of Experiments with R

Normal plot of split-plot effects

```
> fullnormal(Speffects, names(Speffects), alpha = .05)
```

Figure 8.5 *Normal Plot of Sub-Plot Effects—Plasma Experiment*

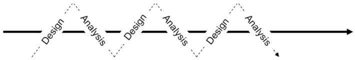


Part IV

Design and Analysis of Preliminary Experiments for Estimating Sources of Variance

Outline of Part IV

Preliminary Exploration

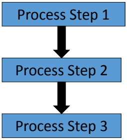


0% **Knowledge** 100%

Objective:	Preliminary Exploration	Screening Factors	Effect Estimation	Optimization	Mechanistic Modeling
No. of Factors		5 - 20	3 - 6	2 - 4	1 - 5
Purpose:	Identify Sources of Variability	Identify Important Factors	Estimate Factor Effects + Interactions	Fit Empirical Model Interpolate	Estimate Parameters of Theory Extrapolate

Identify fruitful areas for identifying factors

Sampling Experiments

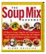


- Identify Process Steps that contribute the most variability
- Later identify factors in variable process steps that cause the variability

Two sources of variability

Hare (1988) discussed experiments to control variability in dry soup mix “intermix” (vegetable oil, salt flavorings etc.).

- too little not enough flavor
- too much too strong



Soup batch and Sample within batch

Step 1. Make a batch of soup and dry it on a rotary dryer

Step 2. Place dry soup in a mixer where intermix is injected through ports

Possible Factors

A - Ingredients
B - Cook temperature
C - Dryer temperature
D - Dryer RPM, etc

E - number of mixer ports for Vegetable oil
F - temperature of mixer jacket
G - Mixing time
H - Batch weight
I - delay time between mixing and packaging, etc. ...

Method of Moments Estimators

$$y_{ij} = \mu + t_i + \varepsilon_{(i)j} \quad i = 1,4, \quad j = 1,3, \quad k = 4, \quad r = 3$$

Table 5.4 *Variability in Dry Soup Intermix Weights*

Batch	Weight
1	0.52, 2.94, 2.03
2	4.59, 1.26, 2.78
3	2.87, 1.77, 2.68
4	1.38, 1.57, 4.10

Source	df	MS	EMS
Factor T	$t-1$	msT	$\sigma^2 + r\sigma_t^2$
Error	$t(r-1)$	msE	σ^2

Method of Moments Estimators

R Console

```
> library(daewr)
> mod1<-aov(weight ~ batch, data=soupmx)
> summary(mod1)
            Df Sum Sq Mean Sq F value Pr(>F)
batch        3  1.661  0.5535    0.32  0.811
Residuals    8 13.850  1.7312
>
```

$\sigma^2 + 3\sigma_b^2$

σ^2

$$\hat{\sigma}^2 = 1.7312$$

$$\hat{\sigma}_b^2 = \frac{0.5535 - 1.7312}{3} < 0.0$$

Maximum Likelihood and REML estimators

$$y_{ij} = \mu + t_i + \varepsilon_{(i)j} \qquad \boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \qquad \boldsymbol{\beta}' = (\mu, \boldsymbol{t}')$$

$$\begin{pmatrix} \boldsymbol{t} \\ \boldsymbol{\epsilon} \end{pmatrix} \sim MVN\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \sigma_t^2 \boldsymbol{I}_t & 0 \\ 0 & \sigma^2 \boldsymbol{I}_n \end{pmatrix}\right), \qquad \boldsymbol{I}_t \text{ is a } t \times t \text{ Identity matrix}$$

Maximum Likelihood and REML estimators

$y_{ij} = \mu + t_i + \varepsilon_{(i)j}$ $\quad$ $\boldsymbol{y} = \boldsymbol{X}\beta + \epsilon,$ $\quad$ $\beta' = (\mu, \boldsymbol{t}')$

$$\begin{pmatrix} \boldsymbol{t} \\ \epsilon \end{pmatrix} \sim MVN\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \sigma_t^2 \boldsymbol{I}_t & 0 \\ 0 & \sigma^2 \boldsymbol{I}_n \end{pmatrix} \right),$$

$\boldsymbol{I}_t$ is a $t \times t$ Identity matrix

maximum likelihood estimators for σ_t^2 and σ^2 are found my maximizing

$$L(\mu, \boldsymbol{V}|\boldsymbol{y}) = \frac{\exp\left[-\frac{1}{2}(\boldsymbol{y} - \mu\boldsymbol{1_n})'\boldsymbol{V}^{-1}(\boldsymbol{y} - \mu\boldsymbol{1_n})\right]}{(2\pi)^{\frac{1}{2}n}|V|^{\frac{1}{2}}} = \frac{\exp\left\{-\frac{1}{2}\left[\frac{ssE}{\sigma^2} + \frac{ssT}{\lambda} + \frac{(\bar{y}_{\cdot\cdot} - \mu)^2}{\lambda/n}\right]\right\}}{(2\pi)^{\frac{1}{2}n}\sigma^{2[\frac{1}{2}n]}\lambda^{\frac{1}{2}T}}$$

Preliminary

Introduction
One-Factor Designs
Two-Factor Designs
Staggered Nested Designs for Multiple Factors
Graphical Methods to Check Assumptions
Chemistry Example

Maximum Likelihood and REML estimators

$y_{ij} = \mu + t_i + \varepsilon_{(i)j}$ $\quad$ $\boldsymbol{y} = \boldsymbol{X}\beta + \epsilon,$ $\quad$ $\beta' = (\mu, \boldsymbol{t}')$

$$\begin{pmatrix} \boldsymbol{t} \\ \epsilon \end{pmatrix} \sim MVN\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \sigma_t^2 \boldsymbol{I}_t & 0 \\ 0 & \sigma^2 \boldsymbol{I}_n \end{pmatrix} \right),$$

$\boldsymbol{I}_t$ is a $t \times t$ Identity matrix

maximum likelihood estimators for σ_t^2 and σ^2 are found my maximizing

$$L(\mu, \boldsymbol{V}|\boldsymbol{y}) = \frac{\exp\left[-\frac{1}{2}(\boldsymbol{y} - \mu\boldsymbol{1_n})'\boldsymbol{V}^{-1}(\boldsymbol{y} - \mu\boldsymbol{1_n})\right]}{(2\pi)^{\frac{1}{2}n}|V|^{\frac{1}{2}}} = \frac{\exp\left\{-\frac{1}{2}\left[\frac{ssE}{\sigma^2} + \frac{ssT}{\lambda} + \frac{(\bar{y}_{\cdot\cdot} - \mu)^2}{\lambda/n}\right]\right\}}{(2\pi)^{\frac{1}{2}n}\sigma^{2[\frac{1}{2}n]}\lambda^{\frac{1}{2}T}}$$

REML estimators for σ_t^2 and σ^2 are found my maximizing

$$L(\sigma^2, \sigma_t^2 | ssT, ssE) = \frac{L(\mu, \sigma^2, \lambda|\boldsymbol{y})}{L(\mu|\bar{y}_{\cdot\cdot})}$$

Maximum Likelihood and REML estimators

```
> library(daewr)
> library(lme4)
> mod2<-lmer(weight ~ 1 + (1|batch), data=soupmx)
> summary(mod2)
Linear mixed model fit by REML ['lmerMod']
Formula: weight ~ 1 + (1 | batch)
   Data: soupmx

REML criterion at convergence: 37.5

Scaled residuals:
     Min       1Q   Median       3Q      Max
-1.56147 -0.71722 -0.01614  0.43230  1.86604

Random effects:
 Groups   Name        Variance Std.Dev.
 batch    (Intercept) 0.00     0.000
 Residual             1.41     1.187
Number of obs: 12, groups:  batch, 4

Fixed effects:
            Estimate Std. Error t value
(Intercept)   2.3742     0.3428   6.926
```

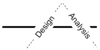
$\hat{\sigma}_b^2 = 0.0$

$\hat{\sigma}^2 = 1.41$

Preliminary

Introduction
One-Factor Designs
Two-Factor Designs
Staggered Nested Designs for Multiple Factors
Graphical Methods to Check Assumptions
Chemistry Example

The next step - screening factors

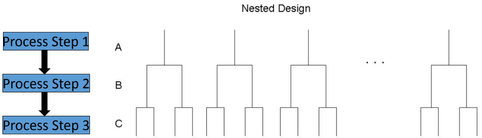


Objective:	Preliminary Exploration	Screening Factors
No. of Factors		5 - 20

Step 2. Place dry soup in a mixer where intermix is injected through ports

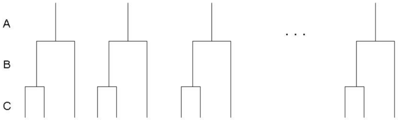
Factor Label	Name	Low Level	High Level
A	Number of Ports	1	3
B	Temperature	Cooling Water	Ambient
C	Mixing Time	60 sec.	80 sec.
D	Batch Weight	1500 lb	2000 lb
E	Delay Days	7	1

Nested design



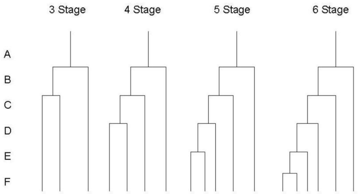
Preliminary
Introduction
One-Factor Designs
Two-Factor Designs
Staggered Nested Designs for Multiple Factors
Graphical Methods to Check Assumptions
Chemistry Example

Staggered nested design



John Lawson FTC Short Course - Design and Analysis of Experiments with R

Staggered nested design



Preliminary
Introduction
One-Factor Designs
Two-Factor Designs
Staggered Nested Designs for Multiple Factors
Graphical Methods to Check Assumptions
Chemistry Example

Method of moments estimation

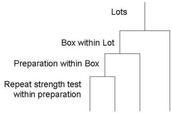
Source	Staggered Nested df	Nested df
A	$a-1$	$a-1$
B in A	a	a
C in B	a	$2a$
D in C	a	$4a$

Stages	Term	EMS
3	A	$\sigma_C^2 + (5/3)\sigma_B^2 + 3\sigma_A^2$
	B	$\sigma_C^2 + (4/3)\sigma_B^2$
	C	σ_C^2
4	A	$\sigma_D^2 + (3/2)\sigma_C^2 + (5/2)\sigma_B^2 + 4\sigma_A^2$
	B	$\sigma_D^2 + (7/6)\sigma_C^2 + (3/2)\sigma_B^2$
	C	$\sigma_D^2 + (4/3)\sigma_C^2$
	D	σ_D^2

John Lawson
FTC Short Course - Design and Analysis of Experiments with R

An Example

Mason et al. (1989) described a study where a staggered nested design was used to estimate the sources of variability in a continuous polymerization process. In this process polyethylene pellets are produced in lots of one hundred thousand pounds. A four-stage design was used to partition the source of variability in tensile strength between lots, within lots and due to the measurement process.



Data from the first 10 of 30 lots

Table 5.13 *Data from Polymerization Strength Variability Study*

	Box 1			Box 2
	Preparation			Preparation
	1		2	1
Lot	test 1	test 2	test 1	test 1
1	9.76	9.24	11.91	9.02
2	10.65	7.77	10.00	13.69
3	6.50	6.26	8.02	7.95
4	8.08	5.28	9.15	7.46
5	7.84	5.91	7.43	6.11
6	9.00	8.38	7.01	8.58
7	12.81	13.58	11.13	10.00
8	10.62	11.71	14.07	14.56
9	4.88	4.96	4.08	4.76
10	9.38	8.02	6.73	6.99

Method of moments estimators

R Console

```
> mod2<-aov(strength ~ lot + lot:box + lot:box:prep, data = polymer)
> summary(mod2)
             Df Sum Sq Mean Sq F value   Pr(>F)    
lot          29  856.0  29.516  45.552  < 2e-16 ***
lot:box      30   50.1   1.670   2.577 0.005774 ** 
lot:box:prep 30   68.4   2.281   3.521 0.000457 ***
Residuals    30   19.4   0.648                     
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> 
```

Data frame polymer is in the daewr package

$\sigma_R^2 = 0.648$

$\sigma_P^2 = (2.281 - 0.648)/(4/3) = 1.22475$

$\sigma_B^2 = (1.670 - [0.648 + (7/6)1.22475])/(3/2) = \boxed{-0.27125}$

$\sigma_L^2 = (29.516 - [0.648 + (3/2)(1.22475) + (5/2)(-0.27125)])/4 = 6.92725$

Preliminary
Introduction
One-Factor Designs
Two-Factor Designs
Staggered Nested Designs for Multiple Factors
Graphical Methods to Check Assumptions
Chemistry Example

REML estimators

R Console

```
> modr3 <- lmer( strength ~ 1 + (1|lot) + (1|lot:box) + (1|lot:box:prep), data = polymer)
> summary(modr3)
Linear mixed model fit by REML ['lmerMod']
Formula: strength ~ 1 + (1 | lot) + (1 | lot:box) + (1 | lot:box:prep)
   Data: polymer

REML criterion at convergence: 468.9

Scaled residuals: 
    Min      1Q  Median      3Q     Max 
-2.1896 -0.4119 -0.0206  0.3826  1.7703 

Random effects:
 Groups       Name        Variance Std.Dev.
 lot:box:prep (Intercept) 1.0296   1.0147  
 lot:box      (Intercept) 0.0000   0.0000  
 lot          (Intercept) 7.2427   2.6912  
 Residual                 0.6568   0.8104  
Number of obs: 120, groups:  lot:box:prep, 90; lot:box, 60; lot, 30

Fixed effects:
            Estimate Std. Error t value
(Intercept)   7.2208     0.5087    14.2
> 
```

	% Total
$\hat{\sigma}_L^2 = 7.2427$	81.1%
$\hat{\sigma}_B^2 = 0.0$	0.0%
$\hat{\sigma}_P^2 = 0.1225$	12.3%
$\hat{\sigma}_M^2 = 0.648$	7.4%

Variance components are pooled variances

	Box 1			Box 2
	Preparation			Preparation
	1		2	1
Lot	test 1	test 2	test 1	test 1
i	Y_{1i}	Y_{2i}	Y_{3i}	Y_{4i}

Source	Variance s_i^2
Error or test(prep)	$(Y_{2i} - Y_{1i})^2/2$
prep(box)	$\frac{2}{3}\left(Y_{3i} - \frac{(Y_{1i}+Y_{2i})}{2}\right)^2$
box	$\frac{3}{4}\left(Y_{4i} - \frac{(Y_{1i}+Y_{2i}+Y_{3i})}{3}\right)^2$

Computing and graphing variances in R

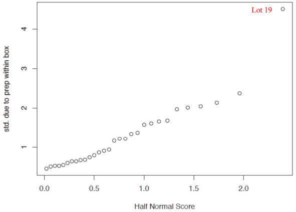
	Box 1			Box 2
	Preparation			Preparation
	1		2	1
Lot	test 1	test 2	test 1	test 1
i	Y_{1i}	Y_{2i}	Y_{3i}	Y_{4i}

