Advanced computational statistics, lecture 5

Frank Miller, Department of Computer and Information Science, Linköping University

April 29, 2025

LINKÖPING UNIVERSITY

Course schedule

- Topic 1: **Gradient based optimisation**
- Topic 2: **Stochastic gradient based optimisation**
- Topic 3: **Gradient free optimisation**
- Topic 4: **Optimisation with constraints**
- Topic 5: **EM algorithm and bootstrap**
- Topic 6: **Simulation of random variables**
- Topic 7: **Numerical and Monte Carlo integration; importance sampling**

Course homepage: http://www.adoptdesign.de/frankmillereu/adcompstat2025.html

Includes schedule, reading material, lecture notes, assignments

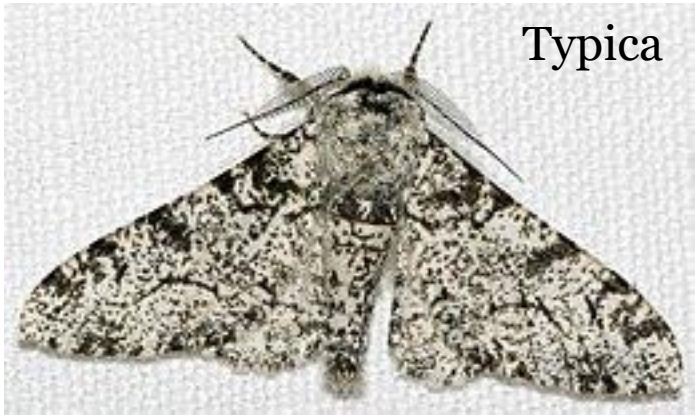
EM algorithm

EM algorithm

- EM = “Expectation-Maximization”
- Main application of this algorithm is in situations where not all data is observed
- E: Expectation will be taken over all (unobserved) data which lead to the observed data
- Algorithm is iterative:
  each iteration has an E step, followed by an M step

EM algorithm: Example 1

- Classical example: Genotype–phenotype
- Peppered moths (see Ex.4.2 in GH, “björkmätare”)



Source: Wikipedia; Pictures of moths taken by Olaf Leillinger. Licence: CC BB-SA 3.0

- Alleles: C, I, T; genotypes: CC,CI,CT; II,IT; TT
- Observed only phenotype: *carbonaria; insularia; typica*
  Frequency observed: n_C; n_I; n_T
- Aim: estimate allele frequencies p_C, p_I, p_T based on observed phenotype frequencies

EM algorithm: Example 1

- Observed data: $\boldsymbol{X} = (N_C, N_I, N_T)$
- Complete data: $\boldsymbol{Y} = (N_{CC}, N_{CI}, N_{CT}, N_{II}, N_{IT}, N_{TT})$
- Aim: estimate $\boldsymbol{p} = (p_C, p_I, p_T)$
- We can specify
  - the expectations $E[\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{p}]$ and
  - the complete data likelihood $f_{\boldsymbol{Y}}(\boldsymbol{y}|\boldsymbol{p})$

EM algorithm: Example 1

- According to biological theory:

$$P(\text{a random moth is CC}) = (p_C)^2$$
$$P(\text{a random moth is CI}) = 2p_C\, p_I$$
$$\dots$$

- The complete data likelihood $f_{\boldsymbol{Y}}(\boldsymbol{y}|\boldsymbol{p})$ is multinomial:

$$f_{\boldsymbol{Y}}(\boldsymbol{y}|\boldsymbol{p}) = (p_C^2)^{N_{CC}} * (2p_C p_I)^{N_{CI}} * \cdots * \binom{N}{N_{CC} \quad N_{CI} \quad \dots}$$

- Complete data log likelihood:

$$\log f_{\boldsymbol{Y}}(\boldsymbol{y}|\boldsymbol{p}) = N_{CC} * \log(p_C^2) + N_{CI} * \log(2p_C p_I) + \cdots$$

- Expectations $E[\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{p}]$ are for example:

$$E[N_{CC}|N_C, N_I, N_T, \boldsymbol{p}] = N_C \frac{p_C^2}{p_C^2 + 2p_C p_I + 2p_C p_T}$$

EM algorithm

- Let $\boldsymbol{X}$ be observed data, $\boldsymbol{Y}$ complete data, $\boldsymbol{\theta}$ unknown parameter-vector, $L(\boldsymbol{\theta}|\boldsymbol{x})$ likelihood to be maximized
- Iteration t $(t = 0,1, \ldots)$: $\boldsymbol{\theta}^{(t)}$
- Let $Q\left(\boldsymbol{\theta}\middle|\boldsymbol{x}; \boldsymbol{\theta}^{(t)}\right) = E\left\{\log L(\boldsymbol{\theta}|Y)|\boldsymbol{x}; \boldsymbol{\theta}^{(t)}\right\}$ be expectation of **log likelihood for complete data** conditional on observed data $\boldsymbol{X} = \boldsymbol{x}$
- EM algorithm:
  1. Initialize parameter-vector with a guess $\boldsymbol{\theta}^{(0)}$, $t = 0$
  2. **E step:** Compute $Q\left(\boldsymbol{\theta}\middle|\boldsymbol{x}; \boldsymbol{\theta}^{(t)}\right)$
  3. **M step:** Maximize $Q\left(\boldsymbol{\theta}\middle|\boldsymbol{x}; \boldsymbol{\theta}^{(t)}\right)$ with respect to $\boldsymbol{\theta}$ -> result is $\boldsymbol{\theta}^{(t+1)}$
  4. If not stopping criterion (e.g. $\left(\boldsymbol{\theta}^{(t+1)} - \boldsymbol{\theta}^{(t)}\right)^T \left(\boldsymbol{\theta}^{(t+1)} - \boldsymbol{\theta}^{(t)}\right) < \epsilon$) met, set `t <- t+1`, and go back to E step

EM algorithm: Example 2

- Effect of a drug to be measured and n patients (randomly chosen out of a population of patients) treated with the drug
- $X_i, i = 1, \ldots, n$, observed for each patient after drug-treatment
- Known that population consists of two groups:
  - One group responds well to the drug (i.e. larger X_i)
  - Another group responds only barely (smaller X_i)
- It is not known which patient belongs to which group

Observed: $\boldsymbol{X_i}$,

Unobserved: $\boldsymbol{Z_i} = \begin{cases} \mathbf{1}, \text{if patient } \boldsymbol{i} \text{ belongs to responder group} \\ \mathbf{0}, \text{otherwise} \end{cases}$

