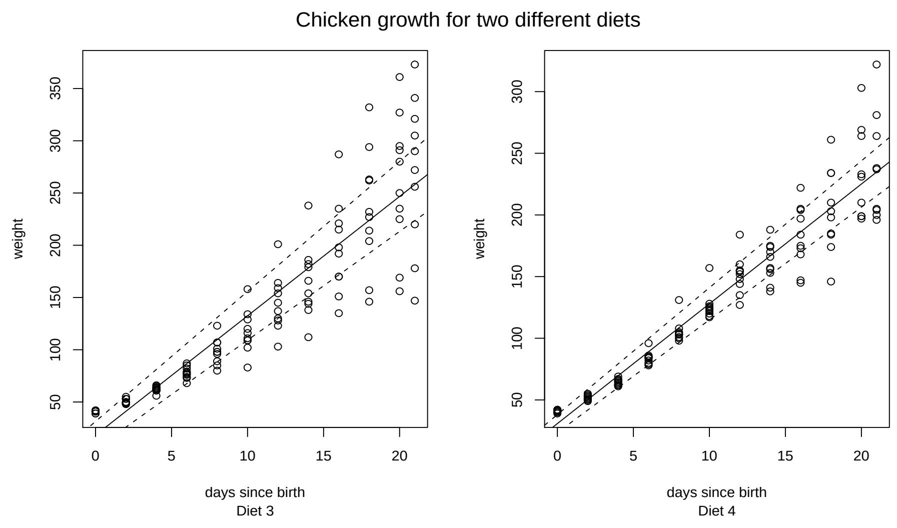
Exercises for the course

"An introduction to R"

Sheet 09

Exercise 42: Recall the ChickWeight data from Exercise 26. We consider only Diet 3 and Diet 4 here. Store the weight vector corresponding to Diet 3 into `w3`. Store the time vector corresponding to Diet 3 into `t3`. Find the linear model which best explains `w3` as function of `t3`. Furthermore extract the confidence intervals for the regression line. The corresponding lines are added to the figure below with line type "dashed". Proceed similarly for Diet 4. What are the values of r squared of the two linear regressions. Produce a picture which resembles the following figure.



Hint: The main title is magnified with 1.5. The ratio between the axis can be changed by using the mouse to resize the plotting window. (5 points)

Exercise 43: Write a function `which.NA()` which returns the vector of indices at which the function argument has `NA`s. Here is how it should work:

```
> which.NA(c(1,2,NA,4,NA,6))
[1] 3 5
```

Hint: `is.na()`. Write a function `rm.NA()` which returns its argument without `NA`s.

```
> rm.NA(c(1,2,NA,4,NA,6))
[1] 1 2 4 6
```

(3 points)

Exercise 44: Write a function `stderr()` which calculates the standard error

$$\frac{\mathrm{sd}(x)}{\sqrt{\mathrm{length}(x)}}$$

of its argument `x`. What happens if you apply this function to `c(3,5,"a",7)` or to `c(3,NA,8,2)`? Improve the definition of `stderr()` as follows. Use `is.numeric()` to check whether the argument is numeric. If it is not numeric, then print the warning message `"Argument is not numeric: returning NA"` with the command `warning()` and return `NA`. Furthermore add an argument `na.rm` to the definition of your function and let its default value be `FALSE`. If that argument is `TRUE`, then remove all `NA`s from the argument vector and continue as before. Here is how it should work:

```
> stderr(c(3,5,"a",7))
[1] NA
Warning message:
In stderr(c(3, 5, "a", 7)) : Argmunt is not numeric: returning NA
> stderr(c(3,NA,8,2))
[1] NA
> stderr(c(3,NA,8,2),na.rm=TRUE)
[1] 1.855921
```

(4 points)

Exercise 45: The sample mean is sensitive to outliers, e.g. `mean(1:4)` is 2.5 whereas `mean(c(1:4,1000))` is 202. To reduce the impact of outliers write a function `trim.mean()` which removes the smallest and the largest element of its argument and returns the mean of the remaining vector. Assume that the argument is numeric and contains no `NA`s. Here is how it should work:

```
> trim.mean(c(-100,1:4,1000))
[1] 2.5
> mean(1:4)
[1] 2.5
```

(2 points)

Exercise 46: Write a function `sum.sq()` which calculates the sum of squares

$$\sum_{i=1}^{\text{length(x)}} x_i^2$$

of its argument `x` with a loop.
Of course we know that `sum(x^2)` is way more efficient. Let us test this. Compare the run time of `sum.sq(x)` with the run time of `sum(x^2)` for $x \in \{1e3, 1e5, 1e7\}$. You get the runtime of a command by passing it as argument to `system.time()`. (3 points)

Exercise 47: Define `cols <- colors()` to be the vector of color names. Using regular expressions find the following elements of `cols`:

- All color names which contain ”yellow”.
- All colors whose name starts with ”yellow”.
- All colors whose name starts with ”yellow” or with ”orange”.
- All color names whose third character is 'c' (This is not interesting for color names. However when looking at DNA data one might want to find sequences with a certain polymorphism at a given locus).

(4 points)