Source	Variance s_i^2
Error or test(prep)	$(Y_{2i} - Y_{1i})^2/2$
prep(box)	$\frac{2}{3}\left(Y_{3i} - \frac{(Y_{1i}+Y_{2i})}{2}\right)^2$
box	$\frac{3}{4}\left(Y_{4i} - \frac{(Y_{1i}+Y_{2i}+Y_{3i})}{3}\right)^2$

```
> library(daewr)
> data(polymer)
> y <- array( polymer$strength, c(4,30) )
> sd1 <- sqrt( (y[2,] - y[1,])**2 / 2)
> sd2 <- sqrt( (2/3) * ( y[3,] - (y[1,] + y[2,]) / 2)**2 )
> sd3 <- sqrt( (3/4) * (y[4,] - (y[1,] + y[2,] + y[3,] )/3 )**2)
> osd2 <- sort(sd2)
> r <- c( 1: length(sd2))
> zscore <- qnorm( ( ( r - .5 ) / length(sd2) +1 )/ 2)
> plot( zscore, osd2, main = "Half-normal plot of prep(box) standard
+ deviations", xlab = "Half Normal Score", ylab = "std. due to prep within
+ box")
```

Computing and graphing variances in R

Figure 5.6 *Half-Normal Plot of Standard Deviations of Prep(Box)*



Preliminary

Introduction
One-Factor Designs
Two-Factor Designs
Staggered Nested Designs for Multiple Factors
Graphical Methods to Check Assumptions
Chemistry Example

Odd value in Lot 19

Table 5.18 *Raw Data for Each Lot and Calculated Standard Deviations*

lot	Y_1	Y_2	Y_3	Y_4	s_1	s_2	s_3
1	9.76	9.24	11.91	9.02	0.368	1.968	1.111
2	10.65	7.77	10.00	13.69	2.036	0.645	3.652
3	6.50	6.26	8.02	7.95	0.170	1.339	0.886
4	8.08	5.28	9.15	7.46	1.980	2.017	0.038
5	7.84	5.91	7.43	6.11	1.365	0.453	0.823
6	9.00	8.38	7.01	8.58	0.438	1.372	0.390
7	12.81	13.58	11.13	10.00	0.544	1.686	2.171
8	10.62	11.71	14.07	14.56	0.771	2.372	2.102
9	4.88	4.96	4.08	4.76	0.057	0.686	0.104
10	9.38	8.02	6.73	6.99	0.962	1.608	0.912
11	5.91	5.79	6.59	6.55	0.085	0.604	0.393
12	7.19	7.22	5.77	8.33	0.021	1.172	1.389
13	7.93	6.48	8.12	7.43	1.025	0.747	0.069
14	3.70	2.86	3.95	5.92	0.594	0.547	2.093
15	4.64	5.70	5.96	5.88	0.750	0.645	0.387
16	5.94	6.28	4.18	5.24	0.240	1.576	0.196
17	9.50	8.00	11.25	11.14	1.061	2.041	1.348
18	10.93	12.16	9.51	12.71	0.870	1.662	1.596
19	11.95	10.58	16.79	13.08	0.969	4.511	0.023

Reanalysis excluding lot 19

Table 5.19 *Comparison of Method of Moments and REML Estimates for Polymerization Study after Removing Lot 19*

Component	Method of Moments Estimator	REML Estimator
Lot (σ_a^2)	5.81864	6.09918
Box(Lot) (σ_b^2)	0.13116	0.04279
Prep(Box) (σ_c^2)	0.76517	0.79604
Error (σ^2)	0.63794	0.64364

Catalyst Support Material

•Interest in catalyst support in lab

•The rate of catalyst reaction is related to the available number of catalytic sites. To increase the number of active sites, catalysts are dispersed on a support

•Interest in making Al_2O_3 catalyst support

1. High thermal stability
2. High surface area
3. Mesoporous nature

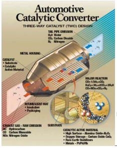
•Important catalyst support properties

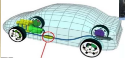
1. High surface area →increase catalyst dispersion and catalytic reaction sites →decrease reaction times.
2. Optimal pore size →each catalytic system requires a unique pore size →better diffusion and selectivity.
3. Thermal stability →many catalytic reactions take place at elevated temperatures.

Applications of Alumina Catalyst Support

- **Aluminum oxides support applications**

1. Automotive Gasoline Catalytic Converters, which converts toxic chemical (carbon monoxide and unburned hydrocarbon) in exhaust to CO_2 and H_2O.
2. Fischer-Tropsch synthesis (FTS), which liquid fuels are produced from natural gas.





Fischer-Tropsch

Process to Create Alumina Catalyst Support

Basic Synthesis Method

Mix metal salt and base

↓

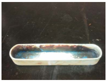
Dry

↓

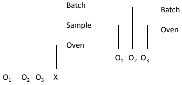
Rinse

↓

Calcine



Exploration Experiment 1



$O_1 = O_2 \neq O_3$

```
> Exp1
   Batch Oven PoreV  SA
1      1    1  1.05 172
2      1    2  1.35 188
3      1    3  1.13 164
4      2    1  1.21 183
5      2    2  1.39 193
6      2    3  1.28 190
7      3    1  1.26 182
8      3    2  1.41 189
9      3    3  1.25 183
10     4    1  1.27 172
11     4    2  1.40 183
12     4    3  1.28 172
13     5    1  1.20 171
14     5    2  1.42 189
15     5    3  1.17 171
16     6    1  1.19 175
17     6    2  1.33 180
18     6    3  1.22 179
19     7    1  1.18 165
20     7    2  1.37 183
21     7    3  1.08 163
22     8    1  1.22 167
23     8    2  1.30 169
24     8    3  1.18 184
25     9    1  1.21 173
26     9    2  1.39 186
27     9    3  1.11 165
28    10    1  1.17 156
29    10    2  1.27 168
30    10    3  1.00 155
```


Analysis of Exploration Experiment 1

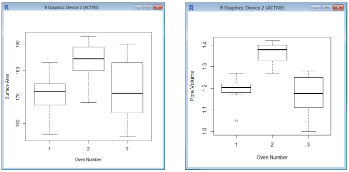
```
> modE1<-lmer(PoreV ~ 1 + (1|Batch), data=Exp1)

> summary(modE1)
Linear mixed model fit by REML ['lmerMod']
Formula: PoreV ~ 1 + (1 | Batch)
   Data: Exp1

REML criterion at convergence: -42.4

Scaled residuals:
     Min       1Q   Median       3Q      Max
-2.21247 -0.57360 -0.07284  0.72383  1.61155

Random effects:
 Groups   Name        Variance Std.Dev.
 Batch    (Intercept) 0.00000  0.0000
 Residual             0.01206  0.1098
Number of obs: 30, groups:  Batch, 10

Fixed effects:
            Estimate Std. Error t value
(Intercept)  1.24300    0.02005   61.99
```

Analysis of Exploration Experiment 1

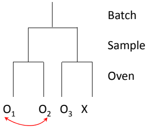
```
> modE1<-lmer(SA ~ 1 + (1|Batch), data=Exp1)
> summary(modE1)
Linear mixed model fit by REML ['lmerMod']
Formula: SA ~ 1 + (1 | Batch)
   Data: Exp1

REML criterion at convergence: 218.1

Scaled residuals:
    Min      1Q  Median      3Q     Max
-1.3054 -0.6465 -0.1551  0.8390  1.5276

Random effects:
 Groups   Name        Variance Std.Dev.
 Batch    (Intercept) 37.09    6.090
 Residual             71.77    8.472
Number of obs: 30, groups:  Batch, 10

Fixed effects:
            Estimate Std. Error t value
(Intercept)   175.67       2.47   71.12
```

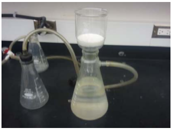

Residual Variability

```
> boxplot(SA~Oven, data=Exp1, ylab="Surface Area", xlab="Oven Number")
```



Possible Explanation



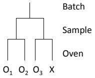


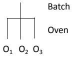
maybe extra time on the bench affects PoreV and SA not Oven

Exploratory Experiment 2

```
> Exp2
   Batch Oven PoreV  SA
1      1    1  1.19 170
2      1    2  1.18 172
3      1    3  1.05 186 <--
4      2    1  1.11 180
5      2    2  1.06 180
6      2    3  1.14 197 <--
7      3    1  1.16 214
8      3    2  1.49 208
9      3    3  1.33 292 <--
10     4    1  1.44 224
11     4    2  1.32 210
12     4    3  2.22 325 <--
```

Another Conjecture






```
> Exp2
   Batch Oven PoreV  SA
1      1    1  1.19 170
2      1    2  1.18 172
3      1    3  1.05 186
4      2    1  1.11 180
5      2    2  1.06 180
6      2    3  1.14 197
7      3    1  1.16 214
8      3    2  1.49 208
9      3    3  1.33 292
10     4    1  1.44 224
11     4    2  1.32 210
12     4    3  2.22 325
```

Batches 3 and 4 used a different (slower) filter and thus had a longer exposure time to sec-butanol which seemed to affect Pore Volume and Surface Area

Experiment to Estimate Effects

Split-Plot Fractional Factorial

```
> Exp3
   Batch Mix_Time Bench_Time Exp_Time Boats PoreV  SA
1      1        1          1        1     1  0.73 177
2      1        1          1       -1    -1  0.64 170
3      1        1         -1       -1    -1  0.66 187
4      1        1         -1        1     1  0.68 210
5      2       -1         -1        1     1  1.17 191
6      2       -1          1        1    -1  1.13 169
7      2       -1         -1       -1    -1  1.11 203
8      2       -1          1       -1     1  1.13 173
9      3        1          1       -1     1  0.95 137
10     3        1          1        1     1  0.98 137
11     3        1         -1       -1    -1  0.96 191
12     3        1         -1        1    -1    NA  NA
13     4       -1         -1        1     1  0.99 218
14     4       -1          1       -1    -1  1.06 191
15     4       -1          1        1    -1  1.24 191
16     4       -1         -1       -1     1  1.11 162
```

Batch 1: Boats = Exposure Time

Batch 2: Boats = Bench Time× Exposure Time

Batch 3: Boats = Bench Time

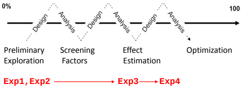
Batch 4: Boats = - Bench Time

Experiment to Further Study Relationships

Split-Plot 3^3 Fractional Factorial

```
> Exp4
   Batch Mix_Time Exp_Time Boats PoreV  SA
1      1        1        1    -1  0.93 187
2      1        1       -1     1  0.94 132
3      2        1        1     1  0.68 210
4      2        1       -1    -1  0.66 187
5      3       -1       -1    -1  1.31 170
6      3       -1        1     1  1.19 217
7      4        0        1     0  0.75 143
8      4        0        0     1  0.75 137
9      5       -1        0     0  1.00 164
10     5       -1        0     0  1.02 171
11     6       -1        1    -1  1.11 203
12     6       -1       -1     1  1.17 191
13     7        0        0     1  0.70 140
14     7        0        1     0  0.76 171
```


Results of Experiments



Effect of Factors on Catalyst Support Properties

	Properties	
Factor	Pore Volume	Surface Area
Mixing Time	+	
Bench Time		–
Exposure Time to sec-Butanol		+

1. High surface area →increase catalyst dispersion and catalytic reaction sites →decrease reaction times.
2. Optimal pore size →each catalytic system requires a unique pore size →better diffusion and selectivity.

Part V

Design and Analysis of Screening Experiments

Outline of Part V

Number of Experiments required for Two-Level Factorials

Number of Factors	Number of Experiments
4	16
5	32
6	64
7	128
8	256
9	512

One-at-a-Time Experiments

A Poor Solution is to Use One-at-a-Time Experiments

Run	A	B	C	D	E	F	G	H
1	-	-	-	-	-	-	-	-
2	+	-	-	-	-	-	-	-
3	-	+	-	-	-	-	-	-
4	-	-	+	-	-	-	-	-
5	-	-	-	+	-	-	-	-
6	-	-	-	-	+	-	-	-
7	-	-	-	-	-	+	-	-
8	-	-	-	-	-	-	+	-
9	-	-	-	-	-	-	-	+

Fractional Factorial Designs

- Method for strategically picking a subset of a two-Level Factorial
- Used for Screening purposes
- Has much higher Power for Detecting Effects than One-at-a-Time Experiments
- Can be used to estimate some interaction effects and do limited optimization

Paradigms that Justify the Use of Fractional Factorials

- *Effect Sparsity Principle*–Box and Meyer (1986)
- *Hierarchical Ordering Principle*–Wu and Hamada(2000)
- *Effect Heredity Principle*–Hamada and Wu(1992)

Procedure for Constructing a Half-Fraction

For example, to construct a one-half fraction of a 2^k design, denoted by $\frac{1}{2}2^k$ or 2^{k-1}, the procedure is as follows:

1. Write down the *base design*, a full factorial plan in $k-1$ factors using the coded factor levels $(-)$ and $(+)$.
2. Add the kth factor to the design by making its coded factor levels equal to the product of the other factor levels (i.e., the highest order interaction).
3. Use these k columns to define the design.