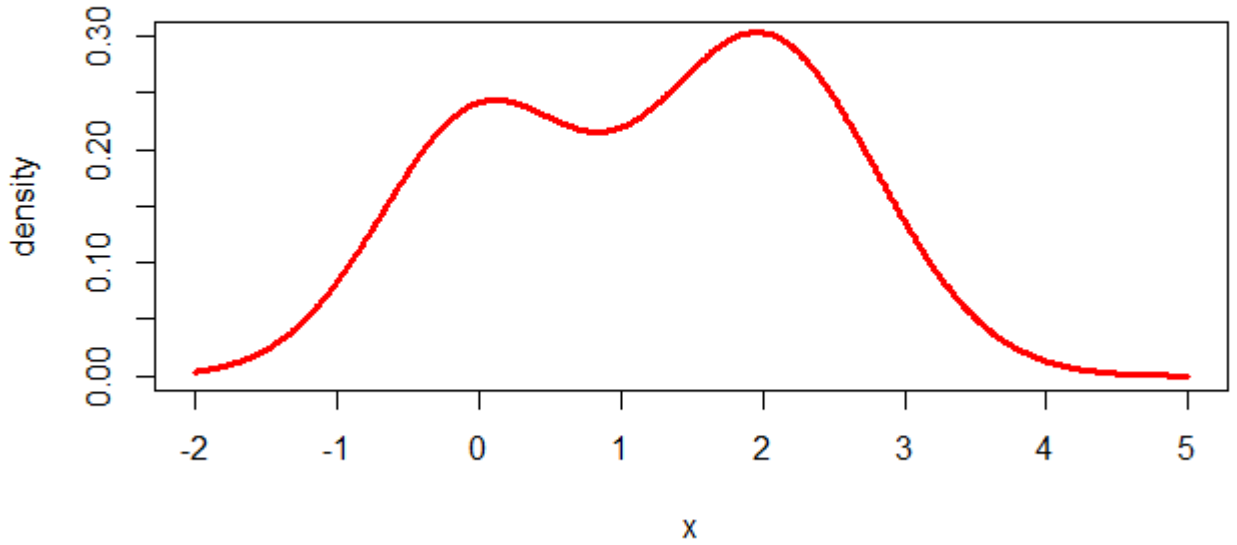
Complete data: $\boldsymbol{Y_i} = (\boldsymbol{X_i}, \boldsymbol{Z_i})$

EM algorithm: Example 2

- In this example, we assume that X_i has normal mixture density f for $c = 2$ groups (responder, non-responder)
- Generally, a normal mixture (also called GMM, Gaussian mixture model) has density f being sum of c weighted densities:
$$f(x) = \sum_{i=1}^{c} p_i \, \varphi(x; \mu_i; \sigma_i),$$
where p_i are weight or mixing coefficients ($p_i \geq 0;\ p_1 + \cdots + p_c = 1$), and $\varphi(x; \mu; \sigma)$ being density of $N(\mu, \sigma^2)$
- Here for $c = 2$ groups ($p = p_1,\ p_2 = 1 - p$):
$$f(x) = p\varphi(x;\ \mu_1; \sigma_1) + (1 - p)\varphi(x; \mu_2; \sigma_2)$$
- 5 parameters to estimate from data: $p;\ \mu_1; \sigma_1; \mu_2; \sigma_2$

EM algorithm: Example 2

- $f_M(x) = p\varphi(x;\ \mu_1;\sigma_1) + (1-p)\varphi(x;\mu_2;\sigma_2)$
- parameters: $p;\ \mu_1;\sigma_1;\mu_2;\sigma_2$



- Example here: $p = 0.4;\ \mu_1 = 0; \sigma_1 = 0.7; \mu_2 = 2; \sigma_2 = 0.8$

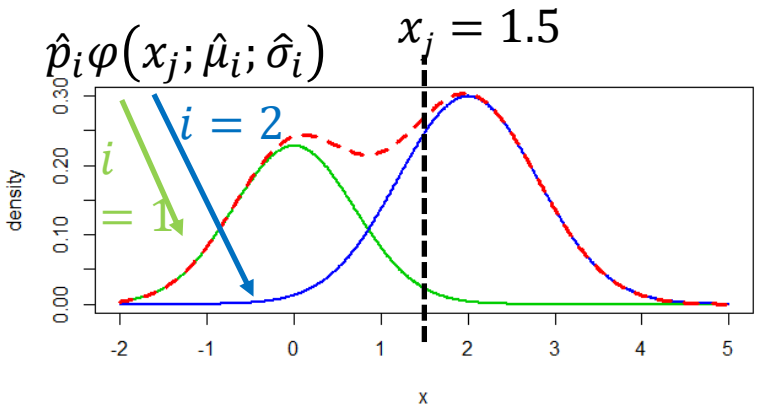
EM algorithm for normal mixtures

- The estimated probability that observation j belongs to group i (of c groups) is

E step

$$\hat{\pi}_{ij} = \frac{\hat{p}_i \varphi(x_j; \hat{\mu}_i; \hat{\sigma}_i)}{\sum_{k=1}^{c} \hat{p}_k \varphi(x_j; \hat{\mu}_k; \hat{\sigma}_k)},$$

where
$\varphi(\cdot; \mu; \sigma)$ is density of normaldist. with mean μ and sd σ



- Model parameters maximizing Q are:

M step

$$\hat{p}_i = \frac{1}{n}\sum_{j=1}^{n} \hat{\pi}_{ij},$$

$$\hat{\mu}_i = \frac{1}{\hat{p}_i n}\sum_{j=1}^{n} \hat{\pi}_{ij} \cdot x_j,$$

$$\hat{\sigma}_i^2 = \frac{1}{\hat{p}_i n}\sum_{j=1}^{n} \hat{\pi}_{ij} \cdot (x_j - \hat{\mu}_i)^2$$

Multivariate case similar, except:

