

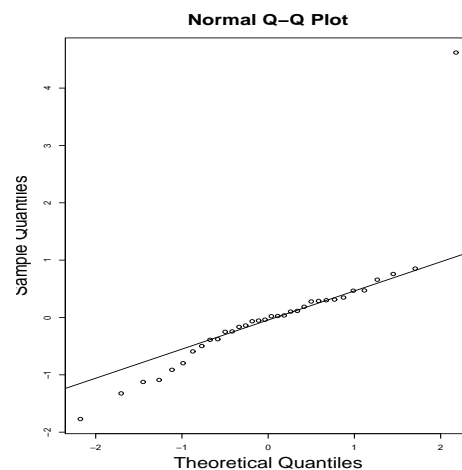
1. Use data set `particle.txt`.

```
(a) > particle <- read.table("hw05.data1.txt", header=T)
> full <- lm(y~temp+time+I(temp^2)+I(time^2)+temp*time, data=particle)
> summary(full)

              Estimate Std. Error t value Pr(>|t|)
(Intercept)  76.79531     1.20649   63.652 < 2e-16 ***
temp         -2.99044     0.16754  -17.850 < 2e-16 ***
time          3.88684     0.40767   9.534 2.73e-10 ***
I(temp^2)    -0.65031     0.05987  -10.862 1.50e-11 ***
I(time^2)    -1.57500     0.36336   -4.335 0.000170 ***
temp*time    -0.55236     0.10716   -5.155 1.82e-05 ***

Residual standard error: 3.554 on 28 degrees of freedom
Multiple R-Squared: 0.9548,    Adjusted R-squared: 0.9467
F-statistic: 118.3 on 5 and 28 DF,  p-value: < 2.2e-16

> library(MASS)
> qqnorm(stdres(full))
```



```
(b) > reduced <- lm(y~temp+time,data=particle)
> anova(reduced,full)
Analysis of Variance Table
Model 1: y ~ temp + time
Model 2: y ~ temp + time + I(temp^2) + I(time^2) + temp * time
  Res.Df  RSS Df Sum of Sq    F    Pr(>F)
1      31 2653.90
2       28  353.66  3   2300.24 60.705 2.259e-12 ***
```

We reject the null hypothesis, $p\text{-value} < 0.0001$. Therefore, a quadratic curvature in response (as a function of x 's) appears to be statistically detectable.

(c) $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2\hat{\beta}_3 & \hat{\beta}_5 \\ \hat{\beta}_5 & 2\hat{\beta}_4 \end{bmatrix}^{-1} \begin{bmatrix} -\hat{\beta}_1 \\ -\hat{\beta}_2 \end{bmatrix} = \begin{bmatrix} -3.050 \\ 1.769 \end{bmatrix}$ is where we have a local maximum. [We should check that it is a maximum and that it is the absolute maximum.]

- Using functions that are built into R.

```
> max.x <- data.frame(-3.050440,1.768824)
> names(max.x) <- c("temp","time")
> predict(full,max.x, interval="confidence",level=0.90)
      fit      lwr      upr
[1,] 84.79397 82.59441 86.99352
> predict(full,max.x, interval="prediction",level=0.90)
      fit      lwr      upr
[1,] 84.79397 78.36052 91.22742
```

- Using matrix calculations.

```
x0 <- c(1, max.x$temp, max.x$time, max.x$temp^2, max.x$time^2, max.x$temp*max.x$time)
y0 <- sum(full$coef*x0)
X <- cbind(rep(1,dim(particle)[1]), particle$temp, particle$time,
           particle$temp^2, particle$time^2, particle$temp*particle$time)
MSE <- anova(full)$Mean[length(anova(full)$Mean)]
cXXc <- x0%*%ginv(t(X)%*%X)%*%x0
ll <- y0 - qt(0.95,full$df.residual)*sqrt(MSE*cXXc) # lower and upper confidence limits
ul <- y0 + qt(0.95,full$df.residual)*sqrt(MSE*cXXc)
ll <- y0 - qt(0.95,full$df.residual)*sqrt(MSE*(1+cXXc)) # lower and upper prediction limits
ul <- y0 + qt(0.95,full$df.residual)*sqrt(MSE*(1+cXXc))
```

