Assignment 2

2022-10-24

Question 1

a)

The test statistic for the hypothesis is

$$T = \frac{\hat{\theta}_3 - 2}{\sqrt{\hat{\Sigma}_{3,3}}} = \frac{4-2}{\sqrt{1.1}} = 1.9069252.$$

Since $1.9069252 > t_{97,0.95} = 1.6607146$ so we reject H_0.

b)

Let $\alpha = 0.05$, the 95% confidence interval is

$$\hat{\theta}_3 \pm t_{97,0.975}\sqrt{\hat{\Sigma}_{3,3}} = 4 \pm 1.98 \times 1.04 = [1.94, 6.05].$$

c)

We have the statistic

$$T = \frac{f(0,\hat{\theta})}{\sqrt{\hat{v}_x^T \hat{\Sigma} \hat{v}}} = \frac{0.54}{0.85} = 0.63$$

which is lower than $t_{97,0.975}$ so we can not reject H_0.

d)

The 95% confidence interval is

$$f(0,\hat{\theta}) \pm t_{97,0.975} \times \sqrt{\hat{v}_x^T \hat{\Sigma} \hat{v}} = [-1.14, 2.22].$$

e)

Since the dataset with the funtion $f(x,\theta)$, we can compute the matrix $\hat{V}$ where

$$\hat{V}_{ij} = \partial f(x_i, \hat{\theta}) / \partial \theta_j.$$

Thus, we can compute $(\hat{V}^T \hat{V})^{-1}$ and

$$\hat{\sigma}^2 = \hat{\Sigma}(\hat{V}^T \hat{V})$$

where $\hat{\Sigma}$ is the given estimated covariance matrix.

Question 2

a)

The estimates of θ are $\hat{\theta} = (0.81, -0.44, 1.98, 1.27)$ and $\hat{\sigma}^2 = \frac{S(\hat{\theta})}{n-p}$ where $S(\hat{\theta}) = RSE^2(n-p)$ where RSE is the residual standard error. Thus, we have $\hat{\sigma}^2 = RSE^2 = 0.275$. The residual sum of squares can be recovered from the residual standard error and the degree of freedoms because

$$RSE = \sqrt{\frac{RSS}{n-p}}$$

where p is the number of parameters θ, which in this case, is 4.

b)

We first obtain the estimated covariance matrix or just the values on the diagonal of $\hat{\Sigma}$ using $x_i = \frac{3(i-1)}{n-1}$ with $n = 100$ and the partial derivatives $\partial f(x_i, \hat{\theta})/\partial \theta_j$. Note that the partial derivatives calculations are quite straightforward so they will be omitted. We have $\hat{\Sigma}_{1,1} = 0.114$, $\hat{\Sigma}_{2,2} = 0.028$, $\hat{\Sigma}_{3,3} = 0.0076$, and $\hat{\Sigma}_{4,4} = 0.1104$. Like before, we have the test statistic

$$T = \frac{0.8}{\sqrt{0.114}} = 2.37 > t_{96,1-0.05/2} = 1.984$$

so we reject H_0. The 95% confidence interval is

$$\hat{\theta}_1 \pm t_{96,1-0.05/2}\sqrt{\hat{\Sigma}_{1,1}} = 0.81 \pm 1.984 \times 0.337 = [0.141, 1.478].$$

c)

Likewise, we have the statistic

$$T = \frac{\hat{\theta}_4 - 1}{\sqrt{\hat{\Sigma}_{4,4}}} = 0.81 < t_{96,1-0.05/2}$$

so we can not reject H_0. Let $\alpha = 0.02$ then the 98% confidence interval is

$$\hat{\theta}_2 \pm t_{96,1-0.02/2}\sqrt{\hat{\Sigma}_{2,2}} = -0.44 \pm 2.36 \times 0.028 = [-0.5, -0.37].$$

d)

e)

Let the global model Ω be the full model, i.e, $S(\hat{\theta_q}) = \min_{\theta_q} ||Y - f_\Omega(x, \theta_q)||^2$, where $\theta_q \in \mathbb{R}^q$, and the linear submodel to be $\omega : S(\hat{\theta_p}) = \min_{\theta_p} ||Y - f_\omega(x, \theta_p)||^2$ where $\theta_p \in \mathbb{R}^p$ with $p < q$ such that $f_\omega(x, \theta_p)$ is linear. We can test for H_0 that the linear submodel fits well at significance level α by using the test statistic

$$V = \frac{[S(\hat{\theta_p}) - S(\hat{\theta_q})]/(q-p)}{S(\hat{\theta_q})/(n-q)}.$$

If $V > F_{q-p,n-q,1-\alpha}$ then we reject H_0 that the linear submodel fits well.