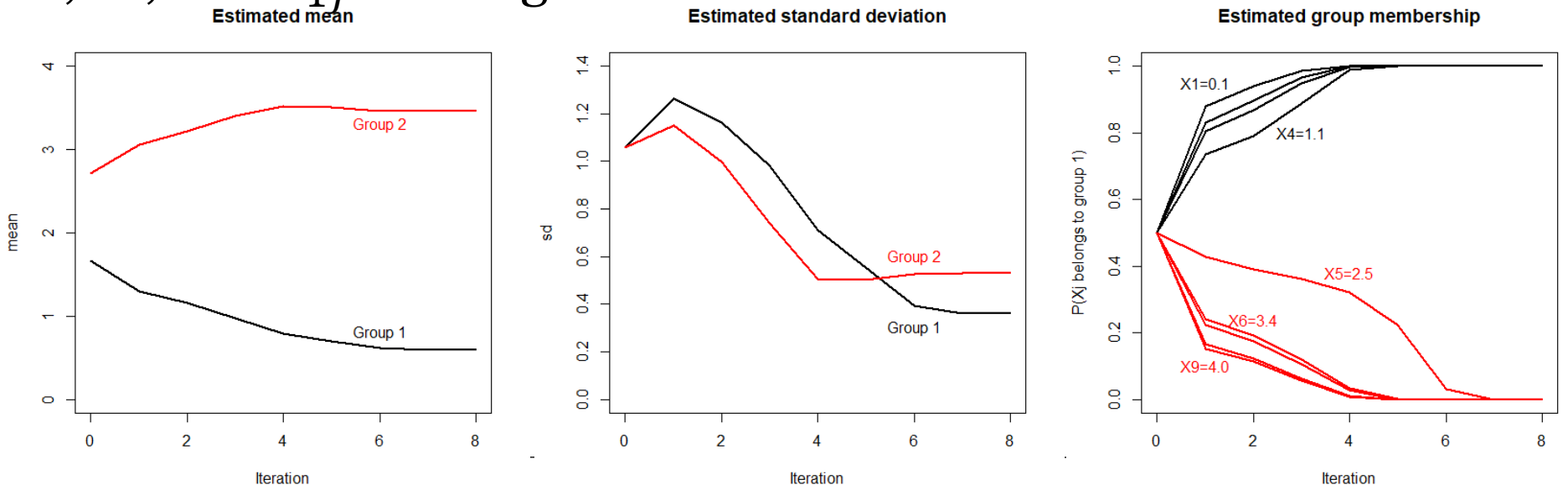
$$\hat{\boldsymbol{\mu}}_i = \frac{1}{\hat{p}_i n}\sum_{j=1}^{n} \hat{\pi}_{ij} \cdot \boldsymbol{x}_j,$$

$$\hat{\Sigma}_i = \frac{1}{\hat{p}_i n}\sum_{j=1}^{n} \hat{\pi}_{ij} \cdot (\boldsymbol{x}_j - \hat{\boldsymbol{\mu}}_i)(\boldsymbol{x}_j - \hat{\boldsymbol{\mu}}_i)^T$$

- $Q = \sum_{i=1}^{c}\sum_{j=1}^{n} \hat{\pi}_{ij}\{\log(\hat{p}_i) + \log \varphi(x_j; \hat{\mu}_i; \hat{\sigma}_i)\}$
- See Section (10.1 and) 10.2 of Lindholm, Wahlström, Lindsten, Schön (2022)

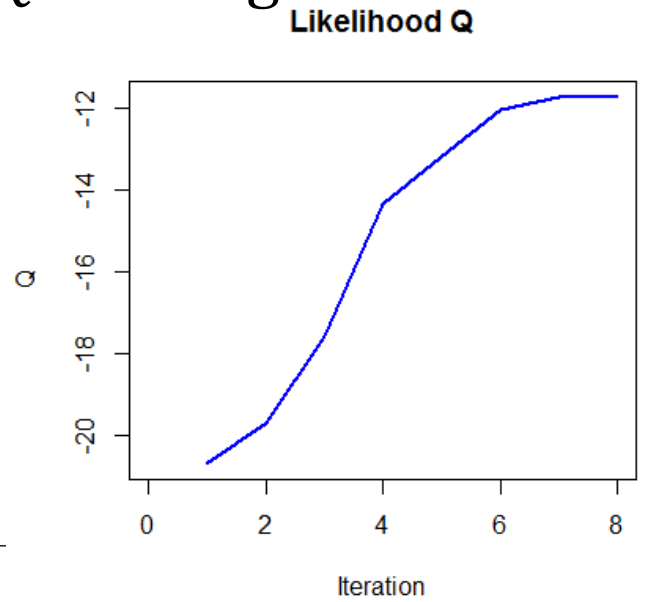
EM algorithm for normal mixtures

- Example for illustration: $n = 9$ observations obtained. Ordered data: 0.1, 0.5, 0.7, 1.1, 2.5, 3.4, 3.5, 3.9, 4.0
- EM algorithm terminates after 8 iterations with: $(p_1^{(8)}, \mu_1^{(8)}, \mu_2^{(8)}, \sigma_1^{(8)}, \sigma_2^{(8)}) = (0.444, 0.600, 3.460, 0.361, 0.532)$
- Mean, sd, and $\hat{\pi}_{1j}$ converge as follows:



EM algorithm for normal mixtures

- Example for illustration: $n = 9$ observations obtained. Ordered data: 0.1, 0.5, 0.7, 1.1, 2.5, 3.4, 3.5, 3.9, 4.0
- EM algorithm terminates after 8 iterations with: $(p_1^{(8)}, \mu_1^{(8)}, \mu_2^{(8)}, \sigma_1^{(8)}, \sigma_2^{(8)}) = (0.444, 0.600, 3.460, 0.361, 0.532)$
- Over the iterations, Q converges as follows:

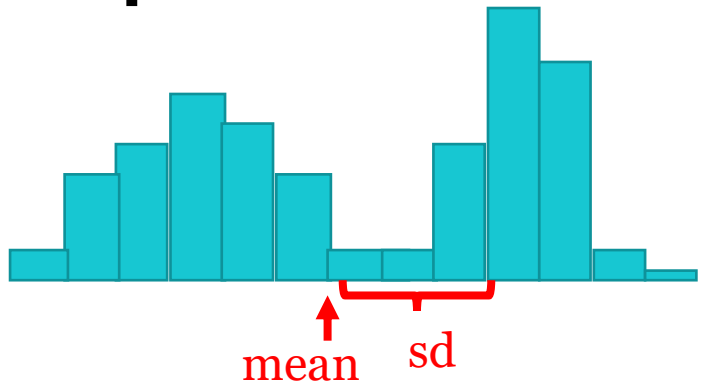


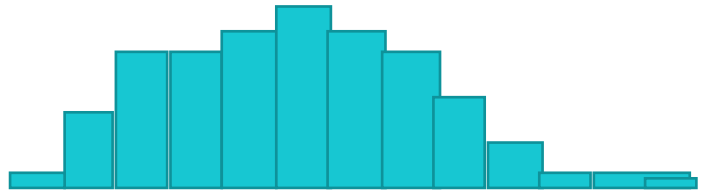
Q
-20.69286
-19.68185
-17.56861
-14.35840
-13.19032
-12.03445
-11.71313
-11.71272

```
emalg <- function(dat, eps=0.000001){
  n      <- length(dat)
  pi     <- rep(NA, n)   #initialize vector for prob. to belong to group 1
  p      <- 0.5          #Starting value for mixing parameter
  sigma1 <- sd(dat)*2/3  #Starting value for variances
  sigma2 <- sigma1
  mu1    <- mean(dat)-sigma1/2 #Starting values for means
  mu2    <- mean(dat)+sigma1/2
  pv     <- c(p, mu1, mu2, sigma1, sigma2)  #parameter vector
  cc     <- eps + 100    #initialize conv. crit. not to stop directly
  while (cc>eps){
    pv1  <- pv           #Save previous parameter vector
    ### E step ###
    for (j in 1:n){
      pi1   <- p*dnorm(dat[j], mean=mu1, sd=sigma1)
      pi2   <- (1-p)*dnorm(dat[j], mean=mu2, sd=sigma2)
      pi[j] <- pi1/(pi1+pi2)
    }
    ### M step ###
    p      <- mean(pi)
    mu1    <- sum(pi*dat)/(p*n)
    mu2    <- sum((1-pi)*dat)/((1-p)*n)
    sigma1 <- sqrt(sum(pi*(dat-mu1)*(dat-mu1)/(p*n)))
    sigma2 <- sqrt(sum((1-pi)*(dat-mu2)*(dat-mu2)/((1-p)*n)))
    ######
    pv     <- c(p, mu1, mu2, sigma1, sigma2)
    cc     <- t(pv-pv1)%*%(pv-pv1)
  }
  pv
}
data <- c(0.1, 0.5, 0.7, 1.1, 2.5, 3.4, 3.5, 3.9, 4.0)
emalg(data)
```