The Base Design

2^{4-1} Base Design

X_A	X_B	X_C
-	-	-
+	-	-
-	+	-
+	+	-
-	-	+
+	-	+
-	+	+
+	+	+

Adding an Interaction Column

2^{4-1} Base Design

X_A	X_B	X_C	X_{ABC}
-	-	-	-
+	-	-	+
-	+	-	+
+	+	-	-
-	-	+	+
+	-	+	-
-	+	+	-
+	+	+	+

Assigning the Added Factor to the Interaction

2^{4-1} Base Design

X_A	X_B	X_C	X_D (~~X_{ABC}~~)
-	-	-	-
+	-	-	+
-	+	-	+
+	+	-	-
-	-	+	+
+	-	+	-
-	+	+	-
+	+	+	+

The Defining Relationship

2^{4-1} Base Design

X_A	X_B	X_C	X_D (~~X_{ABC}~~)
-	-	-	-
+	-	-	+
-	+	-	+
+	+	-	-
-	-	+	+
+	-	+	-
-	+	+	-
+	+	+	+

$D = ABC$ *generator* of the design

$D^2 = ABCD$

or

$I = ABCD$

defining relation for the fractional factorial design

The Confounding Pattern

$A(I) = A(ABCD)$

or

$A = BCD$

X_A	X_B	X_C	X_D
-	-	-	-
+	-	-	+
-	+	-	+
+	+	-	-
-	-	+	+
+	-	+	-
-	+	+	-
+	+	+	+

$$\left.\begin{array}{r} I + ABCD \\ A + BCD \\ B + ACD \\ C + ABD \\ D + ABC \\ AB + CD \\ AC + BD \\ AD + BC \end{array}\right\}$$

confounding pattern

or *alias structure*

An Example of a Half-Fraction

Table 6.3 *Factors and Levels for Soup Mix* 2^{5-1} *Experiment*

Factor Label	Name	Low Level	High Level
A	Number of Ports	1	3
B	Temperature	Cooling Water	Ambient
C	Mixing Time	60 sec.	80 sec.
D	Batch Weight	1500 lbs	2000 lbs
E	Delay Days	7	1

Creating the Design with FrF2

```
> library(FrF2)
> soup <- FrF2(16, 5, generators = "ABCD", factor.names = list(A=c(1,3),
+  B=c("Cool","Ambient"),
+            C=c(60,80),D=c(1500,2000), E=c(7,1)), randomize = FALSE)
> soup
   A       B  C    D E
1  1    Cool 60 1500 1
2  3    Cool 60 1500 7
3  1 Ambient 60 1500 7
4  3 Ambient 60 1500 1
5  1    Cool 80 1500 7
6  3    Cool 80 1500 1
7  1 Ambient 80 1500 1
8  3 Ambient 80 1500 7
9  1    Cool 60 2000 7
10 3    Cool 60 2000 1
11 1 Ambient 60 2000 1
12 3 Ambient 60 2000 7
13 1    Cool 80 2000 1
14 3    Cool 80 2000 7
15 1 Ambient 80 2000 7
16 3 Ambient 80 2000 1
class=design, type= FrF2.generators
```

Adding the Responses

```
> y <- c(1.13, 1.25, .97, 1.70, 1.47, 1.28, 1.18, .98, .78,
+        1.36, 1.85, .62, 1.09, 1.10, .76, 2.10 )
> library(DoE.base)
> soup <- add.response( soup , y )
> soup
   A       B  C    D E    y
1  1    Cool 60 1500 1 1.13
2  3    Cool 60 1500 7 1.25
3  1 Ambient 60 1500 7 0.97
4  3 Ambient 60 1500 1 1.70
5  1    Cool 80 1500 7 1.47
6  3    Cool 80 1500 1 1.28
7  1 Ambient 80 1500 1 1.18
8  3 Ambient 80 1500 7 0.98
9  1    Cool 60 2000 7 0.78
10 3    Cool 60 2000 1 1.36
11 1 Ambient 60 2000 1 1.85
12 3 Ambient 60 2000 7 0.62
13 1    Cool 80 2000 1 1.09
14 3    Cool 80 2000 7 1.10
15 1 Ambient 80 2000 7 0.76
16 3 Ambient 80 2000 1 2.10
class=design, type= FrF2.generators
```


Screening
Introduction
Half-Fractions of Two-Level Factorial Designs
One-Quarter and Higher Fractions of Two-Level Factorial Designs
Criteria for Choosing Generators for Fractional Factorial Designs
Augmenting Fractional Factorial Designs to Resolve Confounding
Plackett-Burman and Model Robust Screening Designs

Checking the Alias Pattern

```
> mod1 <- lm( y ~ (.)^4, data = soup)
> aliases(mod1)

 A = B:C:D:E
 B = A:C:D:E
 C = A:B:D:E
 D = A:B:C:E
 E = A:B:C:D
 A:B = C:D:E
 A:C = B:D:E
 A:D = B:C:E
 A:E = B:C:D
 B:C = A:D:E
 B:D = A:C:E
 B:E = A:C:D
 C:D = A:B:E
 C:E = A:B:D
 D:E = A:B:C
```

John Lawson
FTC Short Course - Design and Analysis of Experiments with R

Paradigms that Simplify the Interpretation of Results

- *Effect Sparsity Principle*–Box and Meyer (1986)
- *Hierarchical Ordering Principle*–Wu and Hamada(2000)
- *Effect Heredity Principle*–Hamada and Wu (1992)

Screening

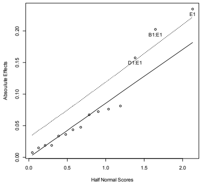
Introduction
Half-Fractions of Two-Level Factorial Designs
One-Quarter and Higher Fractions of Two-Level Factorial Designs
Criteria for Choosing Generators for Fractional Factorial Designs
Augmenting Fractional Factorial Designs to Resolve Confounding
Plackett-Burman and Model Robust Screening Designs

Analyzing the Data

```
> mod2<-lm(y~(.)^2, data=soup)
> summary(mod2)

Call:
lm.default(formula = y ~ (.)^2, data = soup)

Residuals:
ALL 16 residuals are 0: no residual degrees of freedom!

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.22625         NA      NA       NA
A1           0.07250         NA      NA       NA
B1           0.04375         NA      NA       NA
C1           0.01875         NA      NA       NA
D1          -0.01875         NA      NA       NA
E1           0.23500         NA      NA       NA
A1:B1        0.00750         NA      NA       NA
A1:C1        0.04750         NA      NA       NA
A1:D1        0.01500         NA      NA       NA
A1:E1        0.07625         NA      NA       NA
B1:C1       -0.03375         NA      NA       NA
B1:D1        0.08125         NA      NA       NA
B1:E1        0.20250         NA      NA       NA
C1:D1        0.03625         NA      NA       NA
C1:E1       -0.06750         NA      NA       NA
D1:E1        0.15750         NA      NA       NA
```

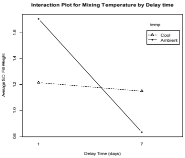
Half-Normal Plot of Coefficients

```
> library(daewr)
> LGB(coef(mod2)[-1], rpt=FALSE)
```



Interpretation of Results

```
> soup <- FrF2(16, 5, generators = "ABCD", factor.names =
+ list(Ports=c(1,3),Temp=c("Cool","Ambient"), MixTime=c(60,80),
+ BatchWt=c(1500,2000), delay=c(7,1)), randomize = FALSE)
> y <- c(1.13, 1.25, .97, 1.70, 1.47, 1.28, 1.18, .98, .78,
+ 1.36, 1.85, .62, 1.09, 1.10, .76, 2.10 )
> library(DoE.base)
> soup <- add.response( soup , y )
> delay <- as.numeric(sub(-1, 7, soup$delay))
> temp <- soup$Temp
> interaction.plot(delay, temp, soup$y, type="b",
+ pch=c(24,18,22), leg.bty="o",
+ main="Interaction Plot for Mixing Temperature by Delay time",
+ xlab="Delay Time (days)", ylab="Average S.D. Fill Weight")
```



Confounding in Higher Order Fractions

$\frac{1}{2^p}2^k = 2^{k-p}$ k is the number of factors, p is the fraction power

- In a one half fraction of a 2^k experiment every effect that could be estimated was confounded with one other effect, thus one half the effects had to be assumed negligible in order to interpret or explain the results
- In a one quarter fraction of a 2^k experiment every effect that can be estimated is confounded with three other effects, thus three quarters of the effects must be assumed negligible in order to interpret or explain the results
- In a one eighth fraction of a 2^k experiment every effect that can be estimated is confounded with seven other effects, thus seven eights of the effects must be assumed negligible in order to interpret or explain the results, etc.

Procedure for Constructing Higher Order Fractions

Creating a 2^{k-p} Design

1. Create a full two-level factorial in k-p factors
2. Add each of the remaining p factors by assigning them to a column of signs for an interaction among the first k-p columns

Example of Quarter Fraction

X_A	X_B	X_C	$\overbrace{X_A X_B}^{X_D}$	$\overbrace{X_A X_C}^{X_E}$	$X_B X_C$	$X_A X_B X_C$
−	−	−	+	+	+	−
+	−	−	−	−	+	+
−	+	−	−	+	−	+
+	+	−	+	−	−	−
−	−	+	+	−	−	+
+	−	+	−	+	−	−
−	+	+	−	−	+	−
+	+	+	+	+	+	+

Example of Quarter Fraction

X_A	X_B	X_C	$\overbrace{X_A X_B}^{X_D}$	$\overbrace{X_A X_C}^{X_E}$	$X_B X_C$	$X_A X_B X_C$
−	−	−	+	+	+	−
+	−	−	−	−	+	+
−	+	−	−	+	−	+
+	+	−	+	−	−	−
−	−	+	+	−	−	+
+	−	+	−	+	−	−
−	+	+	−	−	+	−
+	+	+	+	+	+	+

$\Downarrow$

X_A	X_B	X_C	X_D	X_E
−	−	−	+	+
+	−	−	−	−
−	+	−	−	+
+	+	−	+	−
−	−	+	+	−
+	−	+	−	+
−	+	+	−	−
+	+	+	+	+

$D = AB$ and $E = AC$

These are the generators

Example of Quarter Fraction

$D = AB$ and $E = AC$
$I = ABD$ and $I = ACE$ } the generators

the generalized interaction

since $I^2 = I$ $\quad I = ABD(ACE) \quad I = BCDE$

$I = ABD = ACE = BCDE$

the defining relation

Create the Design in FrF2

```
> frac <- FrF2( 16, 6, generators = c("AB", "AC"),randomize=FALSE)
> frac
    A  B  C  D  E  F
1  -1 -1 -1 -1  1  1
2   1 -1 -1 -1 -1 -1
3  -1  1 -1 -1 -1  1
4   1  1 -1 -1  1 -1
5  -1 -1  1 -1  1 -1
6   1 -1  1 -1 -1  1
7  -1  1  1 -1 -1 -1
8   1  1  1 -1  1  1
9  -1 -1 -1  1  1  1
10  1 -1 -1  1 -1 -1
11 -1  1 -1  1 -1  1
12  1  1 -1  1  1 -1
13 -1 -1  1  1  1 -1
14  1 -1  1  1 -1  1
15 -1  1  1  1 -1 -1
16  1  1  1  1  1  1
class=design, type= FrF2.generators
```

View the Alias Structure

```
> y <- runif( 16, 0, 1 )
> aliases( lm( y ~ (.)^3, data = frac) )

 A = B:E = C:F
 B = C:E:F = A:E
 C = B:E:F = A:F
 E = A:B = B:C:F
 F = A:C = B:C:E
 A:D = C:D:F = B:D:E
 B:C = E:F = A:B:F = A:C:E
 B:D = A:D:E
 B:F = C:E = A:B:C = A:E:F
 C:D = A:D:F
 D:E = A:B:D
 D:F = A:C:D
 B:C:D = D:E:F
 B:D:F = C:D:E
```


Some Generators Better than Others

```
> frac <- FrF2( 16, 6, generators = c("ABC", "BCD"),randomize=FALSE)
> aliases( lm( y ~ (.)^3, data = frac) )

 A = B:C:E = D:E:F
 B = A:C:E = C:D:F
 C = B:D:F = A:B:E
 D = A:E:F = B:C:F
 E = A:D:F = A:B:C
 F = A:D:E = B:C:D
 A:B = C:E
 A:C = B:E
 A:D = E:F
 A:E = B:C = D:F
 A:F = D:E
 B:D = C:F
 B:F = C:D
 A:B:D = A:C:F = B:E:F = C:D:E
 A:B:F = A:C:D = B:D:E = C:E:F
```

Criteria for Choosing Generators

- Resolution–Box and Hunter(1961)
- Minimum Aberration–Fries and Hunter 1980
- Maximum Number of Clear Effects–Chen *et. al.*(1993)

Criteria for Choosing Generators

Resolution–Shortest Word in the Defining Relation

Resolution III Main effects confounded with two-factor interactions

Resolution IV Main effects confounded with three-factor interactions, two-factor interactions confounded with other two-factor interactions

Resolution V Main effects and two-factor interactions estimable, assuming three factor and higher order interactions negligible

Resolution R Each subset of R-1 factors forms a full factorial possibly replicated

FrF2 Default-Minimum Aberration Design

```
> ## maximum resolution minimum aberration design with 9 factors in 32 runs
> ## show design information instead of design itself
> design.info(FrF2(32,9))

$catlg.entry
Design:  9-4.1
   32  runs,  9  factors,
   Resolution  IV
   Generating columns:  7 11 19 29
   WLP (3plus):  0 6 8 0 0 ,  8  clear 2fis
 Factors with all 2fis clear:  J

$aliased
$aliased$legend
[1] "A=A" "B=B" "C=C" "D=D" "E=E" "F=F" "G=G" "H=H" "J=J"

$aliased$main
character(0)

$aliased$fi2
 [1] "AB=CF=DG=EH" "AC=BF"       "AD=BG"       "AE=BH"       "AF=BC"
 [6] "AG=BD"       "AH=BE"       "CD=FG"       "CE=FH"       "CG=DF"
[11] "CH=EF"       "DE=GH"       "DH=EG"
```

8 Clear two-factor interactions

Screening
Introduction
Half-Fractions of Two-Level Factorial Designs
One-Quarter and Higher Fractions of Two-Level Factorial Designs
Criteria for Choosing Generators for Fractional Factorial Designs
Augmenting Fractional Factorial Designs to Resolve Confounding
Plackett-Burman and Model Robust Screening Designs

FrF2 Option-Maximum Number of Clear Effects

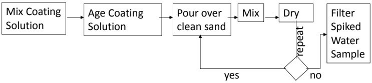
```
> ## maximum number of free 2-factor interactions instead of minimum aberration
> ## show design information instead of design itself
>design.info(FrF2(32,9,MaxC2=TRUE))
$catlg.entry
Design:  9-4.2
   32  runs,  9  factors,
   Resolution  IV
   Generating columns:  7 11 13 30
   WLP (3plus):  0 7 7 0 0 ,  15  clear 2fis
 Factors with all 2fis clear:  E J

$aliased
$aliased$legend
[1] "A=A" "B=B" "C=C" "D=D" "E=E" "F=F" "G=G" "H=H" "J=J"

$aliased$main
character(0)

$aliased$fi2
[1] "AB=CF=DG" "AC=BF=DH" "AD=BG=CH" "AF=BC=GH" "AG=BD=FH" "AH=CD=FG" "BH=CG=DF"
```

15 Clear two-factor interactions

John Lawson FTC Short Course - Design and Analysis of Experiments with R

Example of One-eighth Fraction

Iron Oxide Coated Sand (IOCS) used to remove arsenic from ground water in simple household filtration systems. Coating solution made of ferric nitrate and sodium hydroxide with NAOH added to control pH.



Ramakrishna *et. al.* (2006) conducted experiments to optimize
The coating process.

Factors and Levels

Table 6.7 *Factors and Levels for Arsenic Removal Experiment*

Label	Factors	Levels	
		−	+
A	coating pH	2.0	12.0
B	drying temperature	110°	800°
C	Fe concentration in coating	0.1 M	2 M
D	number of coatings	1	2
E	aging of coating	4 hrs	12 days
F	pH of spiked water	5.0	8.0
G	mass of adsorbent	0.1 g	1 g

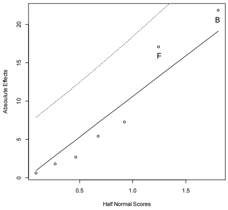
Create Design with FrF2 in Coded Factor Levels

```
> arsrm<-FrF2(8,6,generators = c("AB", "AC", "BC"), randomize=FALSE)
> y<-c(69.95, 58.65, 56.25, 53.25, 94.40, 73.45, 10.0, 2.11)
> library(DoE.base)
> arsrm2<-add.response(arsrm,y)
> arsrm2
   A  B  C  D  E  F     y
1 -1 -1 -1  1  1  1 69.95
2  1 -1 -1 -1 -1  1 58.65
3 -1  1 -1 -1  1 -1 56.25
4  1  1 -1  1 -1 -1 53.25
5 -1 -1  1  1 -1 -1 94.40
6  1 -1  1 -1  1 -1 73.45
7 -1  1  1 -1 -1  1 10.00
8  1  1  1  1  1  1  2.11
class=design, type= FrF2.generators
```


Analysis of the Data

```
> Lmod<-lm(y ~ (.)^2,data=arsrm2)
> estef<-coef(Lmod)[c(2:7,12)]
> library(daewr)
> LGB(estef,rpt=FALSE)
```

```
> aliases(Lmod)

 A = B:D = C:E
 B = C:F = A:D
 C = B:F = A:E
 D = E:F = A:B
 E = D:F = A:C
 F = B:C = D:E
 A:F = B:E = C:D
```



Possible Interpretations of Results from *'Effect Heredity'*

Important factors	Optimal Levels
1. B – Drying Temperature & F – PH of Spiked Water	Low Drying Temp. and Low PH
2. B – Drying Temperature & BC interaction C – Fe concentration in coating	Low Drying Temp. High Fe Conc.
3. F – PH of Spiked Water & CF interaction	Low PH High Fe conc.

