Statistics Coursework 2018

Ioannis Valasakis[1]

[1]Birkbeck, University of London

December 17, 2018

Question 1

To plot the expression levels of the "Ras-Like Protein Tc4" the Golub dataset is loaded from the respective library (hopach from Bioconductor). To perform computations on the expressions of this gene we need to know its row index.

```
# load the Golub data from bioconductor
source("http://www.bioconductor.org/biocLite.R")
biocLite(c("hopach"))
library(hopach)
data(golub)

# find the row index of tc4
tc4 = grep("Ras-Like Protein Tc4", golub.gnames[, 2],
           ignore.case = TRUE)
golubFactor <- factor(golub.cl,
                      levels = 0:1,
                      labels = c("ALL", "AML"))

# side by side boxplot for AML and ALL
boxplot(golub[tc4,] ~ golubFactor, ylim = c(0, 2))
```

The above R program generates the boxplot in Fig. 2. There is a difference in gene expression values of the Tc4 in ALL versus AML patients as shown in Fig. 2. Testing the data for normality with the Shapiro-Wilk test and a plot to indicate the shape of the distribution in Fig. 1

> Shapiro-Wilk normality test
>
> data: golub[tc4,]
> W = 0.9614, p-value = 0.2108

As the plot shows a normal distribution and the Shapiro-Wilk test has a p-value > 0.05, a normal distribution is assumed and we are using a two-sample t-test further. (Karadimitriou, n.d.)

To test the null hypothesis, we can apply a Welch two-sample t-test to determine whether the means of two gene expressions of populations are statistically different. The null hypothesis is:

$$H_0 : \mu_1 = \mu_2 \tag{1}$$

where

- $\mu 1$ is the mean expression for gene tc4 in ALL patients
- $\mu 2$ is the mean expression for gene tc4 in AML patients

Let's suppose that $\overline{x}$ is the mean of the gene expression data for the group ALL and $\overline{y}$ is the mean of the population AML, and s_2^1 the variance of the first and s_2^2 that of the second, for $\{x_1,$

$\cdots$,x_n} and {$y_1, \cdots, y_m$} the data for ALL and AML respectively. Then the t-statistic can be formulated as:

$$t = \frac{(\overline{x} - \overline{y}) - (\mu_1 - \mu_2)}{\sqrt{s_1^2/n + s_2^2/m}} \tag{2}$$

The decision procedure concerning the null-hypothesis is now entirely similar for the above t-test. Note that the t-value is large if the difference between x and y is significant and the standard deviations s1 and s2 are small. We assume that the distribution of both data sets must be normal, to assess if the gene is differentially expressed between the two groups, with α (significant level) equal to 5% ($\alpha = 0.05$) (Krijnen, n.d.). The test can be run in the same environment as the previous R code by using the library function `t.test()` with the parameter `var.equal = FALSE` which calculates the Welch two-sample test:

```
t.test(golub[tc4, ] ~ golubFactor, var.equal = FALSE)
```

We can directly calculate the p-value which is of interest by:

```
t.test(golub[tc4, ] ~ golubFactor, var.equal = FALSE)$p.value
```

The value is very low (p = 0.0002028185) which is considerably lower than the α (0.05 or 5%) threshold for statistical significance level. Therefore we can reject the null hypothesis that the gene expression values of Tc4 are the same between all the ALL versus AML patients.

Question 2

Create a loop that samples 1000 random samples of size k = 20 from a normal distribution of 'true' mean 10 and true standard deviation 5.

```
nsampl = 1000

# e stores the randomly generated samples
e <- vector("list", nsampl)
sample_mean = rep(NA, nsampl)

for (i in 1:nsampl) {
  e[[i]] <- rnorm(n = 20, mean = 10, sd = 5)
  sample_mean[i] <- mean(e[[i]])
}
```

In order to plot the sample mean values for sample size k = 20, we can use `plot()` with the function `abline()` as shown in the Fig 3.

```
plot(sample_mean, xlab="sample size (N)", ylab="sample mean (\\x)")
abline(h=10, col="orange")
```

The following function calculates the standard deviation of the means of 1000 random samples of user-specifiable sample size k, drawn from a normal distribution of true mean m and true standard deviation s.

```
sd_sample_mean <- function(k, m, s) {
  nsampl = 1000

  # e contains the randomly generated samples
  e <- vector("list", nsampl)
  sample_mean = rep(NA, nsampl)

  for (i in 1:nsampl) {
    e[[i]] <- rnorm(n = k, mean = m, sd = s)
    sample_mean[i] <- mean(e[[i]])
  }

  return(sd(sample_mean))
}
```