2. There is weak evidence of “lack of fit”, p-value = 0.055.

- Using functions that are built into R.

```
> cells <- lm(y~as.factor(temp)*as.factor(time),data=particle)
> anova(cells,full)
Analysis of Variance Table
Model 1: y ~ as.factor(temp) * as.factor(time)
Model 2: y ~ temp + time + I(temp^2) + I(time^2) + temp * time
  Res.Df    RSS Df Sum of Sq    F Pr(>F)
1      4   10.44
2     28  353.66 -24  -343.22 5.4792 0.05478
```

- Using matrix calculations.

```
> Y <- particle$y
> F.ratio <- ((t(Y)%*(project(Xstar)-project(X))%*%Y)/(30-6))/
             ((t(Y)%*(diag(rep(1,34))-project(Xstar))%*%Y)/(34-30))
> p.value <- 1-pf(F.ratio,30-6,34-30)
```

3. Use scenario of Problem 14.17 of Rencher.

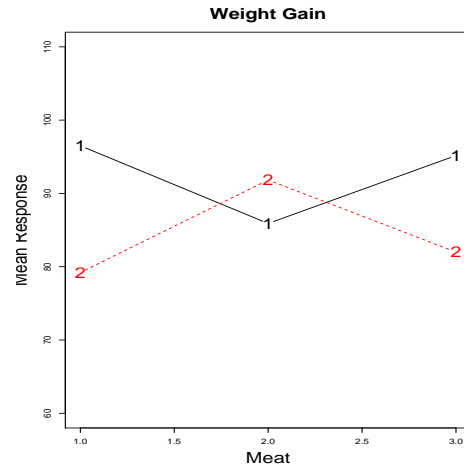
(a) > d

```
      y protein meat
1    73      1    1
2   102      1    1
3   118      1    1
...
46   97      2    3
47  106      2    3
```

(b) > attach(d) # The database "d" is searched by R when evaluating a variable

```
> means <- tapply(y,list(meat,protein),mean)
> means
      1      2
1 96.50000 79.20000
2 85.90000 91.83333
3 95.16667 82.00000
```

```
(c) > matplot(c(1,3),c(60,110),type="n",xlab="Meat",ylab="Mean Response",main="Weight Gain")
> matlines(x.axis,means,type="b",cex=2)
```



```
(d) options(contrasts=c("contr.sum","contr.sum"))
lm.out1 <- lm(y~protein*meat,data=d)
> summary.aov(lm.out1,ssType=1)
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
protein	1	829.1	829.1	4.0296	0.05133 .
meat	2	13.8	6.9	0.0336	0.96702
protein:meat	2	1195.7	597.9	2.9057	0.06605 .
Residuals	41	8436.2	205.8		

```
> lm.out2 <- lm(y~meat*protein,data=d)
> summary.aov(lm.out2,ssType=1)
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
meat	2	16.4	8.2	0.0399	0.96097
protein	1	826.5	826.5	4.0170	0.05168 .
meat:protein	2	1195.7	597.9	2.9057	0.06605 .
Residuals	41	8436.2	205.8		

(e) Using matrix calculations.