Choice of starting values in example before

- We want to create automatically starting values which are meaningful for the data
- My heuristic rule to choose them in the R-code before:
  - Take total data and compute overall mean and sd
  - Overall sd is usually larger than sd's for groups
  - Therefore, I took 2/3* overall sd for the sd in both groups
  - For group means, starting values with 1 sd difference chosen



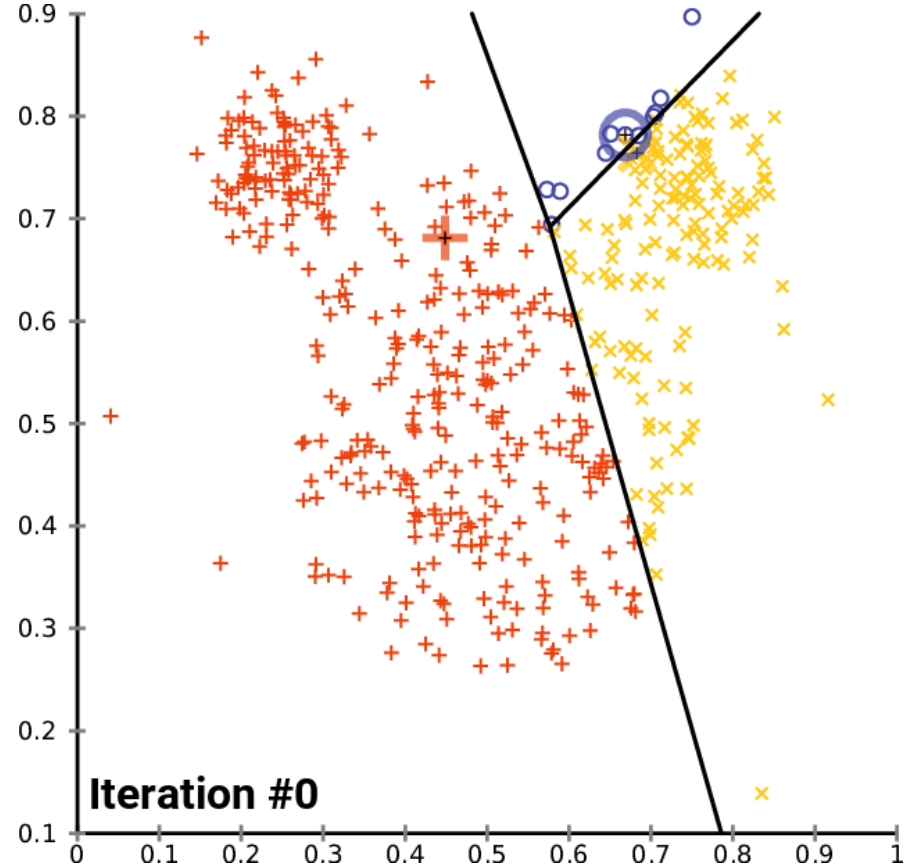


EM algorithm: Example 3

- We consider now an unsupervised learning situation with multivariate data coming from c groups, but it is unknown from which group each observation comes from (i.e., we have unlabeled data)
- Task: estimate to which group the observations belong to (i.e., classification)

Excursus: k-means clustering algorithm

- Initialize with k means
  $$\boldsymbol{\mu}_1^{(0)}, \cdots, \boldsymbol{\mu}_k^{(0)}$$
- <u>Assignment step:</u>
  Each observation is assigned to the nearest mean $\boldsymbol{\mu}_i^{(t)}$
- <u>Update step:</u>
  For each group i calculate the new mean $\boldsymbol{\mu}_i^{(t)}$
- Iterate until groups do no longer change



Annimation by: Chire
https://commons.wikimedia.org/wiki/File:K-means_convergence.gif

Excursus: k-means clustering algorithm

- The k-means algorithm creates clusters of similar size
- Sometimes more flexibility about cluster size desired