The standard deviation of the means, for the specific values of k = 3, k = 100 and m = 10 and s = 5 is:

```
> sd_sample_mean(3,10,5)
[1] 2.881059
> sd_sample_mean(100,10,5)
[1] 0.4869542
```

The theoretical approximation of the Standard Deviation of Error (SDE) for those same values and in relation to the Standard Deviation (SD) can be calculated simply by (Altman and Bland, 2005)

$$SDE = \frac{SD}{\sqrt{(samplesize)}} \tag{3}$$

in our case k is the sample size, therefore, $\text{SDE}_3 = 2.886751$ and $\text{SDE}_{100} = 0.5$ which is pretty close to the above-calculated values.

Question 3

By importing the data of the 20 replicate measurements of the concentration of a protein X for each of the cell lines which are carried out by mass spectrometry:

```
wildtype <-
  c(
    560,
    968,
    3297,
    1200,
    858,
    646,
    992,
    2507,
```

```
    2037,
    546,
    2929,
    1171,
    1389,
    1958,
    3149,
    1165,
    2257,
    2120,
    65,
    1571
  )

knockout <-
  c(
    589,
    232,
    983,
    2597,
    827,
    1363,
    634,
    12,
    643,
    1889,
    2840,
    1291,
    939,
    811,
    3290,
    525,
    90,
    543,
    2400,
    3012
  )
```

If we plot the data, we can observe that they don't fall in the case of a normal distribution. As the distribution of concentrations is skewed, the median could be a better measure of centrality than the mean. The distribution of the concentrations also has heavy tails so the median could be a more efficient estimator of centrality than the mean (Peters, n.d.). Thus to determine if the deletion of the transcription factor changes the concentration of protein X present in the cell, we consider the median value of the concentration. The result of the plot is in Fig. 4. We will also use the percentile bootstrap method (Carpenter and Bithell, 2000) in order to determine the 95% confidence interval.

```
# import the boot library
library(boot)
```

```
w <-
  boot(
    data = wildtype,
    statistic = function(x, i)
      median(wildtype[i]),
    R = 10000
  )

plot(w)

# calculate the 95% confidence
boot.ci(w,
        conf = 0.95,
        type = "perc")
```

The result is:

```
> BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
> Based on 10000 bootstrap replicates

> CALL :
> boot.ci(boot.out = w, conf = 0.95, type = "perc", main = "Histogram of > wildtype")

> Intervals :
> Level     Percentile
> 95%   ( 980, 2078 )
> Calculations and Intervals on Original Scale
```

That gives us 95% certainty that the mean value of the concentration of the protein X is between 980 and 2078. Now we can do the same for the knockout cell line. The plot is in Fig. 5

```
k <-
  boot(
    data = knockout,
    statistic = function(x, i)
      median(knockout[i]),
    R = 10000
  )

plot(k)

# calculate the 95% confidence
boot.ci(k,
        conf = 0.95,
        type = "perc")
```

Which results in:

```
> BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
> Based on 10000 bootstrap replicates

> CALL :
```

```
> boot.ci(boot.out = k, conf = 0.95, type = "perc")

> Intervals :
> Level     Percentile
> 95%   ( 611.5, 1626.0 )
> Calculations and Intervals on Original Scale
```

That means we are 95% confident that the mean concentration of the protein X is between 611.5 and 1626. To calculate the median difference using bootstrap here is a respective function and is pictured in Fig. 6

```
diff.medians <- median(wildtype) - median(knockout)
nboot <- 5000
n <- length(wildtype)
diff.medians.boot <- rep(0, nboot)

for (i in 1:nboot) {
  wildtype.boot <- sample(wildtype, n, replace = T)
  knockout.boot <- sample(knockout, n, replace = T)
  diff.medians.boot[i] <-
    median(wildtype.boot) - median(knockout.boot)
}

hist(diff.medians.boot,
     col = "orange",
     nclass = 20,
     main = "")

title("Bootstrap distribution of difference between medians",
      cex.main = 1.0)

# Compute 95% CLT based Bootstrap confidence intervals
# for difference between medians
ci.lower.sd <- diff.medians - 1.96 * sd(diff.medians.boot)
ci.upper.sd <- diff.medians + 1.96 * sd(diff.medians.boot)
c(ci.lower.sd, ci.upper.sd)

# Compute 95% Boostrap percentile confidence intervals
# for difference between medians
ci.lower.perc <- sort(diff.medians.boot)[0.05 * nboot / 2 + 1]
ci.upper.perc <- sort(diff.medians.boot)[nboot - 0.05 * nboot / 2]
c(ci.lower.perc, ci.upper.perc)
```

Question 4

Here is the data describing the dependence of survival on the mutation of gene X organised in a 2x2 contingency table. Each object is classified according to more than one categorical variable.

