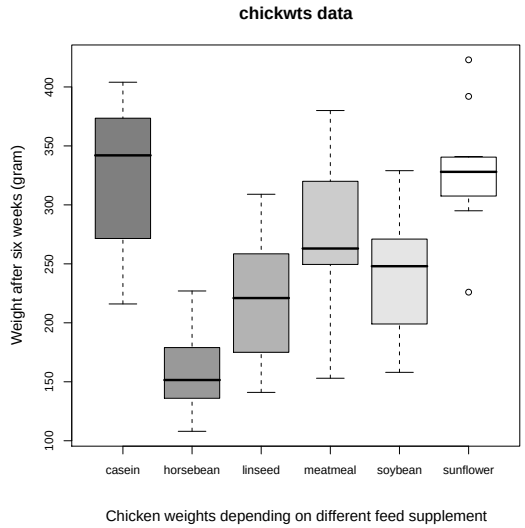
Exercises for the course

"An introduction to R"

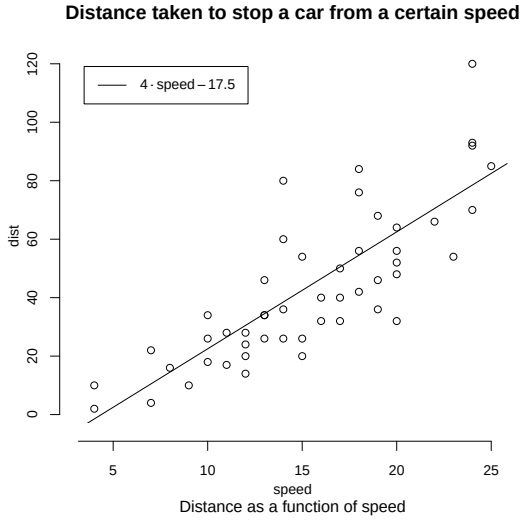
Sheet 05

Exercise 20: *The data frame* `chickwts` *contains measured growth rates of chickens in dependence of various feed supplements.* Produce the following figure:



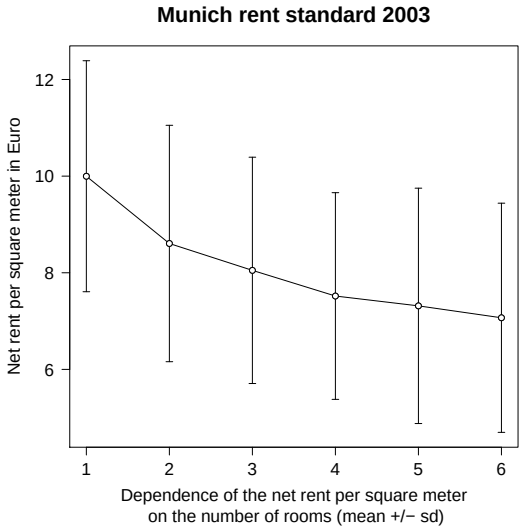
Use the colors 'grey50', 'grey60', 'grey70', 'grey80', 'grey90' and 'grey100'. The main title is magnified with 1.5, the subtitle and the label of the y-axis are magnified with 1.3. (2 points)

Exercise 21: *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()`.



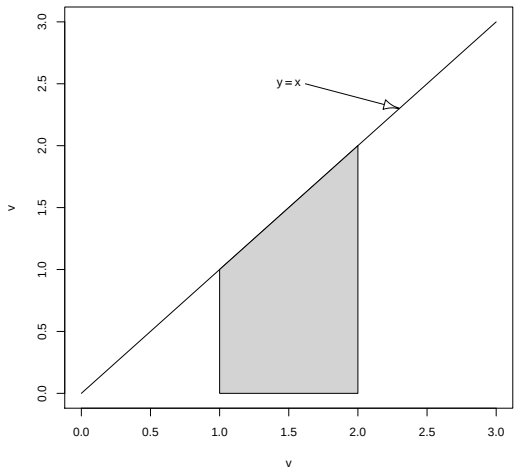
(5 points)

Exercise 22: *The high-level plotting command* `errbar()` *produces a plot with error bars.* Load the library `sfsmisc` with `library(sfsmisc)` to have `errbar()` available. If `sfsmisc` is not installed on your system, then try `install.packages(sfsmisc)`. 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.



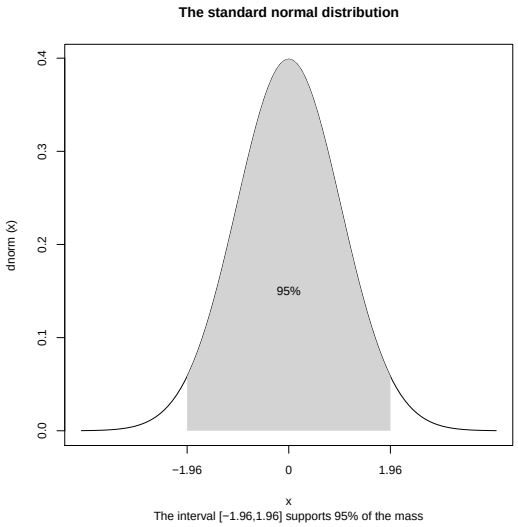
(3 points)

Exercise 23: 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.



(4 points)

Exercise 24: Produce the following figure:



Hints: First plot a the density of the standard normal distribution without axes. Add the y-axis. Add the x-axis and specify where to put the ticks and specify the labels (-1.96,0,1.96) of the ticks. The box around the plot is added with `box()`. The grey area is added with `polygon()` as follows. Define a vector `u` which increases from -1.96 to 1.96. Choose the step size yourself. Define a vector `x` to be the concatenation of `u` and `rev(u)`. So `x` increases from -1.96 to 1.96 and then decreases back to -1.96. Define a second vector `y` to be the concatenation of `rep(0,length(u))` and of `dnorm(rev(u))`. So `y` follows the x-axis as long as `x` increases and then goes back along the graph of `dnorm`. Use `x` and `y` as arguments for `polygon()`. Change the border of the polygon with `border=`.

(5 points)