Question 3

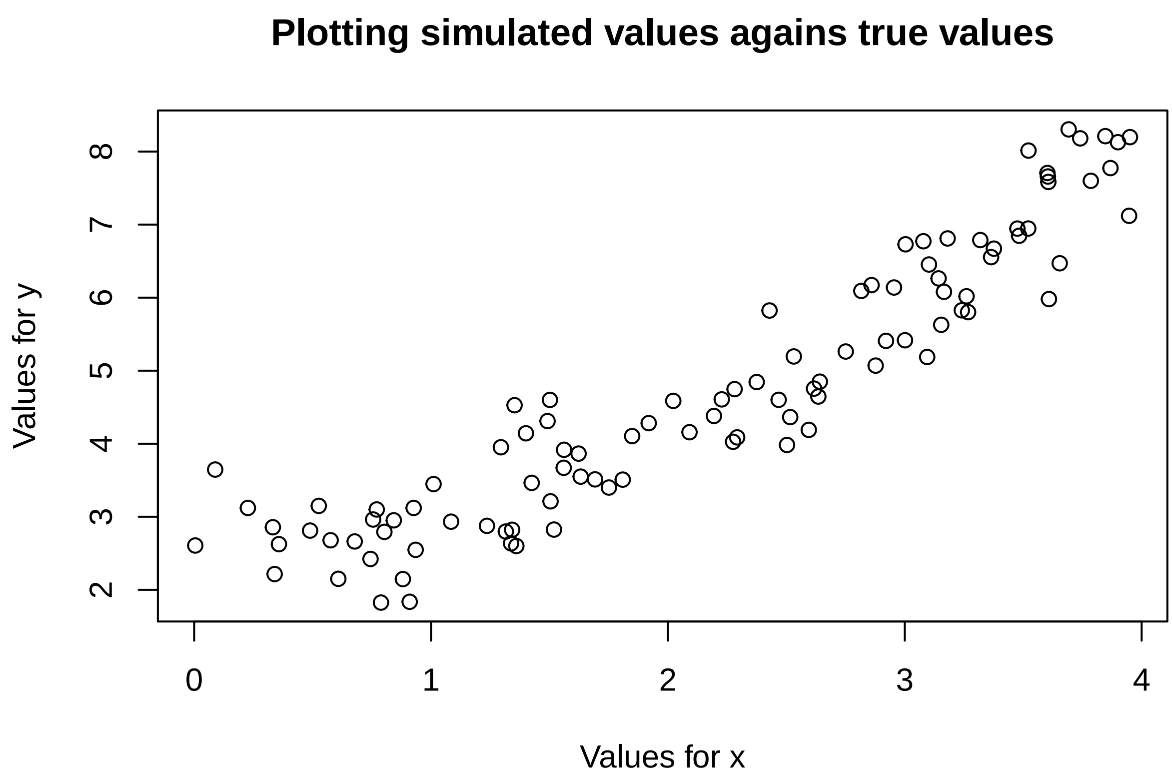
a)

```
N = 100
x = runif(N,0,4)
theta = c(2, 3, 1)
sigma = 0.5 #sigma_sq = 0.25
eps = rnorm(N, 0, sigma)
f <- function(x) {return(theta[1] * x + theta[2] / (theta[3] + 3*x^2))}
ftrue = x # true values
y <- f(x) + eps;
ftrue <- f(x);

form=as.formula(y ~ theta1*x + theta2/(theta3+3*x^2))
dfy = data.frame(y)
dfy['x'] = x
```

In this question, we need to artificially generate our data. We will take $x_i \in [0, 4]$ with $i = 1, \ldots, 100$ and our function $f(x, \theta) = \theta_1 x + \theta_2/(\theta_3 + 3x^2)$. To our simulated response we have added an error term $\epsilon_i \sim N(0, \sigma^2)$ such that $y_i = f(x_i) + \varepsilon_i$. Here, we chose $\theta = (2, 3, 1)$ and $\sigma = 0.5$. The following plot illustrates the dataset.

```
plot(x,y,
     xlab="Values for x",
     ylab="Values for y",
     main='Plotting simulated values agains true values') # plot of dataset
```