Fractional Factorials in Split-Plot Designs

				$(I = PQR)$			
			P	−	+	−	+
$(I = ABC)$			Q	−	−	+	+
A	B	C	R	+	−	−	−
−	−	+		x	x	x	x
+	−	−		x	x	x	x
−	+	−		x	x	x	x
+	+	+		x	x	x	x

$$(I + ABC) \times (I + PQR) = I + ABC + PQR + ABCPQR$$

Resolution III

Split-Plot Confounding

$P = -QR$ when whole-plot factor A is at its low level

$P = +QR$ when the whole-plot factor A is at its high level

$(I = ABC)$			
A	B	C	
$-$	$-$	$+$	$I = -PQR$
$+$	$-$	$-$	$I = +PQR$
$-$	$+$	$-$	$I = -PQR$
$+$	$+$	$+$	$I = +PQR$

Resolution III, but less aberration

$$P = AQR \Rightarrow (I + ABC)(I + APQR) = I + ABC + APQR + BCPQR$$

Creating a Minimum Aberration Split-Plot Fractional Factorial with FrF2

```
> library(FrF2)
> SPFF2 <-FrF2(16,6, WPs = 4, nfac.WP = 3, factor.names = c("A","B","C","P","Q","R"))
> print(SPFF2)
  run.no run.no.std.rp A  B  C  P  Q  R
1      1        12.3.4 1 -1 -1  1  1  1
2      2         9.3.1 1 -1 -1 -1 -1  1
3      3        11.3.3 1 -1 -1  1 -1 -1
4      4        10.3.2 1 -1 -1 -1  1 -1
  run.no run.no.std.rp A B C  P  Q  R
5      5        14.4.2 1 1 1 -1  1 -1
6      6        16.4.4 1 1 1  1  1  1
7      7        15.4.3 1 1 1  1 -1 -1
8      8        13.4.1 1 1 1 -1 -1  1
   run.no run.no.std.rp  A B  C  P  Q  R
9       9          5.2.1 -1 1 -1 -1 -1 -1
10     10          7.2.3 -1 1 -1  1 -1  1
11     11          8.2.4 -1 1 -1  1  1 -1
12     12          6.2.2 -1 1 -1 -1  1  1
   run.no run.no.std.rp  A  B C  P  Q  R
13     13          4.1.4 -1 -1 1  1  1 -1
14     14          2.1.2 -1 -1 1 -1  1  1
15     15          1.1.1 -1 -1 1 -1 -1 -1
16     16          3.1.3 -1 -1 1  1 -1  1
class=design, type= FrF2.splitplot
NOTE: columns run.no and run.no.std.rp are annotation, not part of the data frame
```

Checking the Alias Pattern

```
> y<-rnorm(16,0,1)
> aliases(lm( y ~ (.)^3, data=SPFF2))

 A = P:Q:R = B:C
 B = A:C
 C = A:B
 P = A:Q:R
 Q = A:P:R
 R = A:P:Q
 A:P = Q:R = B:C:P
 A:Q = P:R = B:C:Q
 A:R = P:Q = B:C:R
 B:P = A:C:P = C:Q:R
 B:Q = A:C:Q = C:P:R
 B:R = A:C:R = C:P:Q
 C:P = A:B:P = B:Q:R
 C:Q = A:B:Q = B:P:R
 C:R = A:B:R = B:P:Q
```


Screening

Introduction
Half-Fractions of Two-Level Factorial Designs
One-Quarter and Higher Fractions of Two-Level Factorial Designs
Criteria for Choosing Generators for Fractional Factorial Designs
Augmenting Fractional Factorial Designs to Resolve Confounding
Plackett-Burman and Model Robust Screening Designs

Analyzing a Split-Plot Fractional Factorial

8.5.2 Analysis of a Fractional Factorial Split-Plot

Table 8.10 *Fractional Factorial Split-Plot Design for Gear Distortion*

			P	−	+	−	+
A	B	C	Q	−	−	+	+
−	−	−			x	x	
+	−	−		x			x
−	+	−		x			x
+	+	−			x	x	
−	−	+		x			x
+	−	+			x	x	
−	+	+			x	x	
+	+	+		x			x

The defining relation is $I = ABCPQ$, and the response was the dishing of the gears.

John Lawson FTC Short Course - Design and Analysis of Experiments with R

Whole-Plot and Sub-Plot Effects

Table 8.11 *Estimable Effects for Gear Distortion Experiment*

Whole-Plot Effects	Sub-Plot Effects
$A + BCPQ$	$P + ABCQ$
$B + ACPQ$	$Q + ABCP$
$C + ABPQ$	$AP + BCQ$
$AB + CPQ$	$AQ + BCP$
$AC + BPQ$	$BP + ACQ$
$BC + APQ$	$BQ + ACP$
$ABC + PQ$	$CP + ABQ$
	$CQ + ABP$

Screening
Introduction
Half-Fractions of Two-Level Factorial Designs
One-Quarter and Higher Fractions of Two-Level Factorial Designs
Criteria for Choosing Generators for Fractional Factorial Designs
Augmenting Fractional Factorial Designs to Resolve Confounding
Plackett-Burman and Model Robust Screening Designs

Analysis with R

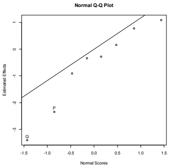
```
> spexp <- FrF2(16,5,WPs=8,nfac.WP=3, factor.names=c("A","B","C","P","Q"),randomize=FALSE)
> y<-c(18.0,21.5,27.5,17.0,22.5,15.0,19.0,22.0,13.0,-4.5,17.5,14.5,0.5,5.5,24.0,13.5)
> sol<-lm( y~A*B*C*P*Q, data=spexp)
> summary(sol)

Call:
lm.default(formula = y ~ A * B * C * P * Q, data = spexp)

Residuals:
ALL 16 residuals are 0: no residual degrees of freedom!

Coefficients: (16 not defined because of singularities)
                 Estimate Std. Error t value Pr(>|t|)
   1    (Intercept)  15.4062         NA      NA       NA
 → 2    A1           -4.9063         NA      NA       NA
 → 3    B1           -0.1562         NA      NA       NA
 → 4    C1            3.9688         NA      NA       NA
   5    P1           -2.3438         NA      NA       NA
   6    Q1           -3.4062         NA      NA       NA
 → 7    A1:B1         0.5313         NA      NA       NA
 → 8    A1:C1         2.9063         NA      NA       NA
 → 9    B1:C1         0.4062         NA      NA       NA
   10   A1:P1        -0.9063         NA      NA       NA
   11   B1:P1         1.0938         NA      NA       NA
   12   C1:P1        -0.2812         NA      NA       NA
   13   A1:Q1        -0.3438         NA      NA       NA
   14   B1:Q1         0.1563         NA      NA       NA
   15   C1:Q1         0.7812         NA      NA       NA
 → 16   P1:Q1         0.5938         NA      NA       NA
```

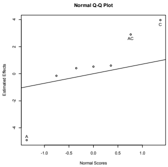
Whole Plot Effects (arrows → mark rows 2, 3, 4, 7, 8, 9, 16)

note: ABC=PQ

Separate Normal Plots of Whole-Plot and Sub-Plot Effects

```
> effects <-coef(sol)
> Wpeffects <- effects[ c(2:4, 7:9, 16) ]
> Speffects <- effects[ c(5:6, 10:15) ]
> fullnormal(Speffects, names(Speffects), alpha=.20)
> fullnormal(Wpeffects, names(Wpeffects), alpha=.10)
```





Augmenting by Foldover

Design Augmented by 2^{6-3}_{III} Design with Signs Reversed on Factor B

Run	A	B	C	D	E	F
1	−	−	−	+	+	+
2	+	−	−	−	−	+
3	−	+	−	−	+	−
4	+	+	−	+	−	−
5	−	−	+	+	−	−
6	+	−	+	−	+	−
7	−	+	+	−	−	+
8	+	+	+	+	+	+
9	−	+	−	+	+	+
10	+	+	−	−	−	+
11	−	−	−	−	+	−
12	+	−	−	+	−	−
13	−	+	+	+	−	−
14	+	+	+	−	+	−
15	−	−	+	−	−	+
16	+	−	+	+	+	+

defining relation is

$I = ABD = ACE = BCF = DEF = BCDE = ACDF = ABEF$

D confounded with *AB*

defining relation is

$I = -ABD = ACE = -BCF = DEF = -BCDE = ACDF = -ABEF$

defining relation is

$I = ACE = DEF = ACDF$

B is clear and
D no longer confounded with *AB*

Augmenting the IOCS Experiment

```
> arsrm3<-fold.design(arsrm, columns='full')
> y<-c(69.95,58.65,56.25,53.25,94.4,73.45,10.0,2.11,16.2,52.85,9.05,31.1,7.4,
+ 9.9,10.85,48.75)
> arsrm4<-add.response(arsrm3,y)
> arsrm4
    A  B  C     fold  D  E  F     y
1  -1 -1 -1 original  1  1  1 69.95
2   1 -1 -1 original -1 -1  1 58.65
3  -1  1 -1 original -1  1 -1 56.25
4   1  1 -1 original  1 -1 -1 53.25
5  -1 -1  1 original  1 -1 -1 94.40
6   1 -1  1 original -1  1 -1 73.45
7  -1  1  1 original -1 -1  1 10.00
8   1  1  1 original  1  1  1  2.11
9   1  1  1   mirror -1 -1 -1 16.20
10 -1  1  1   mirror  1  1 -1 52.85
11  1 -1  1   mirror  1 -1  1  9.05
12 -1 -1  1   mirror -1  1  1 31.10
13  1  1 -1   mirror -1  1  1  7.40
14 -1  1 -1   mirror  1 -1  1  9.90
15  1 -1 -1   mirror  1  1 -1 10.85
16 -1 -1 -1   mirror -1 -1 -1 48.75
class=design, type= FrF2.generators.folded
```

Combining a resolution III design with a mirror image (signs reversed on all factors) results in a resolution IV design where no main effect is confounded with a two-factor interaction

Alternative Explanations after Analysis of Combined Data

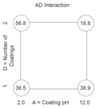
AD confounded with *CF* in the combined data

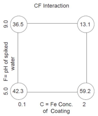
$$(F,\ B,\ A,\ AD)$$

$$\%\ removal = 37.76 - 12.99\left(\frac{pHs-7.0}{2.0}\right) - 11.76\left(\frac{temp-455^\circ}{345^\circ}\right) - 8.89\left(\frac{pHc-7.0}{5.0}\right) - 10.09\left(\frac{pHs-7.0}{2.0}\right)\left(\frac{number\ coats-.75}{.5}\right)$$

$$(F,\ B,\ A,\ CF)$$

$$\%\ removal = 37.76 - 12.99\left(\frac{pHs-7.0}{2.0}\right) - 11.76\left(\frac{temp-455^\circ}{345^\circ}\right) - 8.89\left(\frac{pHc-7.0}{5.0}\right) - 10.09\left(\frac{Fe-1.05M}{0.95M}\right)\left(\frac{pHs-7.0}{2.0}\right)$$





Augmentation by Optimal Design

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

$$\boldsymbol{y} = \begin{pmatrix} 69.95 \\ 58.65 \\ 56.25 \\ 53.25 \\ 94.40 \\ 73.45 \\ 10.00 \\ 2.11 \\ 16.20 \\ 52.85 \\ 9.05 \\ 31.10 \\ 7.40 \\ 9.90 \\ 10.85 \\ 48.75 \\ \\ \end{pmatrix}, \quad \boldsymbol{X} = \begin{pmatrix} 1 & -1 & -1 & -1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 \\ 1 & -1 & 1 & 1 & -1 & 1 & 1 \\ 1 & -1 & -1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & -1 \\ 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 1 & -1 & -1 \\ 1 & 1 & 1 & 1 & -1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & 1 & 1 \\ \\ \end{pmatrix}, \quad \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_{bl} \\ \beta_A \\ \beta_B \\ \beta_F \\ \beta_{AD} \\ \beta_{CF} \end{pmatrix}$$

Additional runs to make $X'X$ invertible

Choose additional runs to maximize $|X'X|$ i.e., D-optimal (Dykstra(1971))

Change Factors to Numeric in New Data Frame

```
> A <- (as.numeric(arsrm3$A)-1.5)/.5
> B <- (as.numeric(arsrm3$B)-1.5)/.5
> C <- (as.numeric(arsrm3$C)-1.5)/.5
> D <- (as.numeric(arsrm3$D)-1.5)/.5
> E <- (as.numeric(arsrm3$E)-1.5)/.5
> F <- (as.numeric(arsrm3$F)-1.5)/.5
> Block<-arsrm3$fold
> augmn<-data.frame(A,B,C,D,E,F,Block)
> augmn
    A  B  C  D  E  F    Block
1  -1 -1 -1  1  1  1 original
2   1 -1 -1 -1 -1  1 original
3  -1  1 -1 -1  1 -1 original
4   1  1 -1  1 -1 -1 original
5  -1 -1  1  1 -1 -1 original
6   1 -1  1 -1  1 -1 original
7  -1  1  1 -1 -1  1 original
8   1  1  1  1  1  1 original
9   1  1  1 -1 -1 -1   mirror
10 -1  1  1  1  1 -1   mirror
11  1 -1  1  1 -1  1   mirror
12 -1 -1  1 -1  1  1   mirror
13  1  1 -1 -1  1  1   mirror
14 -1  1 -1  1 -1  1   mirror
15  1 -1 -1  1  1 -1   mirror
16 -1 -1 -1 -1 -1 -1   mirror
```

Use Federov Algorithm in AlgDesign Package to Find 8 Additional Runs that Maximize the Determinant

```
> library(AlgDesign)
> cand<-gen.factorial(levels = 2, nVar = 6, varNames = c("A","B","C","D","E","F"))
> Block<-rep('cand',64)
> cand<-data.frame(A=cand$A, B=cand$B, C=cand$C, D=cand$D, E=cand$E, F=cand$F,
+ Block)
> all<-rbind(augmn, cand)
> fr<-1:16
> optim<-optFederov( ~ A + B + F + I(A*D) + I(C*F), data=all, nTrials =24,
+ criterion = "D", nRepeats =10, augment=TRUE, rows=fr)
> newruns<-optim$design[ 17:24, ]
> newruns
    A  B  C  D  E  F Block
18  1 -1 -1 -1 -1 -1  cand
23 -1  1  1 -1 -1 -1  cand
32  1  1  1  1 -1 -1  cand
43 -1  1 -1  1  1 -1  cand
49 -1 -1 -1 -1 -1  1  cand
60  1  1 -1  1 -1  1  cand
63 -1  1  1  1 -1  1  cand
72  1  1  1 -1  1  1  cand
```


Plackett-Burman Designs Obtained by Cyclically Rotation

Table 6.9 *Factor Levels for First Run of Plackett-Burman Design*

Run Size	Factor Levels
12	+ + − + + + − − − + −
20	+ + − − + + + + − + − + − − − − + + −
24	+ + + + + − + − + + − − + + − − + − + − − − −

Creating a PB Design with FrF2

```
> library(FrF2)
> pb( nruns = 12, randomize=FALSE)
    A  B  C  D  E  F  G  H  J  K  L
1   1  1 -1  1  1  1 -1 -1 -1  1 -1
2  -1  1  1 -1  1  1  1 -1 -1 -1  1
3   1 -1  1  1 -1  1  1  1 -1 -1 -1
4  -1  1 -1  1  1 -1  1  1  1 -1 -1
5  -1 -1  1 -1  1  1 -1  1  1  1 -1
6  -1 -1 -1  1 -1  1  1 -1  1  1  1
7   1 -1 -1 -1  1 -1  1  1 -1  1  1
8   1  1 -1 -1 -1  1 -1  1  1 -1  1
9   1  1  1 -1 -1 -1  1 -1  1  1 -1
10 -1  1  1  1 -1 -1 -1  1 -1  1  1
11  1 -1  1  1  1 -1 -1 -1  1 -1  1
12 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
class=design, type= pb
```


Example use of a Plackett-Burman Design

Hunter et al. (1982) used a Plackett-Burman Design to study the fatigue life of weld-repaired castings.