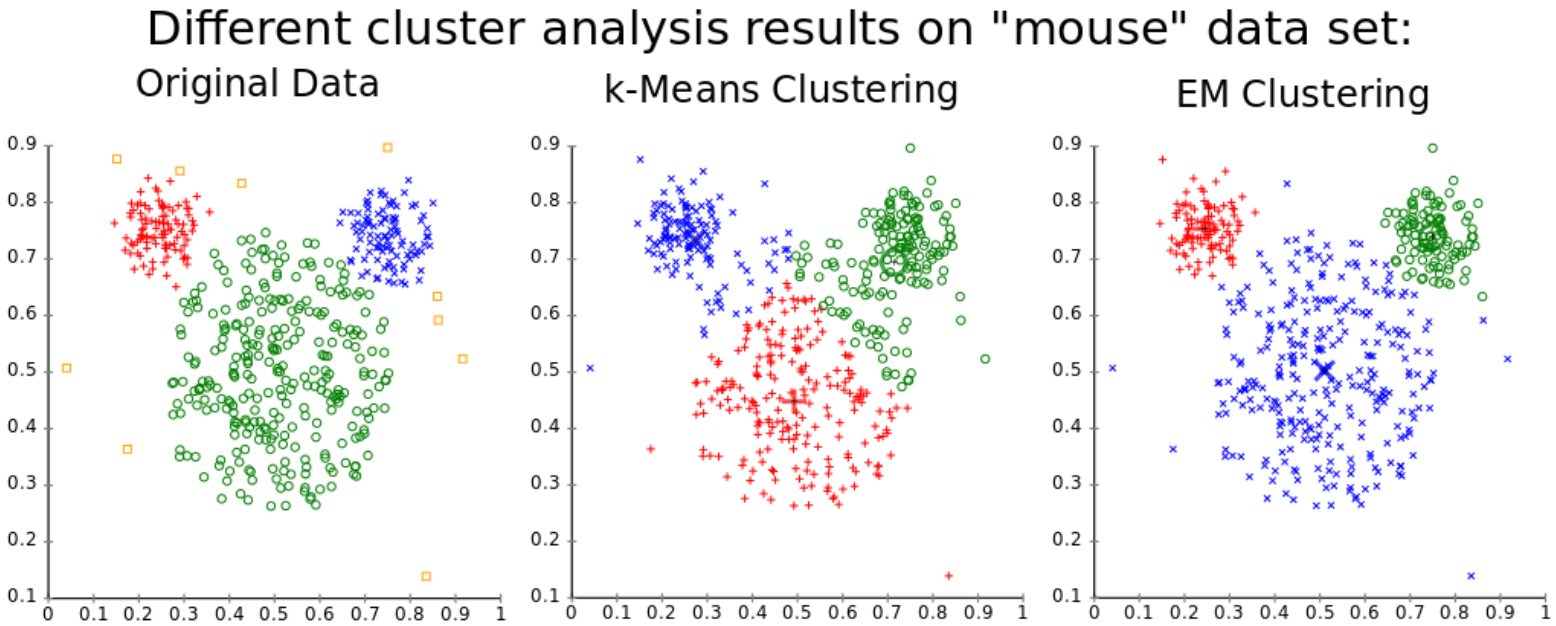


Figure by: Chire
https://upload.wikimedia.org/wikipedia/commons/0/09/ClusterAnalysis_Mouse.svg

- Assuming a multivariate normal mixture enables using the EM algorithm

Choosing starting values, connection to k-means

- We can look at the data and guess the components in the mixture, their mean and variance
- We can use a heuristic rule to determine starting values (like in Example 2)
- We can try a grid of starting parameter values
- Specifically, for the EM algorithm for normal mixtures, we can first run a classification algorithm and use its result as start for the EM algorithm

- Note (cp. Sec. 10.2 of Lindholm, Wahlström, Lindsten, Schön, 2022):
  The k-means algorithm can be seen as special case of the EM algorithm for normal mixtures when the variances tend to 0

Convergence criterion for iterative methods

- Compare $\boldsymbol{\theta}^{(t)}$ and $\boldsymbol{\theta}^{(t+1)}$ and stop if they are "close enough"
  - Absolut stopping criterion, $\left\|\boldsymbol{\theta}^{(t+1)} - \boldsymbol{\theta}^{(t)}\right\| < \epsilon$,
  - Relative stopping criterion, $\left\|\boldsymbol{\theta}^{(t+1)} - \boldsymbol{\theta}^{(t)}\right\| / \left\|\boldsymbol{\theta}^{(t+1)}\right\| < \epsilon$,
  - Modified rel. stopping crit., $\frac{\left\|\boldsymbol{\theta}^{(t+1)} - \boldsymbol{\theta}^{(t)}\right\|}{\left\|\boldsymbol{\theta}^{(t+1)}\right\| + \varepsilon} < \varepsilon$
  - Different norms $\|\cdot\|$ can be used

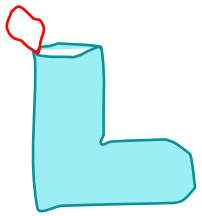
Slide adapted from L1

- Instead of $\boldsymbol{\theta}^{(t)}$ and $\boldsymbol{\theta}^{(t+1)}$, one can compare $g(\boldsymbol{\theta}^{(t)})$ and $g(\boldsymbol{\theta}^{(t+1)})$ (but note: not all iterative methods require the calculation of $g(\boldsymbol{\theta}^{(t)})$ and then, it would add computational time)
- EM: $g(\boldsymbol{\theta}^{(t)}) = Q\left(\boldsymbol{\theta}^{(t)} \middle| \boldsymbol{x}; \boldsymbol{\theta}^{(t-1)}\right)$; therefore, a reasonable stopping criterion is $\left|Q\left(\boldsymbol{\theta}^{(t+1)} \middle| \boldsymbol{x}; \boldsymbol{\theta}^{(t)}\right) - Q\left(\boldsymbol{\theta}^{(t)} \middle| \boldsymbol{x}; \boldsymbol{\theta}^{(t-1)}\right)\right| < \varepsilon$

Bootstrap

Why bootstrap?

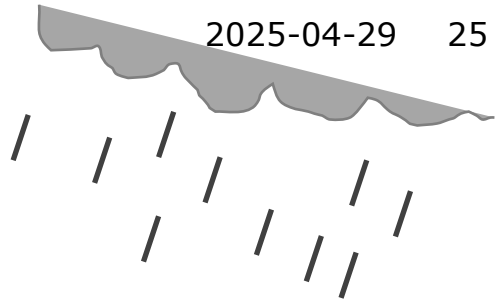
- Assume you have independent samples of some population
- In statistics, we have methods to construct confidence intervals (CIs) for a parameter θ of interest (e.g., mean) based on distributional assumptions; e.g., explicit formulas exist in case of normal distribution
- Sometimes not reasonable to make distributional assumptions
- Aim here: **obtain CIs without these distributional assumption**
- We take the **available sample as assumption for distribution of population** and **resample** from it
- We pull ourselves up by our own capabilities –
  like “pulling us up from the mud by our own **bootstraps**”



Bootstrap method

- Observed data: $D = (X_1, \dots, X_n)$
- Of interest: An estimator $T(D) = \hat{\theta}$ for some parameter θ and its uncertainty (e.g., CI for θ)
- Draw B resamples $D_i^* = (X_1^*, \dots, X_n^*)$ **of size n** from original data D **with replacement**
  - $B = 500$ or 1000 has been used historically; $B = 10000$ is nowadays often no problem
  - Usually, there are repetitions in a resample
- Calculate the property of interest for each resample: $\hat{\theta}_i = T(D_i^*), i = 1, \dots, B$
- The distribution of these B values (”bootstrap distribution”) gives information about distribution of $T(D)$
  - E.g., a CI for θ can be computed

