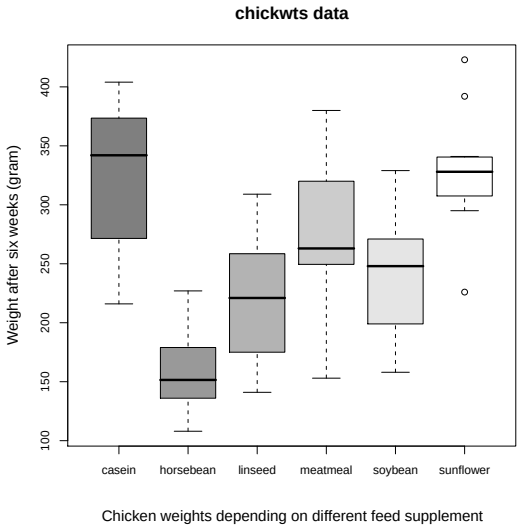
Exercises for the course

"An introduction to R"

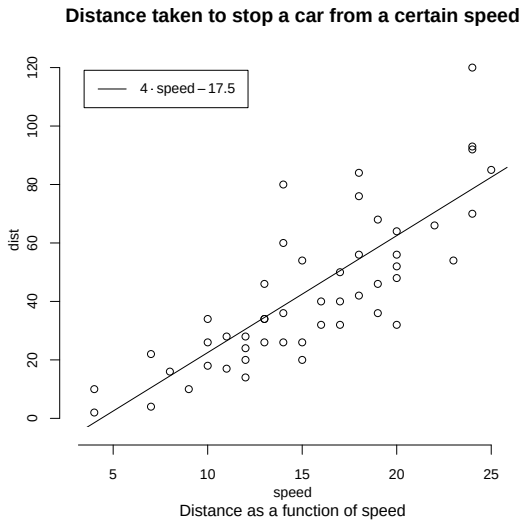
Sheet 04

Exercise 19: *The data frame* `chickwts` *contains measured growth rates of chickens in dependence of various feed supplements.*
If `chickwts` is not known, then say `data(chickwts)` in order to load it. Produce the following figure:

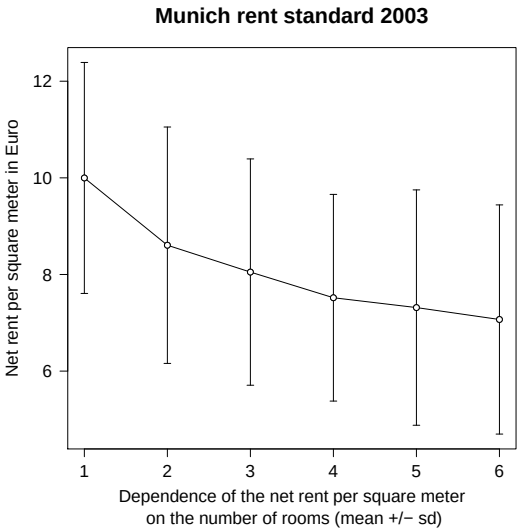


Use the colors 'grey50', 'grey60', 'grey70', 'grey80', 'grey90' and 'grey100'. The main title is magnified with 1.5, the subtitle and the label of the y-axis are magnified with 1.3.

Exercise 20: *The data frame* `cars` *gives the distance taken to stop from a certain speed. Note that the data were recorded in the 1920s.* Plot `dist` as a function of `speed` without titles and without axes (`axes=FALSE`). Add a line with slope 4 and intercept -17.5. Furthermore add axes with the command `axis()`. Specify the positions of the ticks with `at=`. 'Pretty' positions of the ticks are generated with `pretty(speed)` and `pretty(dist)`, respectively. You might want to draw the axis into the margins by one line (`line=1`). Add a legend with `legend()`. The command locator(1) might help you to find a good position for the legend. The text in the legend is generated with the command `expression()`. Furthermore add a main title and a subtitle with the command `title()`.



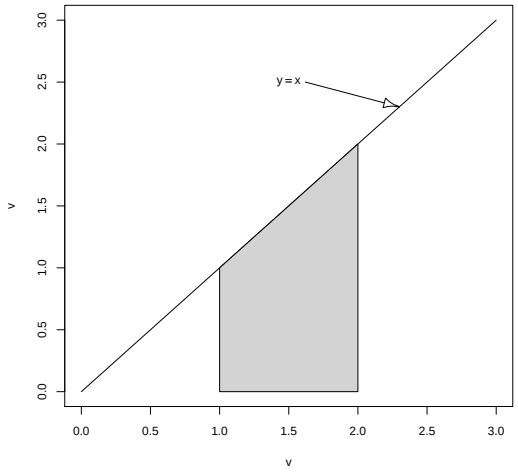
Exercise 21: *The high-level plotting command* `errbar` *produces a plot with error bars.* Load the data from `miete03.txt` (course web page) into the data frame `rent` and attach `rent`. Group the vector `nmqm` of net rents per square meters into subvectors according to `rooms` and calculate the mean for each subvector, that is, apply `mean()` to `nmqm` grouped according to `rooms`. Let the resulting vector be denoted as `means`. Then calculate the standard deviations as follows. Apply `sd()` to `nmqm` grouped according to `rooms` and denote the resulting vector as `sds`. Now plot `means` as function of the number of rooms with the command `errbar(1:6,means,means+sds,means-sds,xlab=NA,ylab=NA,las=1)`. You could also specify axis labels. The last plot command adds error bars between `means-sds` and `means+sds`. Then add lines which connect the means. To produce "white circles", overprint the existing circles with the command `points()`. Use point character 21 and fill the circles with colour 'white'. Your result should resemble the following figure.