Now, we estimate the parameter θ by means of a non-linear model. The model's summary will give us our first insights about the quality of our estimation.

```r
nmodel=nls(form,data=dfy,start=c(theta1=theta[1],theta2=theta[2],theta3=theta[3]))
hat.th=coef(nmodel); hat.th ## the LSE estimates of theta's
```

```
##   theta1   theta2   theta3
## 1.950372 3.483419 1.223956
```

```r
summary(nmodel) # more informative output
```

```
##
## Formula: y ~ theta1 * x + theta2/(theta3 + 3 * x^2)
##
## Parameters:
##        Estimate Std. Error t value Pr(>|t|)
## theta1  1.95037    0.02633  74.063  < 2e-16 ***
## theta2  3.48342    0.72518   4.804 5.68e-06 ***
## theta3  1.22396    0.34054   3.594 0.000514 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5537 on 97 degrees of freedom
##
## [...]
```

Indeed, we can observe that our estimates seem to be quite accurate, but not to the same extent for the three dimensions of θ, yet the estimation for θ_2 seems to be slightly more off with respect to the other two dimesions of θ.

If we see our estimate for the error's variance, we will see that it does give a relatively good estimation (although that obviously depends on the context of the data:

```r
RSS=deviance(nmodel);RSS # the same info in nmodel
```

```
## [1] 29.73797
```

```r
y_var=RSS/(N-2)
y_var
```

```
## [1] 0.3034487
```

Similarly, here we have our covariance matrix using our estimates:

```r
cov.est=vcov(nmodel)
cov.est
```

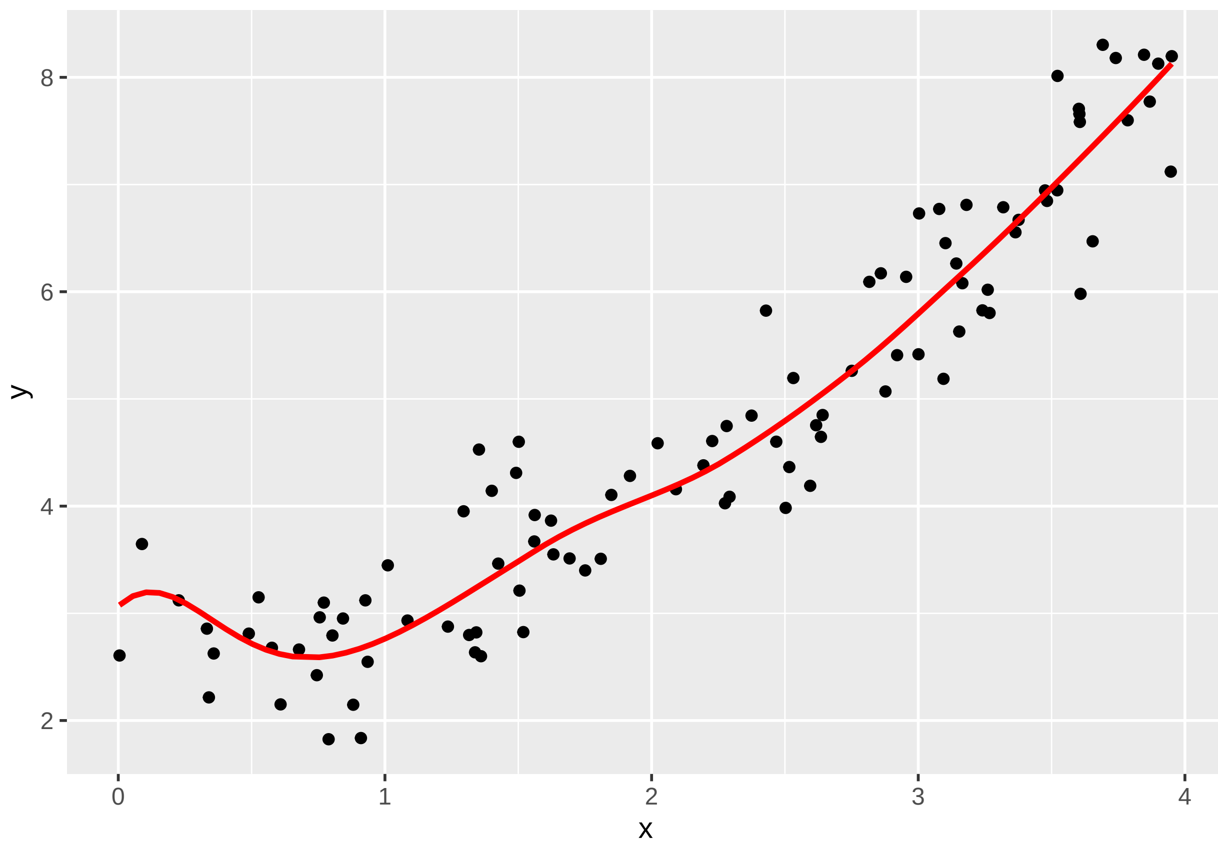
```
##               theta1       theta2       theta3
## theta1  0.0006934823 -0.009584406 -0.004024743
## theta2 -0.0095844061  0.525889361  0.232613942
## theta3 -0.0040247434  0.232613942  0.115969839
```