i. Using the sum restrictions. Numerical results are shown in part (f).

```
Xeffects <- model.matrix(lm.out1)
```

```
# Type I
```

```
ssA <- t(d$y)%*(project(Xeffects[,1:2])-project(Xeffects[,1]))%*d$y # R(a|mu)
```

```
ssB <- t(d$y)%*(project(Xeffects[,1:4])-project(Xeffects[,1:2]))%*d$y # R(b|mu,a)
```

```
ssAB <- t(d$y)%*(project(Xeffects)-project(Xeffects[,1:4]))%*d$y # R(ab|mu,a,b)
```

```
# Type II
```

```
ssA <- t(d$y)%*(project(Xeffects[,1:4])-project(Xeffects[,c(1,3,4)]))%*d$y # R(a|mu,b)
```

```
ssB <- t(d$y)%*(project(Xeffects[,1:4])-project(Xeffects[,c(1,2)]))%*d$y # R(b|mu,a)
```

```
ssAB <- t(d$y)%*(project(Xeffects)-project(Xeffects[,1:4]))%*d$y # R(ab|mu,a,b)
```

```
Xcells <- matrix(0,47,6)
```

```
Xcells[1:8,]<- matrix(rep(c(1,0,0,0,0,0),8),8,6,byrow=T)
```

```
Xcells[9:18,]<- matrix(rep(c(0,1,0,0,0,0),10),10,6,byrow=T)
```

```
Xcells[19:24,]<- matrix(rep(c(0,0,1,0,0,0),6),6,6,byrow=T)
```

```
Xcells[25:34,]<- matrix(rep(c(0,0,0,1,0,0),10),10,6,byrow=T)
```

```
Xcells[35:40,]<- matrix(rep(c(0,0,0,0,1,0),6),6,6,byrow=T)
```

```
Xcells[41:47,]<- matrix(rep(c(0,0,0,0,0,1),7),7,6,byrow=T)
```

```

# Type III
C <- matrix(c(1/3,1/3,1/3,-1/3,-1/3,-1/3),1,6,byrow=T) # For SSprotein
C <- matrix(c(rep(c(1/3,-1/3,0),2),rep(c(1/3,0,-1/3),2)),2,6,byrow=T) # For SSmeat

CXXC <- C%*%ginv(t(Xcells)%*%Xcells)%*%t(C)
Cb <- C%*%ginv(t(Xcells)%*%Xcells)%*%t(Xcells)%*%d$y
SSH0 <- t(Cb)%*%solve(CXXC)%*%Cb # Gives the value of SS

```

ii. Using the SAS baseline restriction we obtain the same results as with the sum restriction.

```

Xsas <- matrix(NA,47,6)
Xsas[,1] <- rep(1,47)
Xsas[,2] <- c(rep(1,24),rep(0,23))
Xsas[,3] <- c(rep(1,8),rep(0,16),rep(1,10),rep(0,13))
Xsas[,4] <- c(rep(0,8),rep(1,10),rep(0,16),rep(1,6),rep(0,7))
Xsas[,5] <- Xsas[,2]*Xsas[,3]
Xsas[,6] <- Xsas[,2]*Xsas[,4]

```

```

# Type I
ssA <- t(d$y)%*(project(Xsas[,1:2])-project(Xsas[,1]))%*%d$y # R(a|mu)
ssB <- t(d$y)%*(project(Xsas[,1:4])-project(Xsas[,1:2]))%*%d$y # R(b|mu,a)
ssAB <- t(d$y)%*(project(Xsas)-project(Xsas[,1:4]))%*%d$y # R(ab|mu,a,b)

```

```

# Type II
ssA <- t(d$y)%*(project(Xsas[,1:4])-project(Xsas[,c(1,3,4)]))%*%d$y # R(a|mu,b)
ssB <- t(d$y)%*(project(Xsas[,1:4])-project(Xsas[,c(1,2)]))%*%d$y # R(b|mu,a)
ssAB <- t(d$y)%*(project(Xsas)-project(Xsas[,1:4]))%*%d$y # R(ab|mu,a,b)

```

(f) Using function `anova()` for Type I and function `Anova()` from package `car` for Types II and III.

Anova Table (Type I tests)

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
protein	1	829.1	829.1	4.0296	0.05133 .
meat	2	13.8	6.9	0.0336	0.96702
protein:meat	2	1195.7	597.9	2.9057	0.06605 .
Residuals	41	8436.2	205.8		

Anova Table (Type II tests)

	Sum Sq	Df	F value	Pr(>F)
protein	826.5	1	4.0170	0.05168 .
meat	13.8	2	0.0336	0.96702
protein:meat	1195.7	2	2.9057	0.06605 .
Residuals	8436.2	41		

Anova Table (Type III tests)

	Sum Sq	Df	F value	Pr(>F)
(Intercept)	351398	1	1707.8016	< 2e-16 ***
protein	751	1	3.6510	0.06304 .
meat	9	2	0.0221	0.97820
protein:meat	1196	2	2.9057	0.06605 .
Residuals	8436	41		