The objects are mutant and non-mutant and the variables died and survived.

	died	survived	total
mutant	40	42	82
non-mutant	47	107	154
total	87	149	236

The null hypothesis for the following chi-square test is that the events are not associated, i.e. that the survival is not dependent on the mutation of the gene X.

```
# create the contingency table
t1 <- data.frame(died = c(40, 42), survived = c(47, 107))
rownames(t1)[1] <- "mutant"
rownames(t1)[2] <- "nonmutant"

# perform a chi-square test
chisq <- chisq.test(t1)
```

which results in:

> Pearson's Chi-squared test with Yates' continuity correction
>
> data: t1 X-squared = 6.9019, df = 1, p-value = 0.008611

As the p-value is much smaller than 0.05 our null hypothesis is rejected which means that there is an association between the mutation of the gene to the survival rate of the patients.

We can also extrapolate the expected and observed results:

```
> chisq$observed
          died survived
mutant      40       47
nonmutant   42      107
> round(chisq$expected,0)
          died survived
mutant      30       57
nonmutant   52       97
```

We can perform a 2-sample test for equality of proportions with continuity correction.

```
mut_strength <- matrix(c(47, 40, 107, 42), nrow = 2)
prop.test(mut_strength)
```

Here is the result which shows a 95% confidence interval not overlapping with 0, thus we can predict that there is a difference between the two variables.

```
> prop.test(mut_strength)

    2-sample test for equality of proportions with continuity correction

data:  mut_strength
X-squared = 6.9019, df = 1, p-value = 0.008611
alternative hypothesis: two.sided
95 percent confidence interval:
```

```
 -0.32231661 -0.04290353
sample estimates:
   prop 1    prop 2
0.3051948 0.4878049
```

Question 5

We have the following data:

```
enzyme <-
  c(
    0.114,
    0.510,
    0.722,
    1.276,
    1.928,
    2.150,
    2.238,
    2.732,
    2.758,
    3.015,
    3.616,
    3.951,
    4.281,
    5.315,
    6.693,
    6.964,
    7.056,
    8.162,
    8.216,
    8.410
  )
metabolite <-
  c(
    56.1,
    60.6,
    67.2,
    72.7,
    80.5,
    83.2,
    82.2,
    88.9,
    89.5,
    90.6,
    94.9,
    95.2,
    97.1,
```

```
    96.3,
    77.6,
    71.6,
    69.3,
    37.2,
    36.0,
    26.9
  )
```

We will plot the data in order to check for normal distribution:

```
hist(enzyme, probability=T, main="Histogram of enzyme", col="purple")
lines(density(enzyme),col=2)

hist(metabolite, probability=T, main="Histogram of enzyme", col="yellow")
lines(density(metabolite),col=2)
```

The data for enzyme are in Fig. 7 and for metabolite in Fig. 8

From the graphical representation, it is pretty obvious that the metabolite data is skewed while the enzyme data is more normally distributed.

If we plot their relationship we can see that it is definitely non-linear. See also the Fig. 9

```
y <- metabolite * enzyme
plot(enzyme, y)
```

We can try and fit a curve in the above relation using the polynomial equation.

```
x <- metabolite
y <- enzyme
#fit first degree polynomial equation:
fit  <- lm(y~x)
fit2 <- lm(y~poly(x,2,raw=TRUE))
fit3 <- lm(y~poly(x,3,raw=TRUE))
fit4 <- lm(y~poly(x,4,raw=TRUE))

xx <- seq(0,100, length=100)
plot(x,y,pch=10,ylim=c(0,50))
lines(xx, predict(fit, data.frame(x=xx)), col="red")
lines(xx, predict(fit2, data.frame(x=xx)), col="green")
lines(xx, predict(fit3, data.frame(x=xx)), col="blue")
lines(xx, predict(fit4, data.frame(x=xx)), col="purple")
```

The result can be seen in the Fig. 10

Notes

All the R code in this coursework has been beautified using the tidyverse style guide which is a standard plug-in in the R-Studio suite.