Finally, we can use our estimate function, i.e. $f(x, \hat{\theta})$, in the plot.

```r
f=function(x,theta){return(theta[1] * x + theta[2] / (theta[3] + 3*x^2))}
ggplot(data=dfy, aes(x=x, y=y)) +
  geom_point() +
  geom_smooth(se=FALSE, formula = y ~ f(x, theta), color="red", fill="lightblue")
```

```
## `geom_smooth()` using method = 'loess'
```

```
## Warning: Removed 1 rows containing missing values (geom_smooth).
```



b)

The 98% level confidence interval for $\theta = (\theta_1, \theta_2, \theta_3)$ can be directly computed by asymptotic normality and will yield the following intervals:

```
lb=numeric(3); ub=numeric(3); # rownames(lb)=names(coef(nmodel))
for(i in 1:3) {lb[i]=coef(nmodel)[i]-qt(0.98,N-length(coef(nmodel)))*sqrt(cov.est[i,i])
      ub[i]=coef(nmodel)[i]+qt(0.98,N-length(coef(nmodel)))*sqrt(cov.est[i,i])}
ci=cbind(lb,ub); rownames(ci)=names(coef(nmodel))
ci
```

```
##                 lb       ub
## theta1 1.8955515 2.005192
## theta2 1.9737826 4.993056
## theta3 0.5150351 1.932877
```

Also, we can compute confidence intervals of the same confidence level by means of the bootstrap method:

```
B=1000 # 1000
par.boot=matrix(NA,B,length(coef(nmodel)))
rownames(par.boot)=paste("B",1:B,sep="")
colnames(par.boot)=names(coef(nmodel))
res.centered=resid(nmodel)-mean(resid(nmodel))

for(b in 1:B){
  # cat("b = ",b,"\n")
  # Bootstrap samples from centered residuals
  res=sample(res.centered,replace=T)
```

```
  # Calculate bootstrap values for the response
  yboot=fitted(nmodel)+res #Y*_1,...Y*_n
  # Fit model using new response and get bootstrap estimates for parameter
  modelBoot=nls(yboot~theta1*x+theta2/(theta3+3*x^2),
              data=data.frame(yboot,x),start=list(theta1=2,theta2=3,theta3=1))
  # Store estimated (by bootstrap) parameters
  par.boot[b,]=coef(modelBoot) #\theta*_1,..., \theta*_B
}

lb.boot=2*coef(nmodel)-apply(par.boot,2,quantile,prob=0.98)
ub.boot=2*coef(nmodel)-apply(par.boot,2,quantile,prob=0.02)
cbind(lb.boot,ub.boot)
```

```
##          lb.boot  ub.boot
## theta1 1.8982393 2.003711
## theta2 1.4765922 4.621463
## theta3 0.1438138 1.748663
```

c)