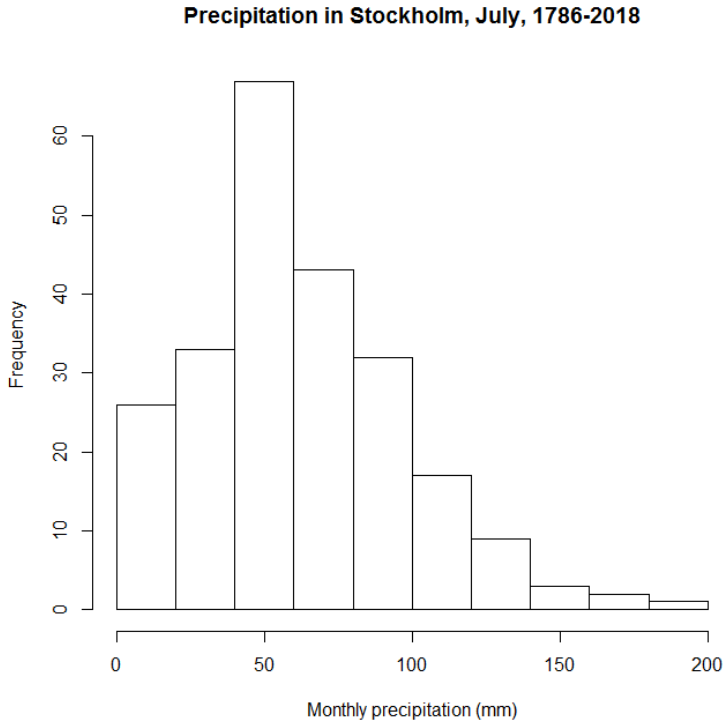
Example: precipitation data



- Rainfall data from July in 233 years in Stockholm
- What is the mean and a 95%-CI for the mean?
- A standard formulae for the CI assumes that data is normally distributed and uses therefore the t-distribution:

$$\bar{x} = 62.6\,mm, s = 35.0, n = 233,$$
$$s_{\bar{x}} = s/\sqrt{n} = 2.29,$$
$$t_{0.025,233} = 1.970$$

- 95%-CI-bounds: $\bar{x} \pm s_{\bar{x}} \cdot t_{0.025,233}$; here: $(58.1, 67.1)$
- But data here is not normally distributed
- Now, we construct a CI using the bootstrap method



Data source: SMHI

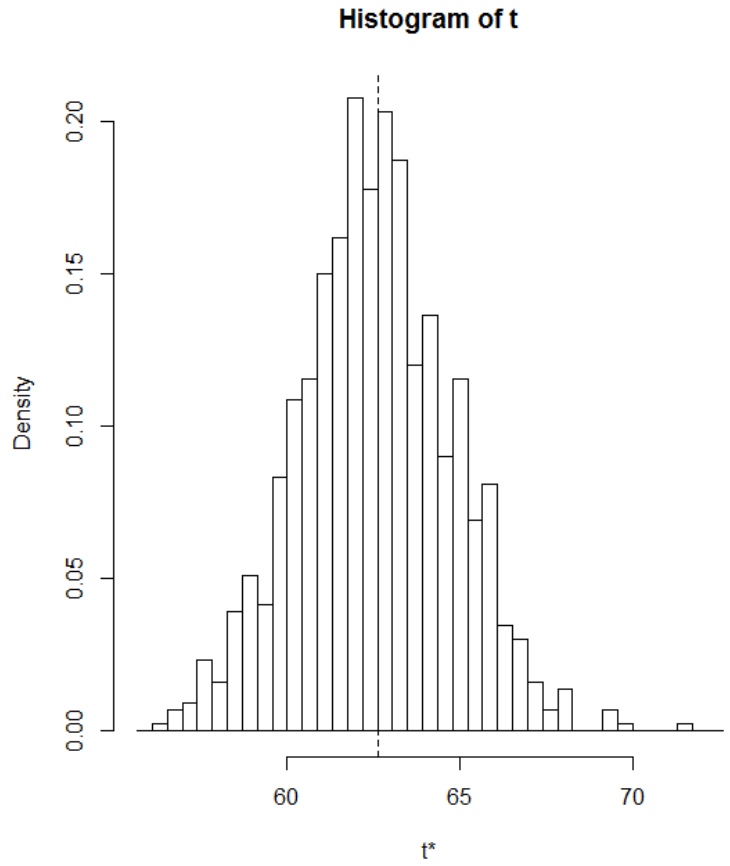
Example: precipitation data

- We illustrate the bootstrap using only the last 6 years: `42.3, 44.1, 91.9, 47.6, 14.6,  5.9`
- First resample: `5.9, 42.3,  5.9, 47.6, 91.9, 91.9`
- Second resample: `42.3, 44.1, 42.3, 91.9, 42.3, 14.6`
- Third resample: `47.6, 44.1, 42.3, 14.6, 91.9, 14.6`
- ...
- B-th resample: `47.6, 42.3, 91.9, 91.9,  5.9, 42.3`

- The mean of each resample: 47.6, 46.3, 42.5, ..., 53.7

Example: precipitation data

- From the complete data, we made $B = 1000$ resamples; the 1000 means of those are in the histogram
- The mean of the means: 62.6 mm
  (bootstrap estimate is here the same as the usual estimate of the mean $\bar{x}$)
- The middle 95% of the means are from 58.2 to 66.7 – this is our 95%-bootstrap-CI for the mean
  This is: limits are the 2.5% and 97.5% percentiles
- This way to define the CI is called **percentile method**



Bootstrap in R

- R code using a loop for bootstrap replicates:

```
bo <- 1000   # bootstrap replicates
bs <- c()    # to save the results for the means
for (l in 1:bo){
  x  <- sample(mrain, size=length(mrain), replace=TRUE)
  bs <- c(bs, mean(x))
}
hist(bs)
bss  <- sort(bs)
ci95 <- c(bss[round(bo*0.025)], bss[round(bo*0.975)])
ci95
```

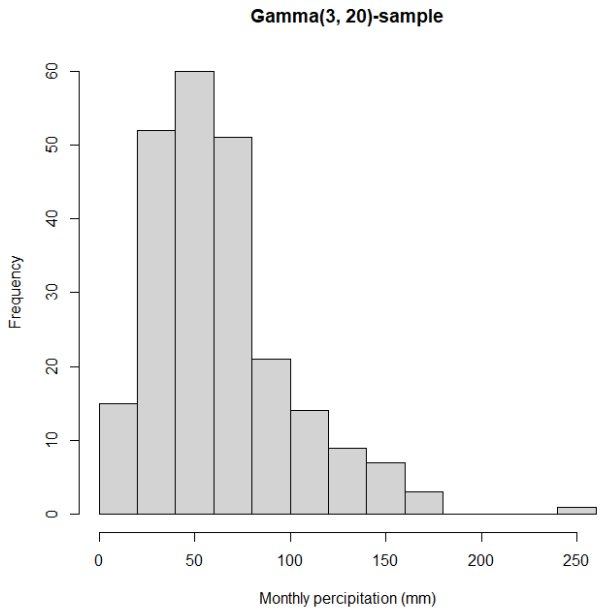
- A run of this code gave (58.2, 66.7) as 95% bootstrap confidence interval

Bootstrap in R with package `boot`

- As alternative, package `boot` with functions `boot` and `boot.ci` can be used
  `library(boot)`
- Define first function of interest, e.g. the mean:
  `bootmean <- function(x, i) mean(x[i])`
- Generate B bootstrap resamples with function `boot`:
  `bss <- boot(mrain, bootmean, R=1000)`
- You can plot a histogram of the bootstrap distribution:
  `hist(bss$t)`
- A 95%-CI is between 2.5%- and 97.5%-percentile of bootstrap distribution:
  `boot.ci(bss, type="perc")`

