R programming

José R. Valverde
jrvalverde@cnb.csic.es
CNB/CSIC

Exercises

Statistics means never having to say youíre certain.

Functions

Defining your own functions

- There are many functions already available
- But sometimes we'll want to define our own ones:

```
f <- function(x) return(2*x^2 -
                        0.9*x - 1)

f(0)

f(1.5)
```

Functions are also variables

- You can use a function as you would use a variable, e.g. when calling another function.
  - If your function produces a value, it is the same as a variable that contains a value

```
plot(f)
plot(f, from=-1, to=2)
```

Exercise

- Define the function

$$g(x) = -4x^2 + 0.2x + 4$$

- And plot it with

```
plot(g, from=-1, to=2, add=T,
     col="red")
```

Minima and maxima

- We can see that f(x) has a minimum around 0.25 and g(x) a maximum around 0
- We can find the minimum with

```
optimize(f, interval=c(-1,2))
```

- Exercise:
  - Find the maximum of g(x) using optimize

Minima and maxima

- We can see that f(x) has a minimum around 0.25 and g(x) a maximum around 0
- We can find the minimum with
  - `optimize(f, interval=c(-1,2))`
- Exercise:
  - Find the maximum of g(x) using optimize
  - **Which function has a minimum at the same point where g(x) has a maximum?**
    - **DEFINE AND CHECK IT**
    - **HINT: you can even use a lambda function**

Functions with several arguments

$$f(x) = -5 - 3x_2 + 4x_2 + x_1^2 - x_1 x_2 + x_2^2$$

- We can define it in several ways
  - As a function taking <u>two</u> argumens (x_1 and x_2)
  - As a function taking <u>one</u> argument, which is a vector with two components ($x = (x_1, x_2)$)
    - val = c(x1, x1) ---> val[1] = x_1, val[2] = x_2
  - As a function taking <u>one</u> argument: a list with two components ($x = x_1, x_2$))
    - val = list(x1, x1) --> val[[1]] = val$x1 = x_1, val[[2]] = val$x2 = x_2
- Try it

Solution

```
f1 <- function(x1,x2) return(
    -5 - 3*x1 + 4*x2 + x1^2 - x1*x2 + x2^2)
f1(0,0)
f1(1,2)

f2 <- function(x) return(
-5 - 3*x[1] + 4*x[2] + x[1]^2 - x[1]*x[2] +
    x[2]^2)
f2(c(0,0))
f2(c(1,2))
```

Contour plot

- For functions of one argument we plot a line, for two arguments (dimensions) it is convenient to draw a contour plot.
  - The axis are X1 and X2
  - We draw lines/curves corresponding to different values of the function
    - All values of x1, x2 in a line give the same result for the function

Contour plot (2)

```
x1 <- seq(0,2,length=51)
x1
x2 <- seq(-3,1, length=51)
x2
fVals <- outer(x1,x2,f1)
dim(fVals)
contour(x1,x2,fVals)
```

Contour plot (3)

- We want a plot with 50 grid points in X1 and X2
  - So we need 51 points, from 0 to 50
- Function **outer( )** will compute the function in all grid points
- And then we use function **contour( )** to draw the plot.

Another dimension

- Define a function 'fun' that calculates ($x^2 + y^2$)
- now try:

```
x <- seq(-10, 10, length= 30)
y <- x
z <- outer(x, y, fun)# calculate h for all x,y points
z[is.na(z)] <- 1     # set to 1 all NAs in the result
op <- par(bg = "white")  # set a white background
persp(x, y, z,       # plot values of h in the x,y plane
      theta = 30, phi = 30,    # help(persp)
      expand = 0.5,
      col = "lightblue")
```

And yet another one

- Using the former 'h' function, color by height:

```
library(lattice)
wireframe(z, drape=T,
          col.regions=rainbow(100))
```

- Indeed, there are many ways to skin a cat!

Enzyme Kinetics

Enzyme kinetics

- Enzyme activity has been measured in various different concentrations of the substrate and the inhibitor. Activity is measured as reaction rate.
- We have measures of the reaction rate (R) with (I) no inhibitor, 50 μM and 100 μM, and for each series, we have measured 6 different substrate concentrations (S) from 10 μM to 600 μM
- There are two replicas of the experiment
- Data is in **inhib.xlsx** and **inhib.csv**

Inspect the data

- Read the data into variable '**inhib**'
- Make a scatterplot with substrate concentration (S) on the x-axis and reaction rate (R) on the y-axis.
- Can you spot any problem?