References

Karadimitriou, S.M., n.d. Checking normality for parametric tests in R. https://www.sheffield.ac.uk/polopoly_fs/1.579191!/file/stcp-karadimitriou-normalR.pdf.

Krijnen, W.P., n.d. Applied Statistics for Bioinformatics using R.

Altman, D.G., Bland, J.M., 2005. Standard deviations and standard errors. BMJ 331, 903. doi:10.1136/bmj.331.7521.903

Peters, C.A., n.d. Statistics for Analysis of Experimental Data.

Carpenter, J., Bithell, J., 2000. Bootstrap confidence intervals: when which, what? A practical guide for medical statisticians. Statistics in Medicine 19, 1141–1164. doi:10.1002/(sici)1097-0258(20000515)19:9¡1141::aid-sim479¿3.0.co;2-f

Figure Captions

Figure 1. Are Tc4 data normally distributed?

Figure 2. Expression levels of the Ras-Like Protein Tc4

Figure 3. Sample mean values of true mean 10

Figure 4. Wildtype cell line boost plot

Figure 5. Knockout cell line boost plot

Figure 6. Bootstrap distribution of difference between medians

Figure 7. Enzyme distribution

Figure 8. Metabolite distribution

Figure 9. Enzyme and metabolite non-linearity

Figure 10. Polynomial fit

Figures

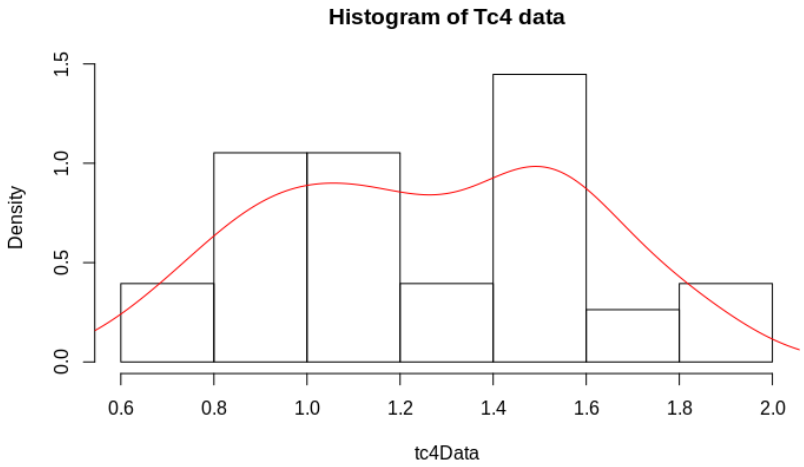

Figure 1: Are Tc4 data normally distributed?