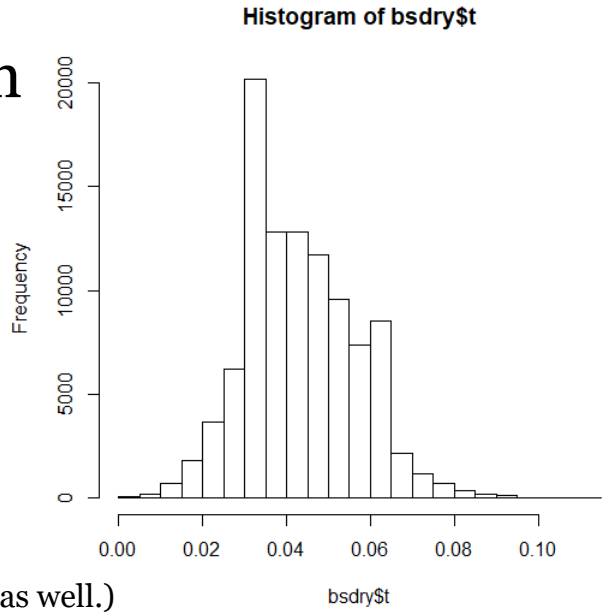
Parametric bootstrap

- When a parametric model for the data is known or believed to represent the reality well, we can do parametric bootstrap and sample according to the assumed model
- Example: We assume that monthly precipitation in July follows a Gamma(3, 20)-distribution
- We sample 233 datapoints from Gamma(3, 20) and calculate parameter of interest
- Do this B times and derive e.g. a confidence interval



Example: precipitation data

- What is an estimated probability for "less than 10 *mm* rain in next July"? How good is our estimation? (➔ CI)
- Reasonable to calculate proportion of years with July-rain < 10 *mm*. Here: in 10 of 233 years = 0.043
- To calculate a 95%-CI, we generate a bootstrap distribution
  (We resample *B* times and compute for each resample the proportion of years with July-rain < 10 *mm*)
- We use it's 2.5%- and 97.5%-percentile:
  `(0.0172, 0.0687)`
- Conclusion: The probability for < 10 *mm* rain in July is between 1.7% and 6.9%; estimate is 4.3%



- (With normal assumption an estimate would be 6.6%. But a probability for < 0 *mm* rain would be 3.7%...
  To use bootstrap gives here much better estimates than with normal assumption! You get easily a confidence interval as well.)

Bootstrap in R with package `boot`

- Define function of interest, here proportion below 10 mm:
  `bootdry <- function(x, i) mean((x[i]<10))`
- Generate $B = 100000$ bootstrap resamples:
  `bsdry <- boot(mrain, bootdry, R=100000)`
- Plot a histogram of bootstrap distribution:
  `hist(bsdry$t)`
- Estimate proportion:
  `bootdry(mrain)`
- A 95%-CI is between the 2.5%- and 97.5%-percentile of the bootstrap distribution:
  `boot.ci(bsdry, type="perc")`

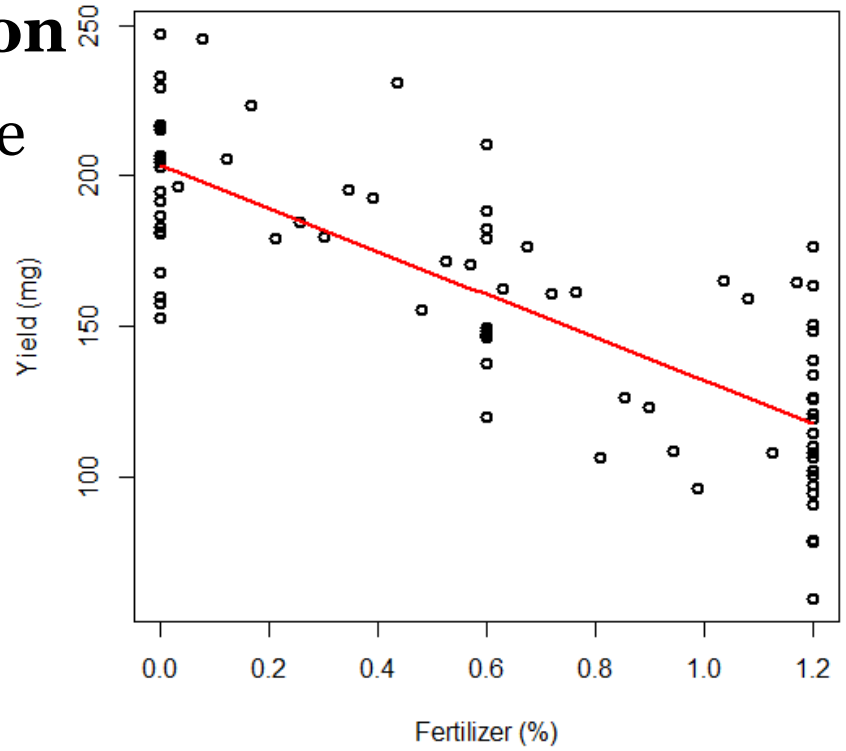
Bootstrap for regression models

- We can use the bootstrap method very flexibly, e.g. **in linear regression** if we want a **CI for the slope or the residual standarddeviation**
- Example: Experiment about the (toxic) influence of a fertilizer on the growth of garden cress (yield vs. amount of fertilizer, $n = 81$)
- Estimated linear regression:

$$\text{yield} = 203.3 - 71.3 \cdot \text{fertilizer}$$

with residual standarddeviation $\hat{\sigma} = 26.7$

- CI for slope? CI for $\hat{\sigma}$?

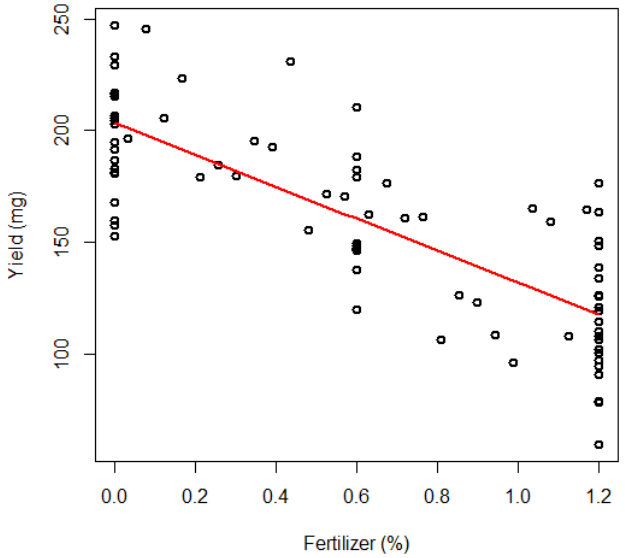


Bootstrap for regression models

- The dataset has $n = 81$ pairs of fertilizer-yield-values
- The bootstrap resamples ***n* pairs** with replacement, computes regression-slope and $\hat{\sigma}$
- This is done B times; R-code:

```
cressdat <- data.frame(fertilizer, yield)
cmslope  <- function(dat, i){
  cm  <- lm(yield~fertilizer, subset=i, data=dat)
  coef(cm)[2]
}
cb <- boot(cressdat, cmslope, R=10000)
boot.ci(cb, type="perc")
```