Table 6.11 *Design Matrix and Lifetime Data for Cast Fatigue Experiment*

Run	A	B	C	D	E	F	G	c8	c9	c10	c11	
1	+	−	+	+	+	−	−	−	+	−	+	4.733
2	−	+	+	+	−	−	−	+	−	+	+	4.625
3	+	+	+	−	−	−	+	−	+	+	−	5.899
4	+	+	−	−	−	+	−	+	+	−	+	7.000
5	+	−	−	−	+	−	+	+	−	+	+	5.752
6	−	−	−	+	−	+	+	−	+	+	+	5.682
7	−	−	+	−	+	+	−	+	+	+	−	6.607
8	−	+	−	+	+	−	+	+	+	−	−	5.818
9	+	−	+	+	−	+	+	+	−	−	−	5.917
10	−	+	+	−	+	+	+	−	−	−	+	5.863
11	+	+	−	+	+	+	−	+	−	+	−	6.058
12	−	−	−	−	−	−	−	−	−	−	−	4.809

Note: This design is created using a different first row than FrF2 uses.

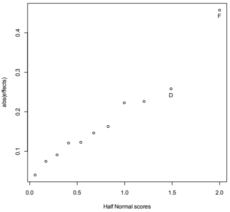
Recall the Design from the BsMD package

```
> data( PB12Des, package = "BsMD" )
> colnames(PB12Des) <- c("c11", "c10", "c9", "c8", "G", "F", "E", "D", "C", "B", "A")
> castf <- PB12Des[c(11,10,9,8,7,6,5,4,3,2,1)]
> castf
    A  B  C  D  E  F  G c8 c9 c10 c11
1   1 -1  1  1  1 -1 -1 -1  1  -1   1
2  -1  1  1  1 -1 -1 -1  1 -1   1   1
3   1  1  1 -1 -1 -1  1 -1  1   1  -1
4   1  1 -1 -1 -1  1 -1  1  1  -1   1
5   1 -1 -1 -1  1 -1  1  1 -1   1   1
6  -1 -1 -1  1 -1  1  1 -1  1   1   1
7  -1 -1  1 -1  1  1 -1  1  1   1  -1
8  -1  1 -1  1  1 -1  1  1  1  -1  -1
9   1 -1  1  1 -1  1  1  1 -1  -1  -1
10 -1  1  1 -1  1  1  1 -1 -1  -1   1
11  1  1 -1  1  1  1 -1 -1 -1   1  -1
12 -1 -1 -1 -1 -1 -1 -1 -1 -1  -1  -1
```


Analysis Shows only Factor F Possibly Significant

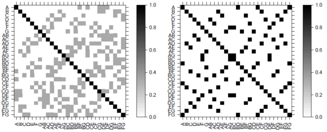
```
> y<-c(4.733, 4.625, 5.899, 7.0, 5.752, 5.682,
+ 6.607, 5.818, 5.917, 5.863, 6.058, 4.809)
> castf<-cbind(castf,y)
> modpb<-lm(y~ (.), data=castf)
> library(daewr)
> cfs<-coef(modpb)[2:12]
> names<-names(cfs)
> halfnorm(cfs, names, alpha = .35,
+ refline=FALSE)
```

$R^2 = .55$



Partially Confounded Main Effects Allows Estimation of Some Interactions by Regression

Figure 6.13 *Color Map Comparison of Confounding between PB and FF Designs*



(a) Plackett-Burman Design (b) 2^{7-4}_{III} design

Jones and Nachtsheim(2011) Propose a Forward Stepwise Regression Algorithm Guided by *Effect Heredity*

1. Model matrix includes main effects and two-factor interactions
2. When an interaction enters as the next term in the model, main effects involved in that interaction are included to preserve *effect heredity*

istep, fstep, bstep Functions in daewr Package Perform this Algorithm - FG interaction first term entered

```
> des<-castf[ , c(1:7)]
> y<-castf[ ,12]
> library(daewr)
> trm<-ihstep(y,des)

Call:
lm(formula = y ~ (.), data = d1)

Residuals:
     Min       1Q   Median       3Q      Max
-0.49700 -0.07758  0.02650  0.07867  0.44500

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  5.73025    0.07260  78.930  7.4e-13 ***
F            0.45758    0.07260   6.303 0.000232 ***
G            0.09158    0.07260   1.261 0.242669
F.G         -0.45875    0.07260  -6.319 0.000228 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2515 on 8 degrees of freedom
Multiple R-squared:  0.9104,    Adjusted R-squared:  0.8767
F-statistic: 27.08 on 3 and 8 DF,  p-value: 0.0001531
```


This Interaction was Detected with Forward Stepwise Regression

Table 6.13 *Summary of Data from Cast Fatigue Experiment*

Factor F	Factor G −	Factor G +
−	4.733	5.899
	4.625	5.752
	4.809	5.818
+	6.058	5.682
	7.000	5.917
	6.607	5.863

Alternative to Plackett-Burman when 16 Runs Needed

Jones and Montgomery (2010) have proposed alternate 16-run screening designs for 6, 7, and 8 factors

```
> library(daewr)
> ascr <-Altscreen(nfac = 6, randomize = FALSE)
> head(ascr)
   A  B  C  D  E  F
1  1  1  1  1  1  1
2  1  1 -1 -1 -1 -1
3 -1 -1  1  1 -1 -1
4 -1 -1 -1 -1  1  1
5  1  1  1 -1  1 -1
6  1  1 -1  1 -1  1
```

nfac $= 6, 7,$ or 8

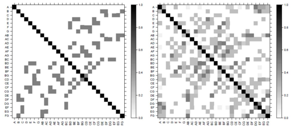
Alternative to Plackett-Burman when 16 Runs Needed

Li and Nachtsheim (2000) also developed 8-, 12-, and 16-run model robust screening designs.

```
> library(daewr)
> MR8 <- ModelRobust('MR8m5g2', randomize = FALSE)
> head(MR8)
   A  B  C  D  E
1 -1  1  1  1 -1
2 -1 -1 -1 -1 -1
3 -1  1 -1 -1  1
4  1  1  1  1  1
5  1  1 -1  1 -1
6 -1 -1 -1  1  1
```

Main Effects Partially Confounded with Two-Factor Interactions in These Designs

Figure 6.16 *Color Map Comparison of Confounding between Alternate Screening and Model Robust Designs*



(a) Alternate Screening 7 factors

(b) Model Robust m=7, g=5

Part VI

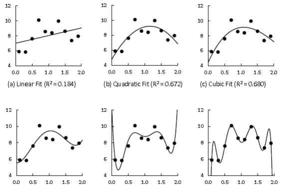
Experimenting to Find Optima

Outline of Part VI

Response Surface Methods–A Package of Statistical Design and Analysis Tools

1. Design and collection of data to fit an equation to approximate the relationship between factors and responses
2. Regression analysis to fit a model to describe the data
3. Examination of the fitted relationship through graphical and numerical techniques

Power Series Models to Approximate Relationships



(a) Linear Fit ($R^2=0.184$) (b) Quadratic Fit ($R^2=0.672$) (c) Cubic Fit ($R^2=0.680$)

(d) Forth Order Fit ($R^2=0.723$) (e) Sixth Order Fit ($R^2=0.881$) (f) Eigth Order Fit ($R^2=0.999$)

Second Order Taylor Series Expansion

10.2.1 Empirical Quadratic Model

$$y = f(x_1, x_2) + \epsilon \tag{10.1}$$

$$\begin{aligned} f(x_1,x_2) \approx f(x_{10},x_{20}) &+ (x_1 - x_{10})\frac{\partial f(x_1,x_2)}{\partial x_1}\Big|_{x_1=x_{10},x_2=x_{20}} \\ &+ (x_2 - x_{20})\frac{\partial f(x_1,x_2)}{\partial x_2}\Big|_{x_1=x_{10},x_2=x_{20}} \\ &+ \frac{(x_1 - x_{10})^2}{2}\frac{\partial^2 f(x_1,x_2)}{\partial x_1^2}\Big|_{x_1=x_{10},x_2=x_{20}} \\ &+ \frac{(x_2 - x_{20})^2}{2}\frac{\partial^2 f(x_1,x_2)}{\partial x_2^2}\Big|_{x_1=x_{10},x_2=x_{20}} \\ &+ \frac{(x_1 - x_{10})(x_2 - x_{20})}{2}\frac{\partial^2 f(x_1,x_2)}{\partial x_1 \partial x_2}\Big|_{x_1=x_{10},x_2=x_{20}} \end{aligned}$$

Results – The General Quadratic Model

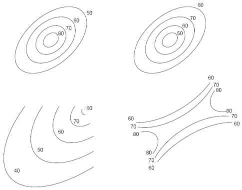
$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{11} x_1^2 + \beta_{22} x_2^2 + \beta_{12} x_1 x_2 + \epsilon \tag{10.3}$$

where $\beta_1 = \frac{\partial f(x_1,x_2)}{\partial x_1}\Big|_{x_1=x_{10},x_2=x_{20}}$ etc. If the region of interest is of moderate

$$y = \beta_0 + \sum_{i=1}^{k} \beta_i x_i + \sum_{i=1}^{k} \beta_{ii} x_i^2 + \sum_{i<j}^{k} \sum^{k} \beta_{ij} x_i x_j + \epsilon, \tag{10.4}$$

Possible Quadratic Surfaces

Figure 10.1 *Surfaces That Can Be Described by General Quadratic Equation*

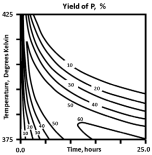


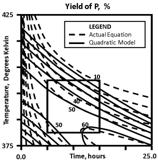
Quadratic Models as Approximations

$[P] = [R]_0 \frac{k_1}{k_1 - k_2}\{\exp(-k_1 t) - \exp(-k_2 t)\}.$

If k_1 and k_2 can be given as functions of temperature by the Arrhenius expressions:

$k_1 = 0.5 \exp[-10{,}000\ (1/T - 1/400)]$ and
$k_2 = 0.2 \exp[-12{,}500\ (1/T - 1/400)]$,





Optimization
Introduction
The Quadratic Response Surface Model
Design Criteria
Standard Designs for Second Order Models
Non-standard Designs
Fitting the Response Surface Model
Determining Optimum Conditions
Split-Plot Response Surface Designs
Screening to Optimization

Matrix Representation of the Quadratic Model

10.2.2 Design Considerations

Quadratic Model $\quad \mathbf{y} = \mathbf{x}\mathbf{b} + \mathbf{x}'\mathbf{B}\mathbf{x} + \epsilon$

where $\mathbf{x}' = (1, x_1, x_2, \ldots, x_k)$, $\mathbf{b}' = (\beta_0, \beta_1, \ldots, \beta_k)$

$$\mathbf{B} = \begin{pmatrix} \beta_{11} & \beta_{12}/2 & \cdots & \beta_{1k}/2 \\ & \beta_{22} & \cdots & \beta_{2k}/2 \\ & & \ddots & \\ & & & \beta_{kk} \end{pmatrix}$$

John Lawson FTC Short Course - Design and Analysis of Experiments with R

Design Consideration for the Linear Model

Linear Model $\mathbf{y} = \mathbf{x}\mathbf{b}$

- the design points are chosen to minimize the variance of the fitted coefficients $\hat{\mathbf{b}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$, $\sigma^2(\mathbf{X}'\mathbf{X})^{-1}$
- design points should be chosen such that $(\mathbf{X}'\mathbf{X})$ matrix is diagonal like the 2^k 2^{k-p} designs diagonal elements of $(\mathbf{X}'\mathbf{X})^{-1}$ minimized

Design Consideration for the Quadratic Model

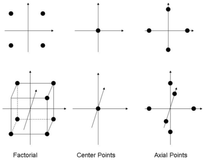
$$Var[\hat{y}(\mathbf{x})] = \sigma^2\mathbf{x}'(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}$$

- Goal is to equalize the variance of a predicted response over the region of interest
- Rotatable Design–variance of a predicted value is only a function of the distance from design center
- Uniform Precision Design–variance of predicted value is near equal within radius of one in coded factor units

Central Composite Designs

10.3.1 Central Composite Design

Figure 10.2 *Central Composite Design in Two and Three Dimensions*



UP Property of Central Composite Designs

Central Composite Design

run	x_1	x_2	
1	-1	-1	Factorial Portion
2	1	-1	
3	-1	1	
4	1	1	
5	0	0	Center Points
6	-α	0	Axial Portion
7	α	0	
8	0	-α	
9	0	α	

By choosing the distance from the origin to the axial points (α in coded units) equal to $\sqrt[4]{F}$ where F is the number of points in the factorial portion of the design, a central composite design will be rotatable. By choosing the correct number of center points the central composite design will have the uniform precision property.