Let's see the expected value for Y when $x = 3$ in a 98% confidence interval. Note that both the actual value of $f(3, \theta)$ and our estimate are included in the interval. This should not be surprising, yet we are looking at a relatively high confidence level.

```
f3=f(3,coef(nmodel))
f3; f(3,theta)
```

```
##   theta1
## 5.974537
```

```
## [1] 6.107143
```

```
grad<-function(x,theta){rbind(x,
                              1 / (theta[3] + 3 * x^2),
                              -theta[2] / (theta[3] + 3*x^2) / (theta[3] + 3*x^2) )}
gradvec=grad(3,coef(nmodel))
se=sqrt(t(gradvec)%*%vcov(nmodel)%*%gradvec) #0.006986284
lb=f3-qt(0.99,N-length(coef(nmodel)))*se #1.156622
ub=f3+qt(0.99,N-length(coef(nmodel)))*se #1.179728
c(lb,ub)
```

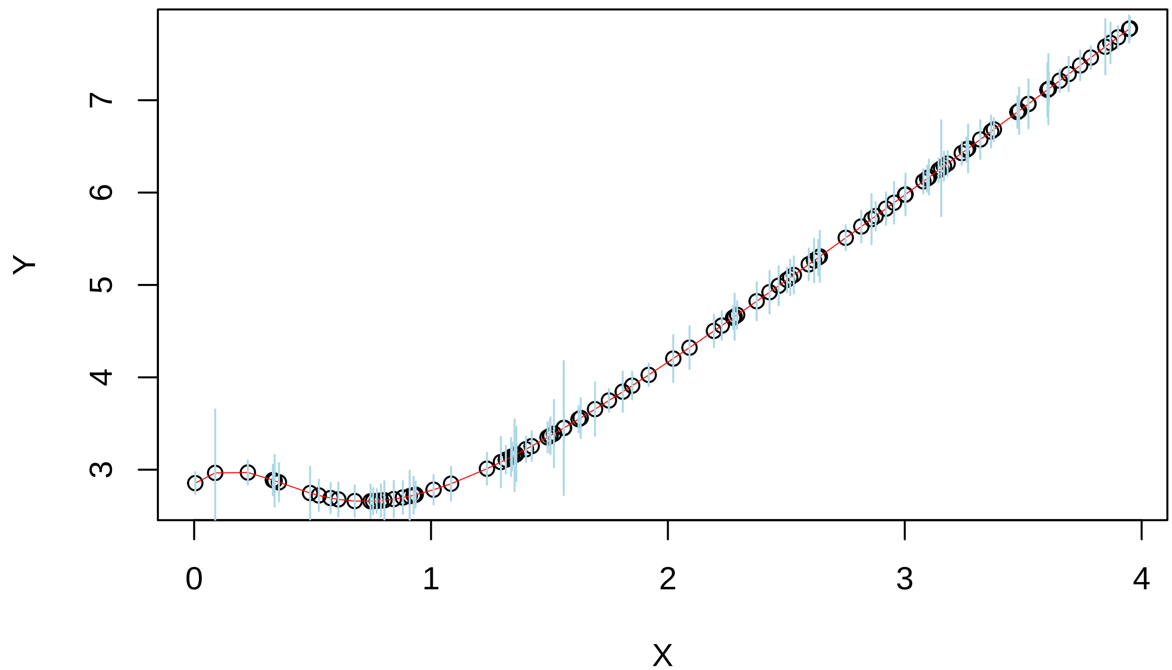
```
## [1] 5.808962 6.140111
```

d)

This can be done for all $x \in [0, 4]$. To gain some insight, we will make use of the plot seen before but this time including the confidence intervals.

```
fe=f(sort(x),coef(nmodel)) # or: fe=predict(nmodel,newdata=data.frame(conc=x))
mygrad=grad(x,coef(nmodel)) # estimated gradients for x's from [0,4]
se<-sqrt(apply(mygrad,2,function(xx) t(xx)%*%vcov(nmodel)%*%xx))
lb<-fe-qt(0.99,N-length(coef(nmodel)))*se
ub<-fe+qt(0.99,N-length(coef(nmodel)))*se

plot(dfy[, 'x'],f(x,coef(nmodel)),xlab="X",ylab="Y"); # data
lines(sort(x),fe,t="l",lwd=0.5,col="red"); # fitted curve
segments(sort(x),lb,sort(x),ub,col="lightblue") # confidence intervals
```



e)

The test, with $\alpha = 0.05$, can be done in the following way: let $U = \hat{\theta_1} - \hat{\theta_2}$, so our null hypothesis will be $H_0 : U = 0$, to contrast with the alternative $H_1 : U \neq 0$. From the way U is defined, we can do the following: $U = \hat{\theta}_1 - \hat{\theta_2} = \hat{\theta_1} - \theta + \theta - \hat{\theta_1} = (\hat{\theta_1} - \theta) - (\hat{\theta_2} - \theta)$.