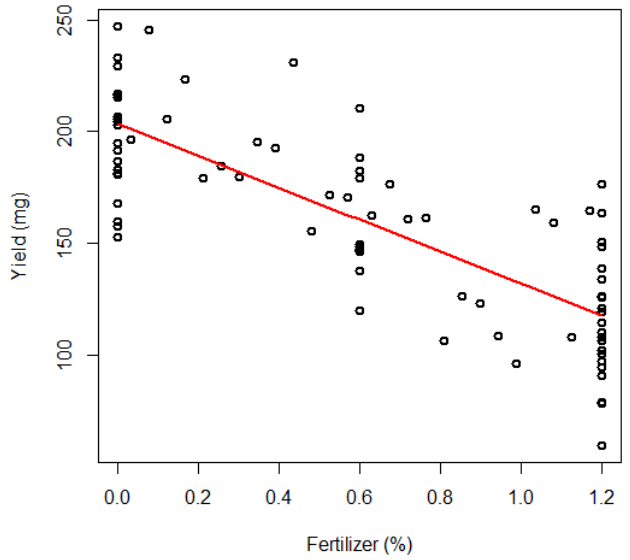
- Result for CI-limits: -83.8, -59.1



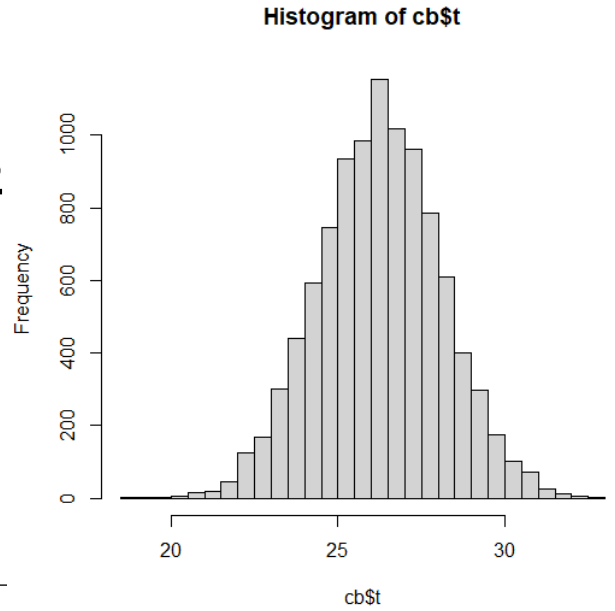
Bootstrap for regression models

- A function for analysis of the residual $\hat{\sigma}$ is:

```
cmressd <- function(dat, i){
  cm  <- lm(yield~fertilizer, subset=i, data=dat)
  summary(cm)$sigma
}
```



- Result for CI-limits: 22.62, 29.89 (percentile method)
- Median (50% percentile) of bootstrap distribution: 26.32
- Residual $\hat{\sigma}$ of data: 26.72
- Percentile CI is constructed around 26.32 while it should be constructed around 26.72 ➔ the CI is biased



Percentile method for CIs and alternatives

- The percentile method can have drawbacks
  - Bias: Estimate $\hat{\theta}$ might be very different from median of bootstrap distribution, median($\hat{\theta}_i$), but we would like a CI constructed around $\hat{\theta}$
  - The bootstrap distribution might be skewed implying that the se($\hat{\theta}$) changes with the true θ
- The BC_a method (bias correction – accelerated) improves the percentile method by
  - correcting for bias and
  - adjusting the boundary alpha-levels to handle dependence of se($\hat{\theta}$) on θ
- If bootstrap distribution has not these issues, BC_a = percentile
- For other methods (and BC_a) see Givens and Hoeting (2013), Chapter 9.3.

BC_a method for bootstrap CIs

- Like percentile method, BC_a uses quantiles from the bootstrap distribution, but instead of $\alpha/2$ and $1-\alpha/2$, it uses the two corrected quantiles
  $\Phi(z_0 + \frac{z_0 \pm z_{\alpha/2}}{1-a(z_0 \pm z_{\alpha/2})})$
- Bias: Define $z_0 = \Phi^{-1}$(proportion of bootstrap values below estimate)
- Handling of skewness with acceleration factor a:

$$a = \frac{\sum_{i=1}^{n}\left(\hat{\theta}_{(\cdot)} - \hat{\theta}_{(i)}\right)^3}{6\left\{\sum_{i=1}^{n}\left(\hat{\theta}_{(\cdot)} - \hat{\theta}_{(i)}\right)^2\right\}^{3/2}}$$

  where $\hat{\theta}_{(i)}$ is estimated leaving out observation i and $\hat{\theta}_{(\cdot)}$ is mean of $\hat{\theta}_{(i)}$
- This is a *jackknife approach* for estimating the change of se($\hat{\theta}$) when θ changes

Jackknife

- Observed data: $D = (X_1, \dots, X_n)$
- Of interest: An estimator $T(D)$ for some parameter
- n resamples defined as $D_i^* = (X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n)$ (leave-one-out sample)
- $T(D_1^*), \dots, T(D_n^*)$ give information about distribution of $T(D)$

- Jackknife variance estimation for $T(D)$:

$$\frac{1}{n(n-1)} \sum_{i=1}^{n} (T(D_i^*) - J)^2, \text{ where } J = \frac{1}{n} \sum_{i=1}^{n} T(D_i^*)$$

- Important application both for Jackknife and bootstrap is variance estimation
- Jackknife is resampling method like bootstrap, but it is deterministic

Bagging (bootstrap aggregating)

- In the examples we discussed, we had an estimate $\hat{\theta}$ and got information about its uncertainty with the bootstrap approach, e.g., constructing a CI
- In bagging, bootstrap is used to improve the estimate $\hat{\theta}$ itself by $\frac{1}{B}\sum_{i=1}^{B}\hat{\theta}_i$
- For example, if $\hat{\theta}$ is based on model-fitting where very different models could be chosen only if some observations are changed, the bootstrap estimate is model averaging
- $\hat{\theta}$ might be based on modelling with neural networks or regression models with data-dependent feature selection
- See Section 7.1-7.2 of Lindholm, Wahlström, Lindsten, Schön (2022)