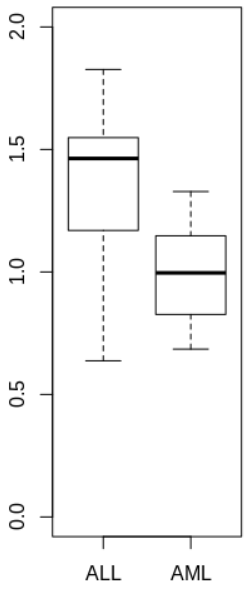

Figure 2: Expression levels of the Ras-Like Protein Tc4

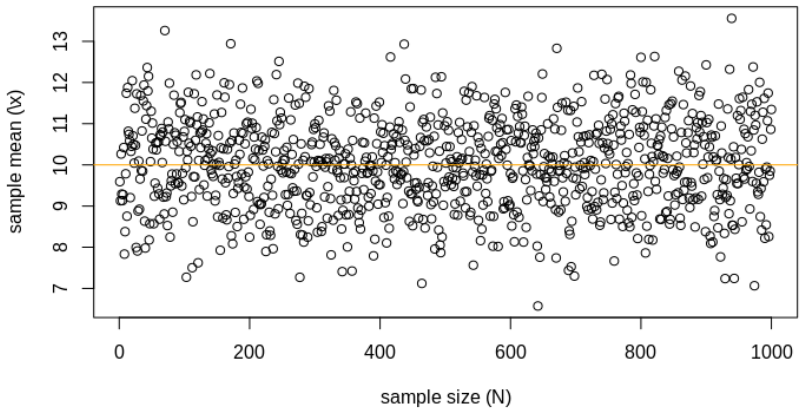

Figure 3: Sample mean values of true mean 10

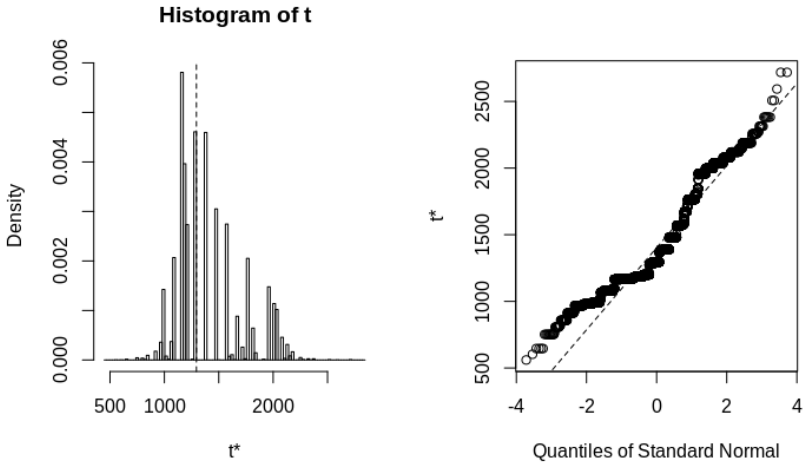

Figure 4: Wildtype cell line boost plot

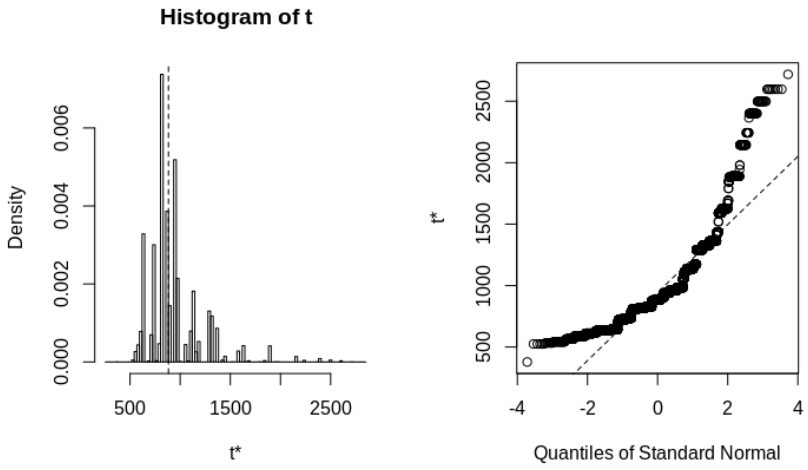

Figure 5: Knockout cell line boost plot

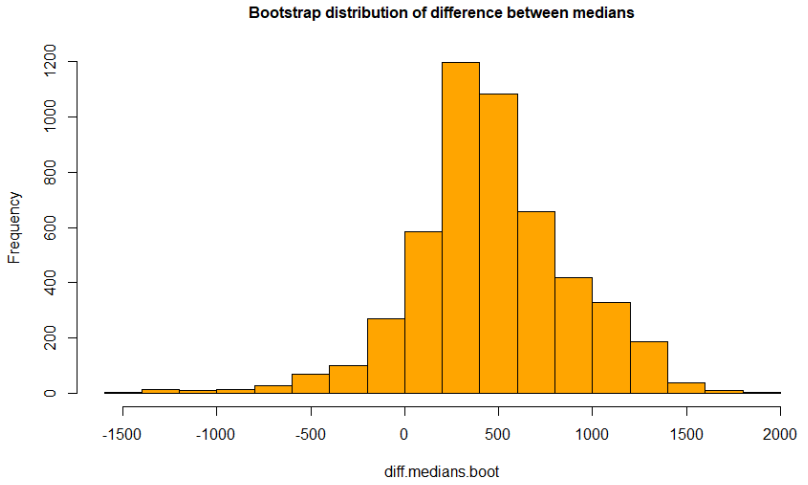