Inspect the data

- Read the data into variable '**inhib**'
- Make a scatterplot with substrate concentration (S) on the x-axis and reaction rate (R) on the y-axis.
- Can you spot any problem?
  - look into 'inhib.csv'

```
inhib <- read.table("inhib.csv",
                    sep=";",
                    header=T)
with(inhib, plot(S, R))
```

Inspect the data

- Read the data into variable '**inhib**'
- Make a scatterplot with substrate concentration (S) on the x-axis and reaction rate (R) on the y-axis.
- Can you spot any problem?
  - **Can you tell apart the reactions with different inhibitor concentrations?**

Inspect the data (2)

- Our problem is that the data is in columnar form. Each inhibitor concentration follows the former ones in the same column.
  - We could convert to horizontal format, but we can also play a neat trick
  - We know that we have 12 observations for each inhibitor concentration, so you can try
  - `attach(inhib)`
  - `grp <- c(rep(1,times=12),`
  - `rep(2,times=12),`
  - `rep(3,times=12))`
  - `plot(S, R, col=grp)`
  - `plot(S, R, pch=grp)`
  - `plot(S, R, col=grp, pch=grp)`

Inspect the data (3)

- We have created a vector with 12 ones, 12 twos and 12 threes.
- Then we simply use this vector to indicate the color (or marker) for each value in S/R.
  - The first 12 observations will use color/marker #1
  - The second 12 obervations, color/marker #2
  - The third 12 observations, colot/marker #3

Normal reaction rate

- In the absence of inhibitors, the reaction rate follows what is called a Michaelis-Menten relation:

$$R \approx \frac{V_{max} \cdot S}{K + S}$$

- Where V_{max} and K need to be estimated from the data (typically using least-squares fitting)

Estimating V_{max} and K

- We can use **nls( )** (non-linear-least-squares) to do the fitting, but we need starting values
  - First, select the first 12 rows (the data subset with no inhibitor (I = 0)
  - Call this selection e.g. 'dat0'

```
dat.i0 <- inhib[1:12, ]
```

  - Make the NLS model

```
mm.i0 <- nls(R ~ Vmax * S / (K + S),
        start = list(Vmax=3, K=100),
        data=dat.i0)
summary(mm.i0)
```

Validation

- We can analyze the raw residuals to check if they (the error in the measure) do indeed follow a normal distribution.

```
plot(fitted(mm.i0),
     residuals(mm.i0))
```

- What do you think?

Validation (2)

- Make a scatterplot for the data and add the line corresponding to the fitted function.
  - You will need to define a function to calculate the Michaelis-Menten curve, using the values for Vmax and K estimated previously.
  - Then plot the data for dat0
  - Then plot your function from 0 to max(dat0) and add it to the previous plot (with argument add=T)

Validation (2)

- Make a scatterplot for the data and add the line corresponding to the fitted function.

```
plot(dat.i0$S, dat.i0$R)
mi.men.i0 <- function(S) 2.9811 * S /
                         (35.853 + S)
plot(mi.men.i0, from=0,
     to=max(dat.i0), add=T)
```

Modelling inhibitor effect

- The association between inhibitor, substrate and reaction rate may be modeled as

$$R \approx \frac{V_{max} \cdot S}{K_1 \cdot \left(1 + I / K_2\right) + S}$$

- Here, we need to estimate V_{max}, K_1 and K_2 from the data.

Do it

- Fit the model with nls( ).
  - You may try starting with $V_{max} = 3$, $K_1 = 100$ and $K_2 = 25$ as an initial guess.
- Make the scatterplot of all the observations using colors for each inhibitor concentration
- Add three different fitted curves, one for each inhibitor concentration using the same color for each concentration as for the datapoints

Solution

```
mmi <- nls(R ~ Vmax * S / (K1 * (1 + (I/K2)) + S),
       start=list(Vmax=3, K1=100, K2=25), data=inhib)
summary(mmi)

plot(S, R, col=grp) # we already have grp
# for [ I ] = 0
fm <- function(S) 2.93828 * S /
                (33.99345 * (1 + 0 / 34.84463 ) + S)
plot(fm, from=0, to=620, add=T, col=1)
fm <- function(S) 2.93828 * S /
                (33.99345 * (1 + 50 / 34.84463 ) + S)
plot(fm, from=0, to=620, add=T, col=2)
fm <- function(S) 2.93828 * S /
                (33.99345 * (1 + 100 / 34.84463 ) + S)
plot(fm, from=0, to=620, add=T, col=3)
```

Compare with initial fit

- Add the line corresponding to the model of the reaction obtained considering only the observed reaction rates in the absence of the inhibitor

```
plot(mi.me.i0, from=0, to=620,
     add=T,
     col=1, lty=2)
```

Getting the coefficients

- We can get them with summary()
- When we "print" it, we see it "nice"
- But it is actually a list, with among others, an element called coefficients... if we assign them names (variables) we can get

```
sum.mmi <- summary(mmi)
coef <- sum.mmi$coefficients
Vmax <- coef['Vmax', 'Estimate']
K1 <- coef['K1', 'Estimate']
K2 <- coef['K2', 'Estimate']
```