4. Use fake 2-way factorial data.

(a) Repeat parts (a)-(e) of Problem 3 on these data.

```

i. > fake
      Y A B
1    12 1 1
2    14 1 2

```

```

3 10 1 3
4 12 1 3
5 9 2 1
6 11 2 2
7 12 2 2
8 6 2 3
9 7 2 3
10 10 3 1
11 11 3 2
12 7 3 3

```

```

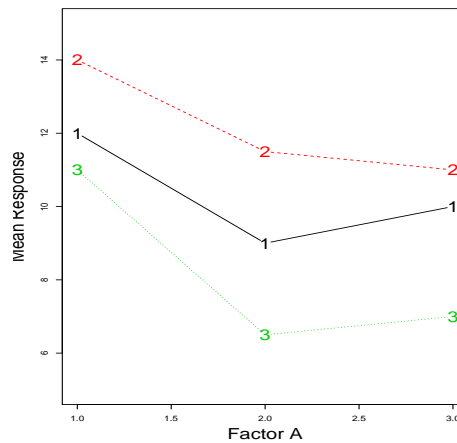
ii. > means <- tapply(fake$Y,list(fake$A,fake$B),mean)
> means
  1  2  3
1 12 14.0 11.0
2 9 11.5 6.5
3 10 11.0 7.0

```

```

iii. > x.axis <- unique(fake$A)
> matplot(c(1,3),c(5,15),type="n",xlab="Factor A",ylab="Mean Response")
> matlines(x.axis,means,type="b")

```



```

iv. > fake$A <- as.factor(fake$A)
> fake$B <- as.factor(fake$B)
> options(contrasts=c("contr.sum", "contr.sum"))
> lm.out1 <- lm(Y~A*B,data=fake)
> summary.aov(lm.out1,ssType=1)

```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
A	2	22.250	11.125	11.1250	0.04095 *
B	2	37.039	18.520	18.5195	0.02051 *
A:B	4	2.628	0.657	0.6569	0.66182
Residuals	3	3.000	1.000		

```

> lm.out2 <- lm(Y~B*A,data=fake)
> summary.aov(lm.out2,ssType=1)

```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
B	2	29.0500	14.5250	14.5250	0.02864 *
A	2	30.2391	15.1195	15.1195	0.02711 *
B:A	4	2.6276	0.6569	0.6569	0.66182
Residuals	3	3.0000	1.0000		

v. Using matrix calculations.

- Using the sum restrictions. Numerical results are shown in part (vi).

```

Xeffects <- model.matrix(lm.out1)
Y <- fake$Y

```

```

# Type I
ssA <- t(Y)%%(project(Xeffects[,1:3])-project(Xeffects[,1]))%*Y # R(a|mu)
ssB <- t(Y)%%(project(Xeffects[,1:5])-project(Xeffects[,1:3]))%*Y # R(b|mu,a)
ssAB <- t(Y)%%(project(Xeffects)-project(Xeffects[,1:5]))%*Y # R(ab|mu,a,b)

# Type II
ssA <- t(Y)%%(project(Xeffects[,1:5])-project(Xeffects[,c(1,4,5)]))%*Y # R(a|mu,b)
ssB <- t(Y)%%(project(Xeffects[,1:5])-project(Xeffects[,c(1,2,3)]))%*Y # R(b|mu,a)
ssAB <- t(Y)%%(project(Xeffects)-project(Xeffects[,1:5]))%*Y # R(ab|mu,a,b)

> Xcells
 1  1  0  0  0  0  0  0  0  0
 2  0  1  0  0  0  0  0  0  0
 3  0  0  1  0  0  0  0  0  0
 4  0  0  1  0  0  0  0  0  0
 5  0  0  0  1  0  0  0  0  0
 6  0  0  0  0  1  0  0  0  0
 7  0  0  0  0  1  0  0  0  0
 8  0  0  0  0  0  1  0  0  0
 9  0  0  0  0  0  1  0  0  0
10  0  0  0  0  0  0  1  0  0
11  0  0  0  0  0  0  0  1  0
12  0  0  0  0  0  0  0  0  1

# Type III
# For SSA
C <- matrix(c(1/3,1/3,1/3,0,0,0,-1/3,-1/3,-1/3,0,0,0,1/3,1/3,1/3,-1/3,-1/3,-1/3),2,9,byrow=T)

# For SSB
C <- matrix(c(rep(c(1/3,0,-1/3),3),rep(c(0,1/3,-1/3),3)),2,9,byrow=T)

CXXC <- C%*%ginv(t(Xcells)%*%Xcells)%*%t(C)
Cb <- C%*%ginv(t(Xcells)%*%Xcells)%*%t(Xcells)%*%Y
SSH0 <- t(Cb)%*%solve(CXXC)%*%Cb # Gives the value of SS