Note that $(\hat{\theta_1} - \theta)$ and $(\hat{\theta_2} - \theta)$ are, under H_0, both normally distributed with mean 0 and variance Σ, where Σ is the covariance matrix. Therefore, $U \sim N(0, \Sigma)$. Note that $\hat{\Sigma}_{kk} = Var(\hat{\theta_k}) = Var(\hat{\theta_k} - a)$ in the usual situation, doing the same thing for U, we have

$$Var(U) = Var(\hat{\theta_1} - \hat{\theta_2}) = Var(\hat{\theta_1}) + Var(\hat{\theta_1}) - 2Cov(\hat{\theta_1}, \hat{\theta_2}) = \Sigma_{11} + \Sigma_{22} - 2\Sigma_{12}^2$$

.

The statistic $t_U = U/\sqrt{}(Var(U))$ follows a $t - distribution$ with $100 - 3$ degrees of freedom. We calculate the p-value of the statistic.

```
U = coef(nmodel)[1] - coef(nmodel)[2]
t = U / sqrt((cov.est[1,1] + cov.est[2,2] - 2 * cov.est[1,2]^2))
pval = pt(abs(t), df = 97, lower.tail = FALSE)
```

Given that the p-value $= 0.0185842$ is smaller than $\alpha = 0.05$, we have sufficient evidence to reject H_0 for this confidence level.

Question 4

a)

```
stormer <- data.frame(stormer);
smod_lin <- lm(Wt * Time ~ Viscosity + Time, data = stormer);
summary(smod_lin)
```

```
##
## Call:
## lm(formula = Wt * Time ~ Viscosity + Time, data = stormer)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -330.7 -153.8    4.7  170.7  368.3
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 220.1381    82.0080   2.684  0.01426 *
## Viscosity    28.0987     0.5663  49.620  < 2e-16 ***
## Time          2.0818     0.7302   2.851  0.00987 **
## [...]
```

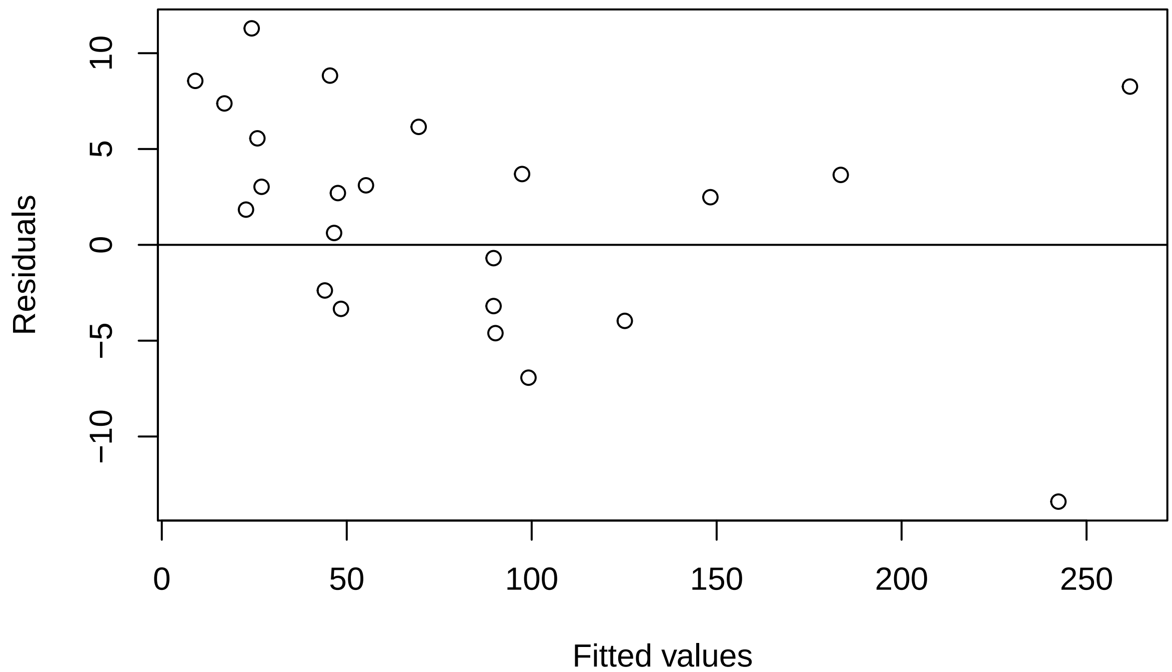
Fitting the linear regression $wT = \theta_1 v + \theta_2 T + (w - \theta_2)\varepsilon$ returns the estimates $\hat{\theta}_1 = 28$ and $\hat{\theta}_2 = 2$ which we will use for the initial values of the non-linear regression.

```
n = nrow(stormer);
p = 2;
smod.nls <- nls(Time ~ (the1 * Viscosity)/(Wt - the2),
               data = stormer,
               start=c(the1=28, the2=2)
               );
RSS=deviance(smod.nls);
smod.evar <- round(RSS/(n - p), 2);
summary(smod.nls)
```