Example of a Central Composite Design

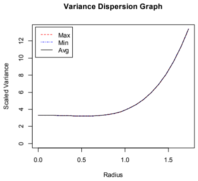
Table 10.1 *Central Composite Design in Coded and Actual Units for Cement Workability Experiment*

run	x_1	x_2	x_3	Water/cement	Black Liq.	SNF	y
1	-1	-1	-1	0.330	0.120	0.080	109.5
2	1	-1	-1	0.350	0.120	0.080	120.0
3	-1	1	-1	0.330	0.180	0.080	110.5
4	1	1	-1	0.350	0.180	0.080	124.5
5	-1	-1	1	0.330	0.120	0.120	117.0
6	1	-1	1	0.350	0.120	0.120	130.0
7	-1	1	1	0.330	0.180	0.120	121.0
8	1	1	1	0.350	0.180	0.120	132.0
9	0	0	0	0.340	0.150	0.100	117.0
10	0	0	0	0.340	0.150	0.100	117.0
11	0	0	0	0.340	0.150	0.100	115.0
12	-1.68	0	0	0.323	0.150	0.100	109.5
13	1.68	0	0	0.357	0.150	0.100	132.0
14	0	-1.68	0	0.340	0.100	0.100	120.0
15	0	1.68	0	0.340	0.200	0.100	121.0
16	0	0	-1.68	0.340	0.150	0.066	115.0
17	0	0	1.68	0.340	0.150	0.134	127.0
18	0	0	0	0.340	0.150	0.100	116.0
19	0	0	0	0.340	0.150	0.100	117.0
20	0	0	0	0.340	0.150	0.100	117.0

$(actual\ level - center\ value)/(half\ range)$ $\pm 1.68 = \sqrt[4]{8}$

Variance Dispersion Graph Shows UP Characteristic

```
> library(daewr)
> data(cement)
> des<-cement[, 2:4]
> library(Vdgraph)
> Vdgraph(des)
number of design points= 20
number of factors= 3
          Radius   Maximum   Minimum   Average
 [1,] 0.00000000  3.326805  3.326805  3.326805
 [2,] 0.08660254  3.320828  3.320828  3.320828
 [3,] 0.17320508  3.303837  3.303837  3.303837
 [4,] 0.25980762  3.278640  3.278640  3.278640
 [5,] 0.34641016  3.249923  3.249923  3.249923
 [6,] 0.43301270  3.224241  3.224241  3.224241
 [7,] 0.51961524  3.210026  3.210026  3.210026
 [8,] 0.60621778  3.217583  3.217583  3.217583
 [9,] 0.69282032  3.259089  3.259089  3.259089
[10,] 0.77942286  3.348596  3.348596  3.348596
[11,] 0.86602540  3.502029  3.502029  3.502029
[12,] 0.95262794  3.737186  3.737186  3.737186
[13,] 1.03923048  4.073740  4.073740  4.073740
[14,] 1.12583302  4.533236  4.533236  4.533236
[15,] 1.21243557  5.139093  5.139093  5.139093
[16,] 1.29903811  5.916603  5.916603  5.916603
[17,] 1.38564065  6.892934  6.892934  6.892934
[18,] 1.47224319  8.097125  8.097125  8.097125
[19,] 1.55884573  9.560089  9.560089  9.560089
```



Creating a Central Composite Design in R

```
> library(rsm)
> rotd <- ccd(3, n0 = c(4,2), alpha = "rotatable", randomize = FALSE)
> rotd
   run.order std.order  x1.as.is  x2.as.is  x3.as.is Block
1          1         1 -1.000000 -1.000000 -1.000000     1
2          2         2  1.000000 -1.000000 -1.000000     1
3          3         3 -1.000000  1.000000 -1.000000     1
4          4         4  1.000000  1.000000 -1.000000     1
5          5         5 -1.000000 -1.000000  1.000000     1
6          6         6  1.000000 -1.000000  1.000000     1
7          7         7 -1.000000  1.000000  1.000000     1
8          8         8  1.000000  1.000000  1.000000     1
9          9         9  0.000000  0.000000  0.000000     1
10        10        10  0.000000  0.000000  0.000000     1
11        11        11  0.000000  0.000000  0.000000     1
12        12        12  0.000000  0.000000  0.000000     1
13         1         1 -1.681793  0.000000  0.000000     2
14         2         2  1.681793  0.000000  0.000000     2
15         3         3  0.000000 -1.681793  0.000000     2
16         4         4  0.000000  1.681793  0.000000     2
17         5         5  0.000000  0.000000 -1.681793     2
18         6         6  0.000000  0.000000  1.681793     2
19         7         7  0.000000  0.000000  0.000000     2
20         8         8  0.000000  0.000000  0.000000     2
```


Optimization

Introduction
The Quadratic Response Surface Model
Design Criteria
Standard Designs for Second Order Models
Non-standard Designs
Fitting the Response Surface Model
Determining Optimum Conditions
Split-Plot Response Surface Designs
Screening to Optimization

Creating a Central Composite Design in R

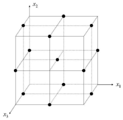
```
> library(rsm)
> ccd.up<-ccd(y~x1+x2+x3,n0=c(4,2),alph="rotatable",coding=list(x1~(Temp-150)/10,
+ x2~(Press-50)/5,x3~(Rate-4)/1),randomize=FALSE)
> head(ccd.up)
  run.order std.order Temp Press Rate  y Block
1         1         1  140    45    3 NA     1
2         2         2  160    45    3 NA     1
3         3         3  140    55    3 NA     1
4         4         4  160    55    3 NA     1
5         5         5  140    45    5 NA     1
6         6         6  160    45    5 NA     1

Data are stored in coded form using these coding formulas ...
x1 ~ (Temp - 150)/10
x2 ~ (Press - 50)/5
x3 ~ (Rate - 4)/1
```

Three Level Box-Behnken Designs

10.3.2 Box-Behnken Design

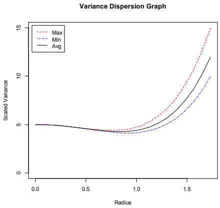
Table 10.2 *Box-Behnken Design in Three Factors*

run	x_1	x_2	x_3
1	-1	-1	0
2	1	-1	0
3	-1	1	0
4	1	1	0
5	-1	0	-1
6	1	0	-1
7	-1	0	1
8	1	0	1
9	0	-1	-1
10	0	1	-1
11	0	-1	1
12	0	1	1
13	0	0	0
14	0	0	0
15	0	0	0



Creating a Box-Behnken Design in R

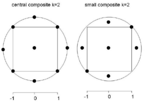
```
> # create design with rsm
> library(rsm)
> bbd3 <- bbd(3,randomize=FALSE,n0=3)
> library(Vdgraph)
> Vdgraph(bbd3[ , 3:5])
number of design points= 15
number of factors= 3
          Radius   Maximum   Minimum   Average
 [1,] 0.00000000  5.000000  5.000000  5.000000
 [2,] 0.08660254  4.984477  4.984445  4.984458
 [3,] 0.17320508  4.939125  4.938625  4.938825
 [4,] 0.25980762  4.867602  4.865070  4.866083
 [5,] 0.34641016  4.776000  4.768000  4.771200
 [6,] 0.43301270  4.672852  4.653320  4.661133
 [7,] 0.51961524  4.569125  4.528625  4.544825
 [8,] 0.60621778  4.478227  4.403195  4.433208
 [9,] 0.69282032  4.416000  4.288000  4.339200
[10,] 0.77942286  4.400727  4.195695  4.277708
[11,] 0.86602540  4.453125  4.140625  4.265625
[12,] 0.95262794  4.596352  4.138820  4.321833
[13,] 1.03923048  4.856000  4.208000  4.467200
[14,] 1.12583302  5.260109  4.367570  4.724583
[15,] 1.21243557  5.839134  4.638625  5.118825
[16,] 1.29903811  6.625977  5.043945  5.676758
[17,] 1.38564065  7.656000  5.608000  6.427200
[18,] 1.47224319  8.966977  6.356945  7.400958
[19,] 1.55884573 10.599125  7.318625  8.630825
[20,] 1.64544827 12.595102  8.522570 10.151583
[21,] 1.73205081 15.000000 10.000000 12.000000
```



Small Composite Designs

10.3.3 Small Composite Design

Figure 10.6 *Graphical Comparison of CCD and Small Composite (with $I = AB$) for k=2*



Hybrid Designs

10.3.4 Hybrid Design

Roquemore (1976) developed hybrid designs that require even fewer runs than the small composite designs. These designs were constructed by making a central composite design in $k-1$ factors and adding a kth factor so that the $\mathbf{X}'\mathbf{X}$ has certain properties and the design is near rotatable.

Table 10.4 *Roquemore 310 Design*

run	x_1	x_2	x_3
1	0	0	1.2906
2	0	0	-0.1360
3	-1	-1	0.6386
4	1	-1	0.6386
5	-1	1	0.6386
6	1	1	0.6386
7	1.736	0	-0.9273
8	-1.736	0	-0.9273
9	0	1.736	-0.9273
10	0	-1.736	-0.9273

Minimal Run Response Surface Designs Available in R package Vdgraph

Small Composite Designs:

Data Frame Name	Description
SCDDL5	Draper and Lin's Design for 5-factors
SCDH2	Hartley's Design for 2-factors
SCDH3	Hartley's Design for 3-factors
SCDH4	Hartley's Design for 4-factors
SCDH5	Hartley's Design for 5-factors
SCDH6	Hartley's Design for 6-factors

Hexagonal Design:

Data Frame Name	Description
Hex2	Hexagonal Design in 2-factors

Data Frame Name	Description
D310	Roquemore's hybrid design D310
D311A	Roquemore's hybrid design D311A
D311B	Roquemore's hybrid design D311B
D416A	Roquemore's hybrid design D416A
D416B	Roquemore's hybrid design D416B
D416C	Roquemore's hybrid design D416C
D628A	Roquemore's hybrid design D628A

Comparing Two Designs with Vdgraph

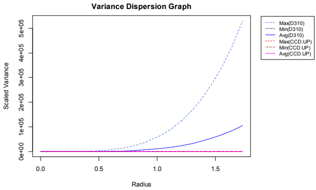
```
> library(rsm)
> ccd.up<-ccd(y~x1+x2+x3,n0=c(4,2),alph="rotatable",coding=list(x1~(Temp-150)/10,
+ x2~(Press-50)/5,x3~(Rate-4)/1),randomize=FALSE)
> head(ccd.up)
  run.order std.order Temp Press Rate  y Block
1         1         1  140    45    3 NA     1
2         2         2  160    45    3 NA     1
3         3         3  140    55    3 NA     1
4         4         4  160    55    3 NA     1
5         5         5  140    45    5 NA     1
6         6         6  160    45    5 NA     1

Data are stored in coded form using these coding formulas ...
x1 ~ (Temp - 150)/10
x2 ~ (Press - 50)/5
x3 ~ (Rate - 4)/1
```

Comparing Two Designs with Vdgraph

```
> library(Vdgraph)
> data(D310)
> D310
        x1     x2      x3
1   0.0000  0.000  1.2906
2   0.0000  0.000 -0.1360
3  -1.0000 -1.000  0.6386
4   1.0000 -1.000  0.6386
5  -1.0000  1.000  0.6386
6   1.0000  1.000  0.6386
7   1.7636  0.000 -0.9273
8  -1.7636  0.000 -0.9273
9   0.0000  1.736 -0.9273
10 0.0000 -1.736 -0.9273
> des<-transform(D310,Temp=10*x1+150, Press=5*x2+50,Rate=x3+4)
> des
        x1     x2      x3    Temp Press   Rate
1   0.0000  0.000  1.2906 150.000 50.00 5.2906
2   0.0000  0.000 -0.1360 150.000 50.00 3.8640
3  -1.0000 -1.000  0.6386 140.000 45.00 4.6386
4   1.0000 -1.000  0.6386 160.000 45.00 4.6386
5  -1.0000  1.000  0.6386 140.000 55.00 4.6386
6   1.0000  1.000  0.6386 160.000 55.00 4.6386
7   1.7636  0.000 -0.9273 167.636 50.00 3.0727
8  -1.7636  0.000 -0.9273 132.364 50.00 3.0727
9   0.0000  1.736 -0.9273 150.000 58.68 3.0727
10  0.0000 -1.736 -0.9273 150.000 41.32 3.0727
```


Comparing Two Designs with Vdgraph

```
> Compare2Vdg(des[, 4:6],ccd.up[, 3:5],"D310","CCD.UP")
```



Standard Designs Inappropriate in Some Situations

10.5 Non-Standard Response Surface Designs

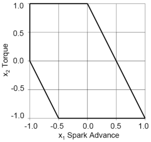
Some design situations do not lend themselves to the use of standard response surface designs

1. Region of experimentation is irregularly shaped
2. Not all combinations of factor levels are feasible
3. There is a nonstandard linear or nonlinear model

Irregular Design Regions

Example 1 – Irregularly shaped region

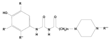
Figure 10.11 *Experimental Region for Engine Experiment*



Finite Number of Possible Design Points

Example 2 – Finite number of candidate points

Figure 10.12 *General Structure of Hydroxyphenylureas*



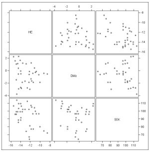


Table 10.5 *Library of Substituted Hydroxyphenylurea Compounds*

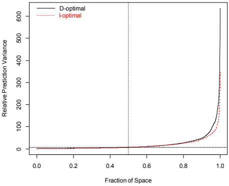
Compound	R	R′	R″	R‴	HE	DMz	S0K
1	H	H	H	CH_3	-12.221	-0.162	64.138
2	H	H	H	CH_2Ph	-14.015	-0.068	88.547
3	H	H	H	Ph	-14.502	0.372	85.567
4	H	H	H	$2CH_3OC_6H_4$	-14.893	1.035	96.053
5	H	OCH_3	H	CH_3	-12.855	1.091	74.124
6	H	OCH_3	H	CH_2Ph	-14.628	1.115	99.002
7	H	OCH_3	H	Ph	-15.123	1.554	96.053
8	H	OCH_3	H	$2CH_3OC_6H_4$	-15.492	2.221	106.607
9	H	OC_2H_5	H	CH_3	-11.813	1.219	77.02
10	H	OC_2H_5	H	CH_2Ph	-13.593	1.188	101.978
11	H	OC_2H_5	H	Ph	-14.088	1.621	99.002
12	CH_3	OC_2H_5	H	$2CH_3OC_6H_4$	-14.46	2.266	109.535
13	CH_3	H	CH_3	CH_3	-8.519	-0.56	71.949
14	CH_3	H	CH_3	CH_2Ph	-10.287	-0.675	96.6
15	CH_3	H	CH_3	Ph	-10.798	-0.134	96.62
16	CH_3	H	CH_3	$2CH_3OC_6H_4$	-11.167	0.418	104.047
17	H	H	H	CH_3	-12.245	-0.609	67.054
18	H	H	H	CH_2Ph	-13.98	-0.518	91.546
19	H	H	H	Ph	-14.491	-0.561	88.547
20	H	H	H	$2CH_3OC_6H_4$	-14.888	-1.478	99.002
21	H	OCH_3	H	CH_3	-11.414	-1.888	77.02
22	H	OCH_3	H	CH_2Ph	-13.121	-1.692	101.978
23	H	OCH_3	H	Ph	-13.66	-1.893	99.002
24	H	OCH_3	H	$2CH_3OC_6H_4$	-14.012	-2.714	109.535
25	H	OC_2H_5	H	CH_3	-10.029	-1.891	79.942
26	H	OC_2H_5	H	CH_2Ph	-11.74	-1.652	104.977
27	H	OC_2H_5	H	Ph	-12.329	-1.902	101.978
28	OCH_3	OC_2H_5	H	$2CH_3OC_6H_4$	-12.637	-2.762	112.492
29	OCH_3	OCH_3	H	CH_3	-12.118	-2.994	81.106
30	OCH_3	OCH_3	H	CH_2Ph	-13.892	-2.845	106.299
31	OCH_3	OCH_3	H	Ph	-14.456	-2.926	103.23
32	OCH_3	OCH_3	H	$2CH_3OC_6H_4$	-14.804	-3.78	113.856
33	CH_3	H	CH_3	CH_3	-9.209	-0.423	74.871
34	CH_3	H	CH_3	CH_2Ph	-10.97	-0.302	99.603
35	CH_3	H	CH_3	Ph	-11.488	-0.453	96.6
36	CH_3	H	CH_3	$2CH_3OC_6H_4$	-11.868	-1.322	107.01