Figure 6: Bootstrap distribution of difference between medians

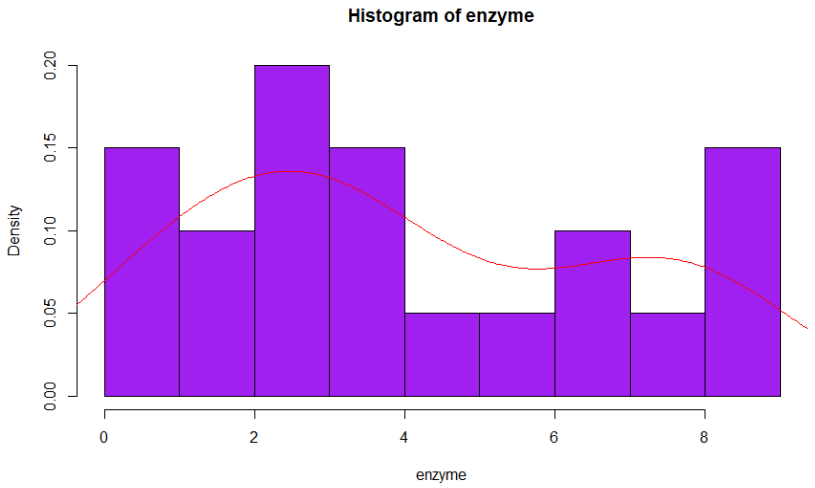

Figure 7: Enzyme distribution

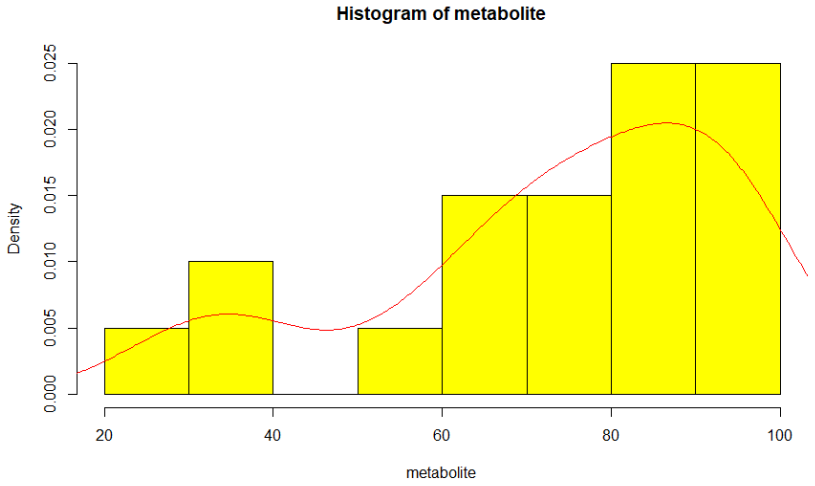

Figure 8: Metabolite distribution

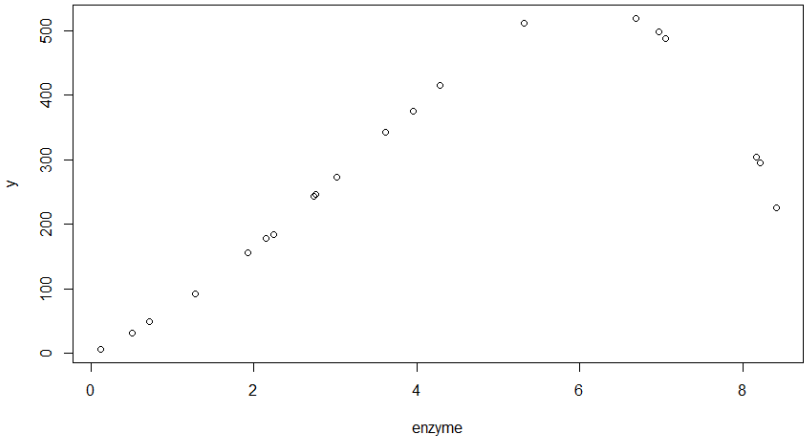

Figure 9: Enzyme and metabolite non-linearity

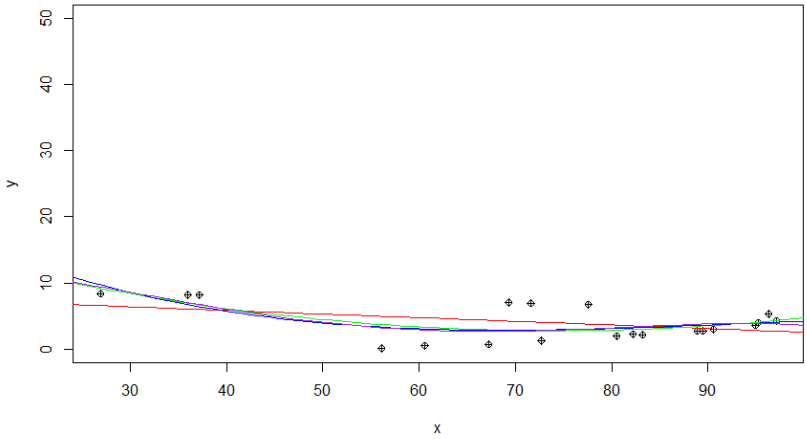

Figure 10: Polynomial fit