```

- Using the SAS baseline restriction we obtain the same results as with the sum restriction.

```

> Xsas
 1  1  1  0  1  0  1  0  0  0
 2  1  1  0  0  1  0  1  0  0
 3  1  1  0  0  0  0  0  0  0
 4  1  1  0  0  0  0  0  0  0
 5  1  0  1  1  0  0  0  1  0
 6  1  0  1  0  1  0  0  0  1
 7  1  0  1  0  1  0  0  0  1
 8  1  0  1  0  0  0  0  0  0
 9  1  0  1  0  0  0  0  0  0
10  1  0  0  1  0  0  0  0  0
11  1  0  0  0  1  0  0  0  0
12  1  0  0  0  0  0  0  0  0

# Type I
ssA <- t(Y)%%(project(Xsas[,1:3])-project(Xsas[,1]))%*Y # R(a|mu)
ssB <- t(Y)%%(project(Xsas[,1:5])-project(Xsas[,1:3]))%*Y # R(b|mu,a)
ssAB <- t(Y)%%(project(Xsas)-project(Xsas[,1:5]))%*Y # R(ab|mu,a,b)

```

```

# Type II
ssA <- t(Y)%*(project(Xsas[,1:5])-project(Xsas[,c(1,4,5)]))%*Y # R(a|mu,b)
ssB <- t(Y)%*(project(Xsas[,1:5])-project(Xsas[,c(1,2,3)]))%*Y # R(b|mu,a)
ssAB <- t(Y)%*(project(Xsas)-project(Xsas[,1:5]))%*Y # R(ab|mu,a,b)

```

vi. Using function `anova()` for Type I and function `Anova()` from package `car` for Types II and III.

```

> anova(lm.out1)
Anova Table (Type I tests)

```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
A	2	22.250	11.125	11.1250	0.04095 *
B	2	37.039	18.520	18.5195	0.02051 *
A:B	4	2.628	0.657	0.6569	0.66182
Residuals	3	3.000	1.000		

```

Anova Table (Type II tests)

```

	Sum Sq	Df	F value	Pr(>F)
A	30.239	2	15.1195	0.02711 *
B	37.039	2	18.5195	0.02051 *
A:B	2.628	4	0.6569	0.66182
Residuals	3.000	3		

```

Anova Table (Type III tests)

```

	Sum Sq	Df	F value	Pr(>F)
(Intercept)	1128.53	1	1128.5333	5.798e-05 ***
A	25.11	2	12.5541	0.03487 *
B	32.33	2	16.1655	0.02474 *
A:B	2.63	4	0.6569	0.66182
Residuals	3.00	3		

```

(b) Xincomp.full <- Xcells[-c(6,7),-5]
Xincomp.red <- Xeffects[-c(6,7),1:5]
Y.incomp <- Y[-c(6,7)]
full <- lm(Y.incomp~Xincomp.full-1)
reduced <- lm(Y.incomp~Xincomp.red-1)
anova(reduced,full)

```

```

Analysis of Variance Table
Model 1: Y.incomp ~ Xincomp.red - 1
Model 2: Y.incomp ~ Xincomp.full - 1

```

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	5	3.8298				
2	2	2.5000	3	1.3298	0.3546	0.7954

F-ratio = 0.35 with p-value= 0.7954. There is not enough evidence that there is interaction between factors A and B.

```

(c) X <- Xeffects[-c(6,7),1:5]
C <- c(1,0,1,0,1)
mu22 <- C%*%ginv(t(X)%*%X)%*%t(X)%*%Y.incomp
> mu22

```

[,1]	
[1,]	10.08511