Create the Design with optFederov function in AlgDesign

```
> library(daewr)
> data(qsar)
> library(AlgDesign)
> desgn1<-optFederov(~quad(.),data=qsar,nTrials=15,center=TRUE,
+                    criterion="D",nRepeats=40)
> desgn2<-optFederov(~quad(.),data=qsar,nTrials=15,center=TRUE,
+                    criterion="I",nRepeats=40)
> desgn2$design
   Compound      HE    DMz     S0K
1         1 -12.221 -0.162  64.138
4         4 -14.893  1.035  96.053
9         9 -11.813  1.219  77.020
12       12 -14.460  2.266 109.535
13       13  -8.519 -0.560  71.949
14       14 -10.287 -0.675  96.600
16       16 -11.167  0.418 104.047
19       19 -14.491 -0.561  88.547
22       22 -13.121 -1.692 101.978
28       28 -12.637 -2.762 112.492
29       29 -12.118 -2.994  81.106
32       32 -14.804 -3.780 113.856
33       33  -9.209 -0.423  74.871
34       34 -10.970 -0.302  99.603
36       36 -11.868 -1.322 107.010
```

Compare the D-Optimal and I-Optimal Designs for the Quadratic Model

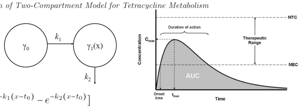
```
> library(Vdgraph)
> Compare2FDS(desgn1$design, desgn2$design, "D-optimal", "I-optimal", mod=2)
```



Known Non-Linear Model

Example 3 – Nonlinear model

Figure 10.14 *Diagram of Two-Compartment Model for Tetracycline Metabolism*



$$y = \gamma_1(x) = \gamma_0[e^{-k_1(x-t_0)} - e^{-k_2(x-t_0)}]$$

$$\begin{aligned} f(x, \gamma_0, k_1, k_2, t_0) = f(x, \gamma_0^*, k_1^*, k_2^*, t_0^*) &+ (\gamma_0 - \gamma_0^*)\left(\frac{\partial f}{\partial \gamma_0}\right)\Big|_{\gamma_0=\gamma_0^*} \\ &+ (k_1 - k_1^*)\left(\frac{\partial f}{\partial k_1}\right)\Big|_{k_1=k_1^*} \\ &+ (k_2 - k_2^*)\left(\frac{\partial f}{\partial k_2}\right)\Big|_{k_2=k_2^*} \\ &+ (t_0 - t_0^*)\left(\frac{\partial f}{\partial t_0}\right)\Big|_{t_0=t_0^*} \end{aligned}$$

Design Strategy

For the compartment model in Equation (10.7)

$$\frac{\partial f}{\partial \gamma_0} = e^{-k_1(x-t_0)} - e^{-k_2(x-t_0)}$$

$$\frac{\partial f}{\partial k_1} = -\gamma_0(x-t_0)e^{-k_1(x-t_0)}$$

$$\frac{\partial f}{\partial k_2} = -\gamma_0(x-t_0)e^{-k_2(x-t_0)}$$

$$\frac{\partial f}{\partial t_0} = \gamma_0 k_1 e^{-k_1(x-t_0)} - \gamma_0 k_2 e^{-k_2(x-t_0)}$$

The strategy is to create a grid of candidates in the independent variable x, calculate the values of each of the four partial derivatives using initial guesses of the parameter values at each candidate point, and then use the `optFederov` function in the `AlgDesign` package to select a D-optimal subset of the grid.

Create the Design in R

```
> k1 <- .15; k2 <- .72; gamma0 <- 2.65; t0 <- 0.41
> x <- c(seq(1:25))
> dfdk1 <- c(rep(0, 25))
> dfdk2 <- c(rep(0, 25))
> dfdgamma0 <- c(rep(0, 25))
> dfdt0 <- c(rep(0, 25))
> for (i in 1:25) {
+   dfdk1[i] <- -1 * gamma0 * exp(-k1 * (x[i] - t0)) *(x[i] - t0)
+   dfdk2[i] <-gamma0 * exp(-k2 * (x[i] - t0)) * (x[i] - t0)
+   dfdgamma0[i] <- exp(-k1 * (x[i] - t0)) - exp( -k2 * ( x[i] - t0))
+   dfdt0[i] <- gamma0 * exp(-k1 * (x[i] - t0)) * k1 - gamma0 *
+     exp(-k2 * (x[i] - t0)) * k2; }
> grid <- data.frame(x, dfdk1, dfdk2, dfdgamma0, dfdt0)
> library(AlgDesign)
> desgn2<-optFederov(~-1+dfdk1+dfdk2+dfdgamma0+dfdt0,data=grid,nTrials=4,center=TRUE,
+ criterion="D",nRepeats=20)
> desgn2$design
    x     dfdk1        dfdk2  dfdgamma0        dfdt0
1   1 -1.431076 1.022374e+00 0.26140256 -0.883809267
2   2 -3.319432 1.341105e+00 0.46952112 -0.294138728
5   5 -6.110079 4.464802e-01 0.46562245  0.129639675
25 25 -1.629706 1.333237e-06 0.02500947  0.009941233
```

Central Composite Design–Cement Grout

Table 10.1 *Central Composite Design in Coded and Actual Units for Cement Workability Experiment*

run	x_1	x_2	x_3	Water/cement	Black Liq.	SNF	y
1	-1	-1	-1	0.330	0.120	0.080	109.5
2	1	-1	-1	0.350	0.120	0.080	120.0
3	-1	1	-1	0.330	0.180	0.080	110.5
4	1	1	-1	0.350	0.180	0.080	124.5
5	-1	-1	1	0.330	0.120	0.120	117.0
6	1	-1	1	0.350	0.120	0.120	130.0
7	-1	1	1	0.330	0.180	0.120	121.0
8	1	1	1	0.350	0.180	0.120	132.0
9	0	0	0	0.340	0.150	0.100	117.0
10	0	0	0	0.340	0.150	0.100	117.0
11	0	0	0	0.340	0.150	0.100	115.0
12	-1.68	0	0	0.323	0.150	0.100	109.5
13	1.68	0	0	0.357	0.150	0.100	132.0
14	0	-1.68	0	0.340	0.100	0.100	120.0
15	0	1.68	0	0.340	0.200	0.100	121.0
16	0	0	-1.68	0.340	0.150	0.066	115.0
17	0	0	1.68	0.340	0.150	0.134	127.0
18	0	0	0	0.340	0.150	0.100	116.0
19	0	0	0	0.340	0.150	0.100	117.0
20	0	0	0	0.340	0.150	0.100	117.0

$(actual\ level - center\ value)/(half\ range)$ $\qquad \pm 1.68 = \sqrt[4]{8}$

Central Composite Design–Cement Grout

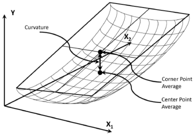
```
> library(daewr)
> data(cement)
> cement
      Block    WatCem     BlackL        SNF     y
C1.1      1 0.3300000 0.12000000 0.08000000 109.5
C1.2      1 0.3500000 0.12000000 0.08000000 117.0
C1.3      1 0.3300000 0.18000000 0.08000000 110.5
C1.4      1 0.3500000 0.18000000 0.08000000 121.0
C1.5      1 0.3300000 0.12000000 0.12000000 120.0
C1.6      1 0.3500000 0.12000000 0.12000000 130.0
C1.7      1 0.3300000 0.18000000 0.12000000 124.0
C1.8      1 0.3500000 0.18000000 0.12000000 132.0
C1.9      1 0.3400000 0.15000000 0.10000000 117.0
C1.10     1 0.3400000 0.15000000 0.10000000 117.0
C1.11     1 0.3400000 0.15000000 0.10000000 115.0
S2.1      2 0.3231821 0.15000000 0.10000000 109.5
S2.2      2 0.3568179 0.15000000 0.10000000 132.0
S2.3      2 0.3400000 0.09954622 0.10000000 120.0
S2.4      2 0.3400000 0.20045378 0.10000000 121.0
S2.5      2 0.3400000 0.15000000 0.06636414 115.0
S2.6      2 0.3400000 0.15000000 0.13363586 127.0
S2.7      2 0.3400000 0.15000000 0.10000000 116.0
S2.8      2 0.3400000 0.15000000 0.10000000 117.0
S2.9      2 0.3400000 0.15000000 0.10000000 117.0

Data are stored in coded form using these coding formulas ...
x1 ~ (WatCem - 0.34)/0.01
x2 ~ (BlackL - 0.15)/0.03
x3 ~ (SNF - 0.1)/0.02
```

Factorial plus centerpoints (C1.1–C1.11)

Axial points plus centerpoints (S2.1–S2.9)

Fit Linear Model–Block 1

```
> library(rsm)
> grout.lin <- rsm(y ~ SO(x1, x2, x3),data = cement, subset = (Block == 1))
Warning message:
In rsm(y ~ SO(x1, x2, x3), data = cement, subset = (Block == 1)) :
  Some coefficients are aliased - cannot use 'rsm' methods.
  Returning an 'lm' object.
> anova(grout.lin)
Analysis of Variance Table

Response: y
                Df Sum Sq Mean Sq F value   Pr(>F)
FO(x1, x2, x3)   3 465.13 155.042 80.3094 0.002307 **
TWI(x1, x2, x3)  3   0.25   0.083  0.0432 0.985889
PQ(x1, x2, x3)   1  37.88  37.879 19.6207 0.021377 *
Residuals        3   5.79   1.931
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
>
```



Fit Quadratic Model–All Data

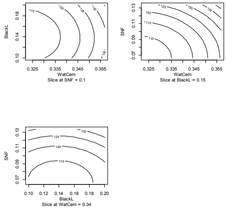
```
> library(daewr)
> data(cement)
> grout.quad <- rsm(y ~ Block + SO(x1,x2,x3), data = cement)
> summary(grout.quad)

Call:
rsm(formula = y ~ Block + SO(x1, x2, x3), data = cement)

               Estimate  Std. Error  t value  Pr(>|t|)
(Intercept)  1.1628e+02  1.0691e+00 108.7658 2.383e-15 ***
Block2       4.4393e-01  1.0203e+00   0.4351   0.67375
x1           5.4068e+00  6.1057e-01   8.8553 9.746e-06 ***
x2           9.2860e-01  6.1057e-01   1.5209   0.16262
x3           4.9925e+00  6.1057e-01   8.1767 1.858e-05 ***
x1:x2        1.2500e-01  7.9775e-01   0.1567   0.87895
x1:x3       -1.3443e-14  7.9775e-01   0.0000   1.00000
x2:x3        1.2500e-01  7.9775e-01   0.1567   0.87895
x1^2         1.4135e+00  5.9582e-01   2.3723   0.04175 *
x2^2         1.3251e+00  5.9582e-01   2.2240   0.05322 .
x3^2         1.5019e+00  5.9582e-01   2.5207   0.03273 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Multiple R-squared:  0.9473,    Adjusted R-squared:  0.8887
F-statistic: 16.17 on 10 and 9 DF,  p-value: 0.0001414
```

Fit Quadratic Model–All Data

```
Analysis of Variance Table

Response: y
               Df Sum Sq Mean Sq F value    Pr(>F)
Block           1   0.00   0.003  0.0006   0.98068
FO(x1, x2, x3)  3 751.41 250.471 49.1962 6.607e-06
TWI(x1, x2, x3) 3   0.25   0.083  0.0164   0.99693
PQ(x1, x2, x3)  3  71.45  23.817  4.6779   0.03106
Residuals       9  45.82   5.091
Lack of fit     5  42.49   8.498 10.1972   0.02149
Pure error      4   3.33   0.833
```

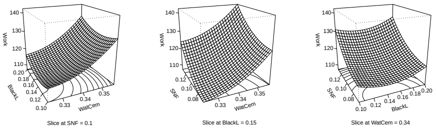

Contour Plots of Fitted Surface

```
> library(rsm)
> contour(grout.quad, ~ x1+x2+x3)
```



Perspective Plots of Fitted Surface

```
> par(mfrow=c(1,3))
> persp(grout.quad, ~ x1+x2+x3, zlab="Work", contours=list(z="bottom"))
```



Optimization
Introduction
The Quadratic Response Surface Model
Design Criteria
Standard Designs for Second Order Models
Non-standard Designs
Fitting the Response Surface Model
Determining Optimum Conditions
Split-Plot Response Surface Designs
Screening to Optimization

Cannonical Analysis

10.7.2 Canonical Analysis

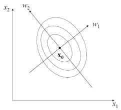
$\mathbf{y} = \mathbf{x}\mathbf{b} + \mathbf{x}'\mathbf{B}\mathbf{x} + \epsilon$ where $\mathbf{x}' = (1, x_1, x_2, \ldots, x_k)$, $\mathbf{b}' = (\beta_0, \beta_1, \ldots, \beta_k)$

$$\mathbf{B} = \begin{pmatrix} \beta_{11} & \beta_{12}/2 & \cdots & \beta_{1k}/2 \\ & \beta_{22} & \cdots & \beta_{2k}/2 \\ & & \ddots & \\ & & & \beta_{kk} \end{pmatrix}$$

Stationary point $\mathbf{x}_0 = -\hat{\mathbf{B}}^{-1}\hat{\mathbf{b}}/2$

Maximum? Minimum? or Saddlepoint?