Use variables whenever possible

- Compare this new version

```
m <- function(I, S) Vmax * S /
                (K1 * (1 + I / K2 ) + S)
```

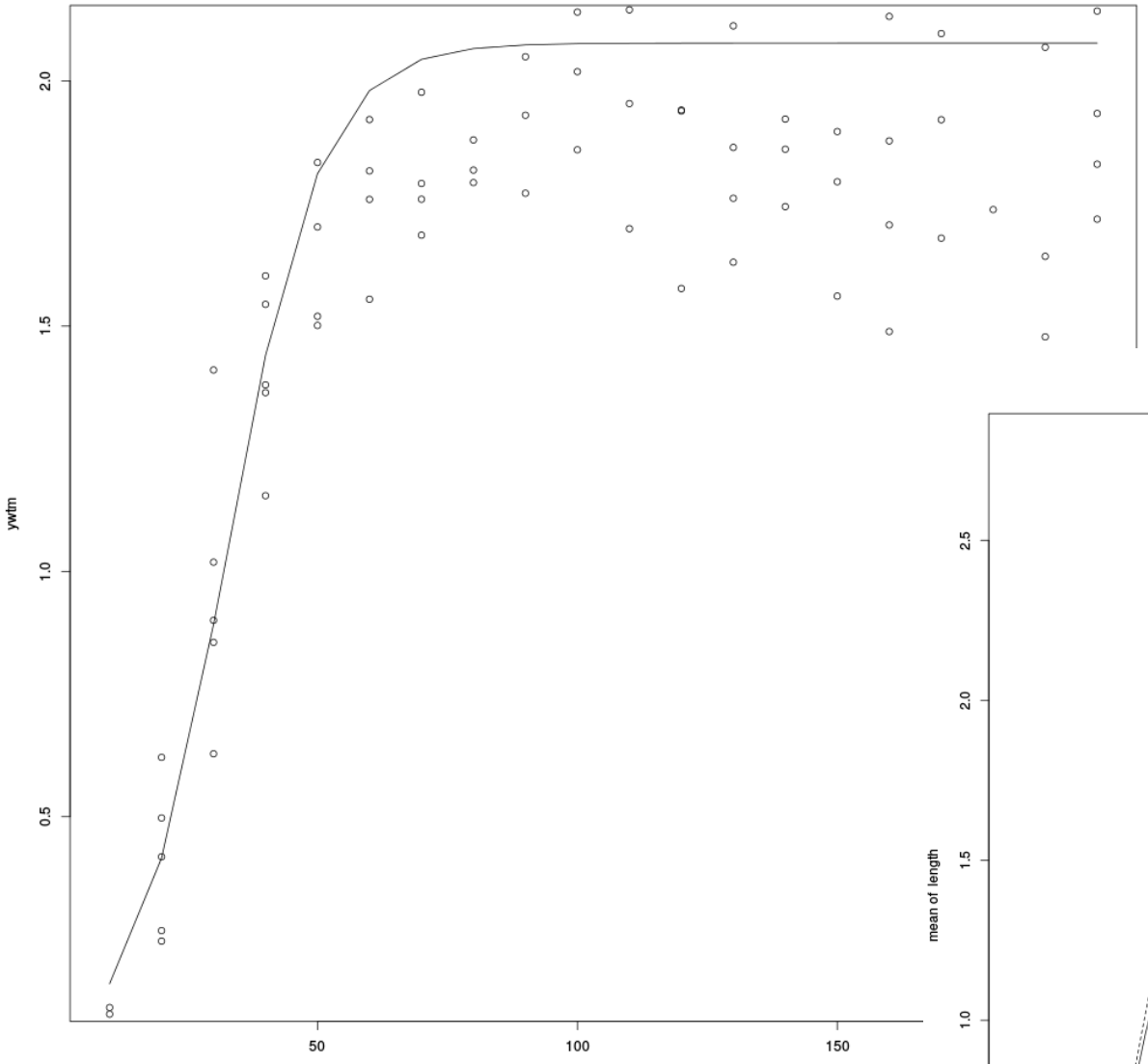
- and using a for loop we can do all inhibitor concentrations at once

```
for (I in c(0, 50, 100))
    plot(function(S) m(I, S), from=0,
         to=620)
```