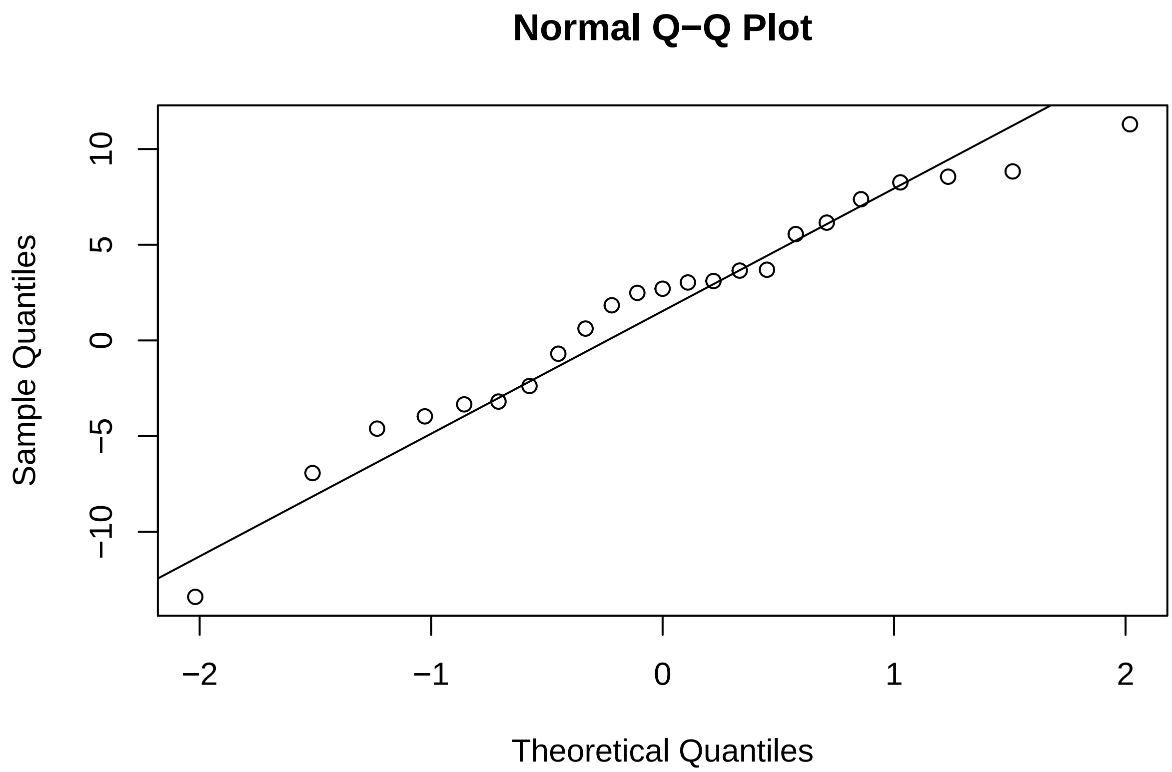
```
##
## Formula: Time ~ (the1 * Viscosity)/(Wt - the2)
##
## Parameters:
##      Estimate Std. Error t value Pr(>|t|)
## the1  29.4013     0.9155  32.114  < 2e-16 ***
## the2   2.2183     0.6655   3.333  0.00316 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [...]
```