Figure 10.18 *Representation of Canonical System with Translated Origin and Rotated Axis*



John Lawson
FTC Short Course - Design and Analysis of Experiments with R

Cannonical Analysis

```
Stationary point of response surface:
        x1         x2         x3 
-1.9045158 -0.1825251 -1.6544845 

Stationary point in original units:
    WatCem     BlackL        SNF 
0.32095484 0.14452425 0.06691031 

Eigenanalysis:
$values
[1] 1.525478 1.436349 1.278634

$vectors
        [,1]       [,2]       [,3]
x1 0.1934409  0.8924556  0.4075580
x2 0.3466186  0.3264506 -0.8793666
x3 0.9178432 -0.3113726  0.2461928
```


Ridge Analysis

10.7.3 Ridge Analysis

$$\text{maximum or minimum of } \mathbf{y} = \mathbf{x}\mathbf{b} + \mathbf{x}'\mathbf{B}\mathbf{x}$$

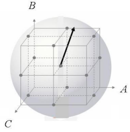
$$\text{subject to} \quad \mathbf{x}'\mathbf{x} = R^2$$

The solution is obtained in a reverse order using Lagrange multipliers. The resulting optimal coordinates are found to be the solution to the equation

$$(\mathbf{B} - \mu\mathbf{I_k})\mathbf{x} = -\mathbf{b}/2. \qquad (10.12)$$

Ridge Analysis

Figure 10.19 *Path of Maximum Ridge Response Through Experimental Region*

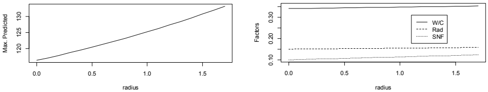


Calculations with rsm package

```
> ridge<-steepest(grout.quad, dist=seq(0, 1.7, by=.1),descent=FALSE)
Path of steepest ascent from ridge analysis:
> ridge
   dist    x1    x2    x3 |  WatCem  BlackL     SNF |    yhat
1   0.0 0.000 0.000 0.000 | 0.34000 0.15000 0.10000 | 116.280
2   0.1 0.073 0.013 0.067 | 0.34073 0.15039 0.10134 | 117.036
3   0.2 0.145 0.026 0.135 | 0.34145 0.15078 0.10270 | 117.821
4   0.3 0.218 0.039 0.203 | 0.34218 0.15117 0.10406 | 118.641
5   0.4 0.290 0.053 0.270 | 0.34290 0.15159 0.10540 | 119.481
6   0.5 0.362 0.067 0.338 | 0.34362 0.15201 0.10676 | 120.355
7   0.6 0.434 0.082 0.406 | 0.34434 0.15246 0.10812 | 121.261
8   0.7 0.505 0.096 0.475 | 0.34505 0.15288 0.10950 | 122.194
9   0.8 0.577 0.112 0.543 | 0.34577 0.15336 0.11086 | 123.160
10  0.9 0.648 0.127 0.611 | 0.34648 0.15381 0.11222 | 124.147
11  1.0 0.719 0.143 0.680 | 0.34719 0.15429 0.11360 | 125.172
12  1.1 0.790 0.159 0.749 | 0.34790 0.15477 0.11498 | 126.227
13  1.2 0.861 0.176 0.818 | 0.34861 0.15528 0.11636 | 127.313
14  1.3 0.931 0.192 0.887 | 0.34931 0.15576 0.11774 | 128.419
15  1.4 1.001 0.209 0.956 | 0.35001 0.15627 0.11912 | 129.557
16  1.5 1.071 0.227 1.025 | 0.35071 0.15681 0.12050 | 130.725
17  1.6 1.141 0.244 1.095 | 0.35141 0.15732 0.12190 | 131.930
18  1.7 1.211 0.262 1.164 | 0.35211 0.15786 0.12328 | 133.158
```

Plotting the Ridge Trace with R

```
> par (mfrow=c(2,1))
> leg.txt<-c("W/C","Rad","SNF")
> plot(ridge$dist,ridge$yhat, type="l",xlab="radius",ylab="Max. Predicted")
> plot(ridge$dist,seq(.10,.355,by=.015), type="n", xlab="radius", ylab="Factors")
> lines(ridge$dist,ridge$WatCem,lty=1)
> lines(ridge$dist,ridge$BlackL,lty=2)
> lines(ridge$dist,ridge$SNF,lty=3)
> legend(1.1,.31,leg.txt,lty=c(1,2,3))
```



Split-Plot Response Surface Designs

Table 10.9 *Data for Cake Baking Experiment*

Oven run	x_1	x_2	y
1	-1	-1	2.7
1	-1	1	2.5
1	-1	0	2.7
2	1	-1	2.9
2	1	1	1.3
2	1	0	2.2
3	0	-1	3.7
3	0	1	2.9
4	0	0	2.9
4	0	0	2.8
4	0	0	2.9

replicate blocks with the same setting for the whole plot factor allow estimation of σ_w^2 → (oven runs 3–4)

whole plot factor is constant within blocks (x_1)

Fitting the Model with lme4 package

```
> library(lme4)
Loading required package: Matrix
Loading required package: Rcpp
> library(daewr)
from 'package:lme4':
    cake
> data(cake)
> cake
   Ovenrun x1 x2   y x1sq x2sq
1        1 -1 -1 2.7    1    1
2        1 -1  1 2.5    1    1
3        1 -1  0 2.7    1    0
4        2  1 -1 2.9    1    1
5        2  1  1 1.3    1    1
6        2  1  0 2.2    1    0
7        3  0 -1 3.7    0    1
8        3  0  1 2.9    0    1
9        4  0  0 2.9    0    0
10       4  0  0 2.8    0    0
11       4  0  0 2.9    0    0
> mmod <- lmer(y ~ x1 +x2 +x1:x2 +x1sq + x2sq +(1|Ovenrun), data=cake)
```


Differences in REML and Least Squares Estimates

Table 10.10 *Comparison of Least Squares and REML Estimates for Split-Plot Response Surface Experiment*

	Least Squares (rsm function)			REMl (lmer function)		
Factor	$\hat{\beta}$	$s_{\hat{\beta}}$	P-value	$\hat{\beta}$	$s_{\hat{\beta}}$	P-value
intercept	2.979	0.1000	<.001	3.1312	0.2667	0.054
x_1	-0.2500	0.0795	0.026	-0.2500	0.2656	0.399
x_2	-0.4333	0.0795	0.003	-0.4333	0.0204	<.001
x_1^2	-0.6974	0.1223	0.002	-0.6835	0.3758	0.143
x_2^2	0.1526	0.1223	0.016	-0.0965	0.0432	0.089
x_1x_2	-0.3500	0.0973	0.268	-0.3500	0.0250	< .001
				$\hat{\sigma}^2_\omega = 0.1402$, $\hat{\sigma}^2 = 0.0025$		

Subplot factor → x_2

Estimation Equivalent Split-Plot RS Design (*EESPRS*)

Factor	Least Squares (`rsm function`) $\hat{\beta}$	$s_{\hat{\beta}}$	P-value	REMl (`lmer function`) $\hat{\beta}$	$s_{\hat{\beta}}$	P-value
intercept	2.979	0.1000	<.001	3.1312	0.2667	0.054
x_1	-0.2500	0.0795	0.026	-0.2500	0.2656	0.399
x_2	-0.4333	0.0795	0.003	-0.4333	0.0204	<.001
x_1^2	-0.6974	0.1223	0.002	-0.6835	0.3758	0.143
x_2^2	0.1526	0.1223	0.016	-0.0965	0.0432	0.089
x_1x_2	-0.3500	0.0973	0.268	-0.3500	0.0250	< .001

$\hat{\sigma}^2_\omega = 0.1402,\ \hat{\sigma}^2 = 0.0025$

$$\mathbf{y} = \mathbf{X}\beta + \epsilon \qquad \mathbf{y} = \mathbf{X}\beta + \omega + \epsilon$$

$$\hat{\beta}_{LS} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} \qquad \hat{\beta}_{REML} = (\mathbf{X}'\mathbf{\Sigma}^{-1}\mathbf{X})^{-1}\mathbf{X}'\mathbf{\Sigma}^{-1}\mathbf{y}$$

EESPRS $\hat{\beta}_{LS} = \hat{\beta}_{REML}$ if $(\mathbf{I}_n - \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}')\mathbf{J}\mathbf{X}) = \mathbf{0}_{n\times p}$

Jones and Goos(2012) *D*-efficient (*EESPRS*)

Table 10.15 **daewr** *Functions for Recalling Jones and Goos's D-Efficient EESPRS Designs*

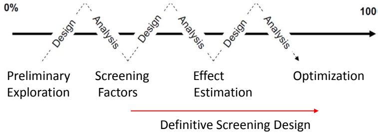
Function Name	Number of Whole-Plot Factors	Number of Split-Plot Factors
EEw1s1	1	1
EEw1s2	1	2
EEw1s3	1	3
EEw2s1	2	1
EEw2s2	2	2
EEw2s3	2	2
EEw3	3	2 or 3

Creating a Design with daewr package

```
> library(daewr)
> EEw2s3()

Catalog of D-efficient Estimation
Equivalent RS
  Designs for (2 wp factors and  3 sp
factors)

   Jones and Goos, JQT(2012) pp. 363-374

Design Name whole plots sub-plots/whole
plot
----------------------------------------
EE21R7WP       7                   3
EE24R8WP       8                   3
EE28R7WP       7                   4
EE32R8WP       8                   4
EE35R7WP       7                   5
EE40R8WP       8                   5
EE42R7WP       7                   6
EE48R8WP       8                   6

==> to retrieve a design type
EE2w3s('EE21R7WP') etc.
```

```
> EEw2s3('EE21R7WP')
   WP w1 w2 s1 s2 s3
1   1  1  1 -1 -1  1
2   1  1  1  1 -1 -1
3   1  1  1 -1  1 -1
4   2  0  1  0  1 -1
5   2  0  1  1 -1  1
6   2  0  1 -1  0  0
7   3 -1  0 -1  1  0
8   3 -1  0  1 -1 -1
9   3 -1  0 -1 -1  1
10  4  1 -1  1 -1  1
11  4  1 -1 -1  1  1
12  4  1 -1  1  1 -1
13  5 -1  1 -1 -1 -1
14  5 -1  1  1  1  0
15  5 -1  1 -1  1  1
16  6  1  0  0  0  1
17  6  1  0  1  1  1
18  6  1  0 -1 -1 -1
19  7 -1 -1  0 -1  0
20  7 -1 -1 -1  0 -1
21  7 -1 -1  1  1  1
```


One-Step Screening to Optimization



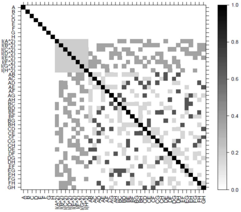
- Jones and Nachtsheim(2011, 2013)
- 3-level designs
- 2k+1 runs for k factors

Creating a Definitive Screening Design with daewr

```
>library(daewr)
> DefScreen(8)
    A  B  C  D  E  F  G  H
1   0 -1  1  1 -1  1  1  1
2   0  1 -1 -1  1 -1 -1 -1
3  -1  0 -1  1  1  1  1 -1
4   1  0  1 -1 -1 -1 -1  1
5  -1 -1  0  1  1 -1 -1  1
6   1  1  0 -1 -1  1  1 -1
7   1 -1  1  0  1  1 -1 -1
8  -1  1 -1  0 -1 -1  1  1
9  -1 -1  1 -1  0 -1  1 -1
10  1  1 -1  1  0  1 -1  1
11  1 -1 -1 -1  1  0  1  1
12 -1  1  1  1 -1  0 -1 -1
13 -1  1  1 -1  1  1  0  1
14  1 -1 -1  1 -1 -1  0 -1
15  1  1  1  1  1 -1  1  0
16 -1 -1 -1 -1 -1  1 -1  0
17  0  0  0  0  0  0  0  0
```


Definitive Screening Designs Are Model Robust

Figure 6.17 *Color Map of 17-Run DSD for 8 Quantitative Factors*



Example of a Definitive Screening Design

Table 13.2 *Factors in the Definitive Screening Experiments of TiO_2 Synthesis*

Label	Factor
A	Speed of H_2O addition
B	Amount of H_2O
C	Drying Time
D	Drying Temperature
E	Calcination Ramp
F	Calcination Temperature
G	Calcination Time
H	Dopant Amount

Analysis using ihstep, fstep in daewr package

```
> des<-DefScreen(8)
> pd<-c(5.35,4.4,12.91,3.79,4.15,14.05,11.4,4.29,3.56,11.4,10.09,5.9,9.54,4.53,3.919,
+ 8.1,5.35)
> trm<-ihstep(pd,des)

Call:
lm(formula = y ~ (.), data = d1)

Residuals:
    Min      1Q  Median      3Q     Max
-5.0201 -0.8301  0.0814  1.0299  3.6799

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   7.2194     0.5140  14.045 4.89e-10 ***
F             3.1508     0.5664   5.563 5.43e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.119 on 15 degrees of freedom
Multiple R-squared:  0.6735,    Adjusted R-squared:  0.6518
F-statistic: 30.94 on 1 and 15 DF,  p-value: 5.429e-05
```

Analysis using ihstep, fstep in daewr package

```
> trm<- fhstep(pd, des, trm)

Call:
lm(formula = y ~ (.), data = d2)

Residuals:
    Min      1Q  Median      3Q     Max
-2.8341 -1.0214 -0.2049  0.5194  2.8378

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   5.0333     1.0345   4.865 0.000309 ***
F             3.1508     0.4789   6.579 1.77e-05 ***
A             0.7664     0.4789   1.600 0.133553
I.A.2.        2.6545     1.1400   2.328 0.036668 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.792 on 13 degrees of freedom
Multiple R-squared:  0.7977,    Adjusted R-squared:  0.751
F-statistic: 17.09 on 3 and 13 DF,  p-value: 8.501e-05
```


Optimization
Introduction
The Quadratic Response Surface Model
Design Criteria
Standard Designs for Second Order Models
Non-standard Designs
Fitting the Response Surface Model
Determining Optimum Conditions
Split-Plot Response Surface Designs
Screening to Optimization

Analysis using ihstep, fstep in daewr package

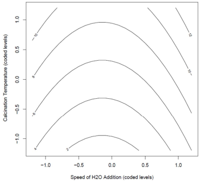
```
> trm <-fhstep(pd, des, trm)

Call:
lm(formula = y ~ (.), data = d2)

Residuals:
    Min      1Q  Median      3Q     Max
-2.8480 -0.6376  0.3167  0.6709  2.4451

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   5.0333     0.9280   5.424 0.000154 ***
F             3.1508     0.4296   7.335 9.04e-06 ***
A             0.7664     0.4296   1.784 0.099715 .
I.A.2.        2.6545     1.0226   2.596 0.023407 *
C            -0.8758     0.4296  -2.039 0.064137 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.607 on 12 degrees of freedom
Multiple R-squared:  0.8498,    Adjusted R-squared:  0.7997
F-statistic: 16.97 on 4 and 12 DF,  p-value: 7.013e-05
```

John Lawson FTC Short Course - Design and Analysis of Experiments with R

Final Results

Pore Diameter = $5.0333 + 0.7664x_1 - 0.8758x_2 + 3.1508x_3 + 2.6545x_1^2$

Figure 13.5 *Contour Plot of Pore Diameter with Drying Time Fixed at Mid-Level*



Recommendations for DSD (Jones)

- Add two dummy factors to create a design with 2k+4 runs for k factors
- Add replicate center points
- Analyze by first fitting the model that includes linear and quadratic main effects only (this leaves at least 4 df for error)
- Eliminate insignificant terms and fit the full quadratic model to the remaining terms