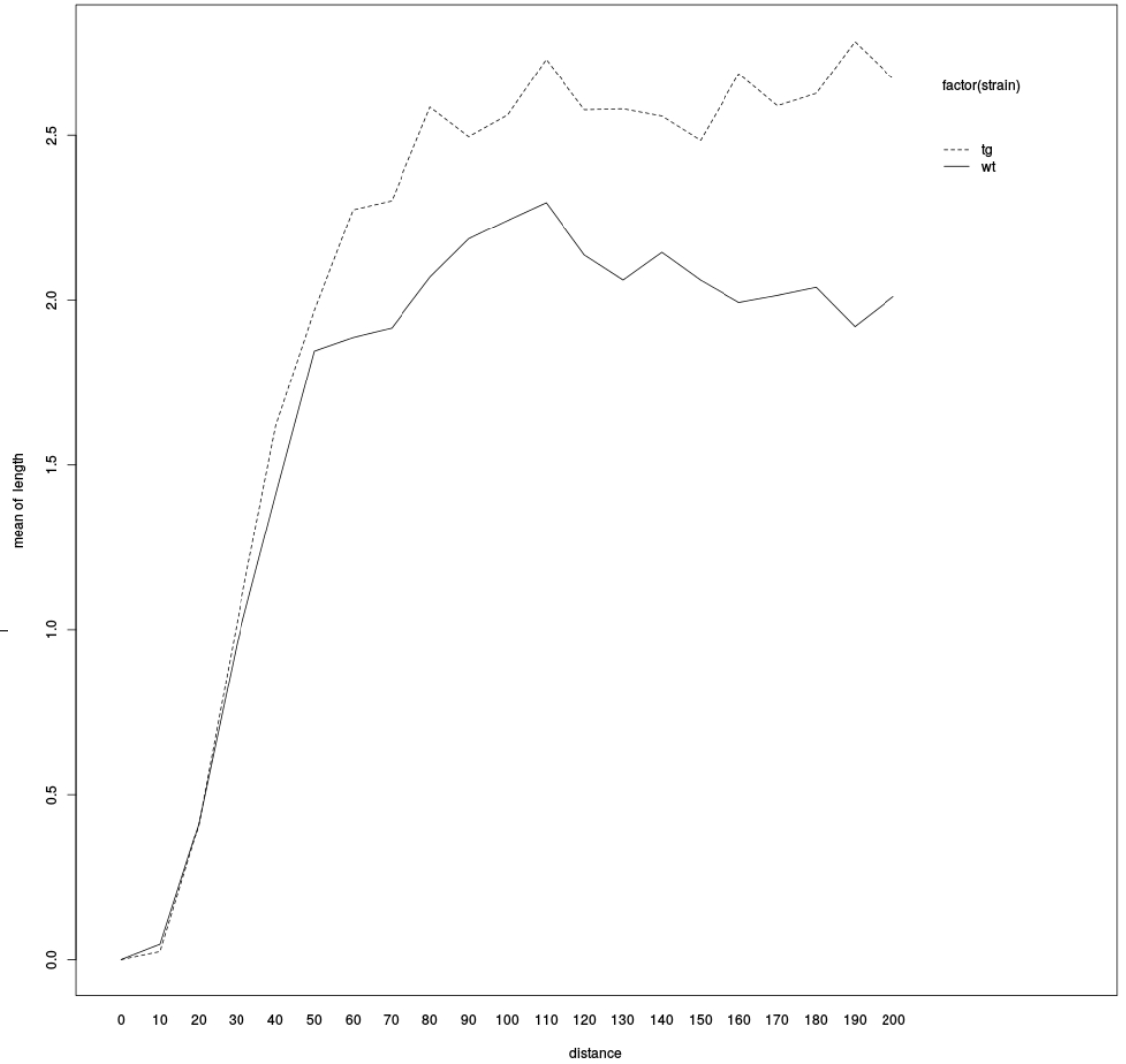
- but, since m takes two arguments, and plot uses a function with one, we make a new lambda function that takes only one argument to pass the inhibitor concentration at each iteration of the loop
- this is incomplete: we need some additional steps to also add colors and symbols, but you get the idea.

Yonder and beyond

- There is a lot more to curve fitting, specially when we talk about enzyme reactions
- You may have more than one substrate, more than one inhibitor and various error (residual) distributions. This may demand
  - Various logistic regression, Generalized Least Squares, ARIMA or mixed-effects modeling methods
  - Fitting to sigmoidal, third-degree polynomials or other types of curves
  - Repeated measures analyses
  - Time series analyses...



This is real world data:
Data from the wt strain (left) is not well fitted by a usual MM curve: it shows a decrease in the reaction rate after the peak, indicating presence of a triggered inhibitor



Comparison with the mutant (right) shows differences in the curves: the mutant does not produce the inhibitor and fits a MM curve. The wt, starts to inhibit the reaction once a trigger concentration of the product is reached.

More Info

- You can obtain more examples, details and information in

  Introduction to R: Exercises

  Helle Sørensen, 2015

  U. Of Copenhagen

- Which it so happens is the source upon which some parts of this presentation are based.

Thank you

Questions?