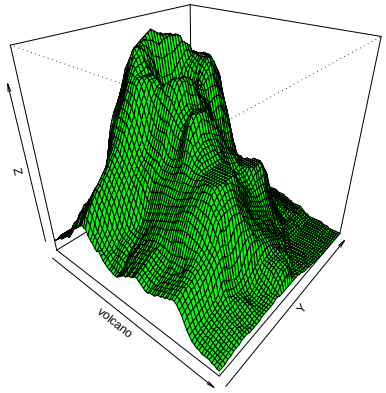
Exercise 22: *This exercise is to practice how to merge data frames*
New Bachelor and Master students startet at the beginning of the semester. So we need to add them to the list of students at the biology department. Download the files 'studentsfile.txt', 'studentsnew1file.txt' and 'studentsnew2file.txt' from the web page.

- Read the files into the variables 'students', 'studentsnew1' and 'studentsnew2'.
- Merge the data frames 'students' and 'studentsnew1'. Consult the help page `?merge` for how to keep *all* rows for the resulting data frame.
- Merge the data frames 'students' and 'studentsnew2'. The difficulty is that now the column names 'name' and 'firstname' do not agree. One solution is to change 'firstname' into 'name'. Find another solution how to merge 'students' and 'studentsnew2' (Hint: *by.x=*, *by.y=*).

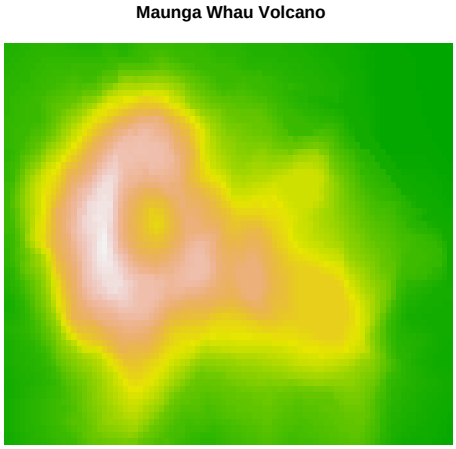
Exercise 23: *How to produce arrows and polygons*
Define a vector `v` which increases from 0 to 3 in steps of size 0.1. Plot `v` against `v` with `type="l"`. Then add a shaded area with the low-level plotting command `polygon(x,y,col="lightgrey")`. Here the vector `x` is to be defined as the vector of the four x-coordinates of the shaded area, that is $(1, 2, 2, 1)$. The vector `y` is to be defined as the vector of the four y-coordinates of the shaded area, that is $(0, 0, 2, 1)$. Then add text at $(1.5, 2.5)$. This text is an expression which produces $y = x$. Load the library 'sfsmisc' to have the command `p.arrows()` at your disposal. See the help page `?p.arrows()`. Use `fill=` to specify the colour of the arrow head. The command `locator()` is useful to find the coordinates of the points between which the arrow is to be drawn. The resulting figure should be as follows.

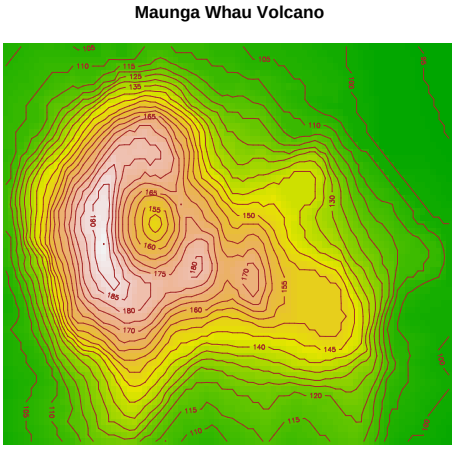


Exercise 24: *The following high-level plotting commands produce pretty plots. We will not go into detail here as such plots are less important for biological data.*
Enter `data(volcano)` to load the volcano data into the variable `volcano`. Plot this variable with the command `persp`. Set the angles `theta=40` and `phi=30` to define a suitable viewing direction. Let the color be `col="green"`.



Next plot the variable `volcano` with the command `image`. By default, `image` uses `heat.colors(12)` to visualize the third dimension with colours. As we have landscape data it is better to use `col=terrain.colors(100)`. The axes have no meaning here so supress both axes. The main title of the following figure (left-hand side) is magnified by a factor 1.5.





To improve the impression of the third dimension, add contour lines with the command `contour()`. The contour lines are added to the previous plot if you specify `add=TRUE`. Specify the levels of the contour plot with `levels=` ; use the vector which increases from 90 to 200 in steps of size 5 for the levels. Use `col=` to change the colour of the level lines to 'brown'. You could also try `heat.colors()` or `topo.colors()` instead of `terrain.colors()`.