Applying non-linear regression, we have the estimates for (θ_1, θ_2) as $(\hat{\theta}_1, \hat{\theta}_2) = (29.4, 2.2)$ which are very close to the inital values.The estimated variance of the error $\hat{\sigma}^2 = 39.29$. To test for the validity of the model's assumptions, we use residual plot and the qq plot, Shapiro-Wilk test to check for the normality of the residual.

```
plot(fitted(smod.nls), resid(smod.nls), xlab="Fitted values", ylab="Residuals");
abline(0, 0)
```




```
qqnorm(resid(smod.nls))
qqline(resid(smod.nls))
```




```
shapiro.test(resid(smod.nls))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(smod.nls)
## W = 0.96402, p-value = 0.5491
```

It can be seen that the pattern of the points are somewhat random so the non-linear model is somewhat good. Since the p-value of the Shapiro-Wilk test is 0.55, which is higher than 0.05, we reject the null hypothesis that the data is not normally distributed. The QQ-plot also confirms the normality assumption.

b)

Calculate the test statistic T using the covariance matrix, we have that $T = 0.3005162 < t_{21,1-0.05/2} = 2.08$ so we can not reject H_0.

c)

```
lb=numeric(2); ub=numeric(2);
for(i in 1:2) {lb[i]=coef(smod.nls)[i]-qt(0.975,n-length(coef(smod.nls)))*sqrt(smod.cov[i,i])
      ub[i]=coef(smod.nls)[i]+qt(0.975,n-length(coef(smod.nls)))*sqrt(smod.cov[i,i])}
ci=cbind(lb,ub); rownames(ci)=names(coef(smod.nls)); ci
```

```
##             lb        ub
## the1 27.497301 31.305213
## the2  0.834246  3.602302
```

The 95% confidence interval for $\hat{\theta}_1$ is $[27.49, 31.3]$ and $\hat{\theta}_2$ is $[0.83, 3.6]$.

d)

```
grad<-function(v,w,the){rbind(v/(w - the[2]), the[1]*v/(w - the[2])^2)};
gradvec <- grad(100, 60, coef(smod.nls));

se=sqrt(t(gradvec)%*%vcov(smod.nls)%*%gradvec);
f <- function(v, w, the) { the[1]*v/(w - the[2]) };
f4 <- f(100, 60, coef(smod.nls))

lb=f4-qt(0.05/2,n-length(coef(smod.nls)), lower.tail=FALSE)*se
ub=f4+qt(0.05/2,n-length(coef(smod.nls)), lower.tail=FALSE)*se

c(lb, ub)
```

```
## [1] 48.65760 53.10902
```

The 95% confidence interval for the expected value of T is therefore $[48.65, 53.1]$.

e)

```
form2 <- as.formula(Time ~ (the1 * Viscosity)/(Wt));
smod.nls2 <- nls(form2, data=stormer, start=c(the1=28));
anova(smod.nls, smod.nls2)
```

```
## Analysis of Variance Table
##
## Model 1: Time ~ (the1 * Viscosity)/(Wt - the2)
## Model 2: Time ~ (the1 * Viscosity)/(Wt)
##   Res.Df Res.Sum Sq Df  Sum Sq F value  Pr(>F)
## 1     21     825.05
## 2     22    1210.38 -1 -385.33  9.8078 0.00504 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Using ANOVA, the reduced model gives a worse fit than the full one since its residual sum of squares is $1210 > 825.05$. We also calculate the V statistic to compare against the F-distribution.

```
SSq <- deviance(smod.nls);
SSp <- deviance(smod.nls2);

n <- length(resid(smod.nls));
q<- length(coef(smod.nls));
p <- length(coef(smod.nls2))

fstat <- ((SSp-SSq)/(q-p))/(SSq/(n-q));
pval <- 1 - pf(fstat, q-p, n-p);
```

The obtained F-statistic is $9.8078075 > F_{1,21,0.95} = 4.32$
and its p-value is $0.004851 < 0.05$ so we reject the null hypothesis H_0 that the smaller model ω is appropriate.

```
AIC(smod.nls)
```

```
## [1] 153.6101
```

```
AIC(smod.nls2)
```

```
## [1] 160.4247
```

The AIC of the full model is 153.6 while the AIC for the smaller one is 160.4. Clearly, $153 < 160$ so the full model fits better.