# Lecture 1: Computer arithmetics

# Course structure

- We use LISAM(accessed via Student portal)
- Lectures
- Labs (computer). **Deadlines,** approximately a week after lab session
  - Individual report, group report.
- Seminars – obligatory attendance
  - Speaker groups
  - Opponent groups
- One written final exam (computer)
- Course book: *Computational statistics* by J.E. Gentle.

# Computational statistics

- Statistical analysis is often complex, paper and pen is often not enough -> computer assistance is needed

- In computational statistics, we answer questions like:
  - How to implement statistical procedures that we do not get problems like overflow?
  - How do we generate a random variable,  several correlated variables, variables coming from some multivariate distribution?
  - How to compute maximum likelihood numerically?
  - How to compute confidence (credible) intervals for complex distributions when deriving formulas is not helping?

# Course contents

- Computer Arithmetics
- Optimization
- Random number generation
- Monte Carlo methods, MCMC
- Numerical model selection and hypothesis testing
- EM algorithm and stochastic optimization

# Why statisticians need to think of computer arithmetics?

- Magnitude of numbers affects many statistical computations:

```
> t=rnorm(5,10^18,1)
> t[3]-t[4]
[1] 0
> t[1]-t[2]
[1] 0
>
```

```
> x=10^800
> dispersion=10^400
> xnew=x/dispersion
> xnew
[1] NaN
```

# Data presentation and measures

- Computer data is stored in binary form (bits)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |

- 1 Byte=8bit (typical unit!)
- 1 Word = 32 or 64 bit (depend on comp)
- 1KB=1024bytes
- 1MB=1024 KB
- …

# Characters

- ASCII (American standard code for information exchange)
  - Each character – 1 byte (totally $2^8$ characters)
  - English letters+arabic numerals+punctuation

- Unicode
  - Each character – 2 bytes
  - Variety of languages

# Fixed-point (integer) system

- Each integer can be represented as a sequence of bits: $A=a_0 2^0+a_1 2^1+a_2 2^2+\ldots.$ Try with A=5 !

- Integer may occupy a word, half of word or double word

- Negative numbers:
  - **Leading bit**: first bit=1 if negative (easy)
  - Two's complement (short numbers): 8=00001000 , -8=11110111+1=11111000

    Try to add +8 and -8!
  - If k bits used, range becomes $[-2^{k-1},2^{k-1}-1]$

# Arithmetic operations

- Addition, multiplication: work with bits
- Substraction: A-B= A+ (-B)
- Division: Is not easy to do,  rounded towards zero

- Overflow: If adding two large numbers, sign bit can be treated as high order bit, in some (old) architectures resulted in a negative number!

# Floating-point system

- Sign
- Exponent
- Mantissa=Significand

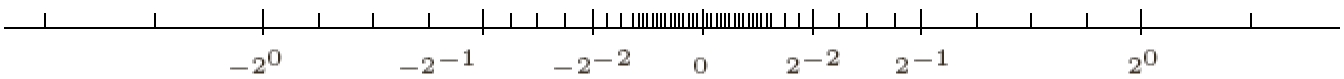
| sign (1bit) | Exponent (11bits) | Mantissa (52 bits) |
|---|---|---|

- $\pm 0.d1d2\ldots d_p * b^e$
- Values to be presented approx $[-10^{300}, 10^{300}]$

# Floating-point system

- Real numbers are not the same as computer floats!! But they are instead rounded towards floats…

- **Ex.**: Assume there are only 5 digits for mantissa, number 4.0000567 becomes $0.40000*10^1$
- In most computer systems, base is 2, not 10, but the problem remains.
- How computer floats are distributed




- Very dense from -1 to 1, density decreases. If N points for numbers having exponent $*10^1$, also N points for numbers with exponent $10^3$

# Special floating-point numbers

- Usually maximal allowed number in the exponent is one unit less than it could be
- ± Inf : exponent is $exp_{max}$+1 and mantissa is zero
- NaN: exponent is $exp_{max}$+1 and mantissa is nonzero

**Overflow/underflow:**

- $10^{200}*10^{200}$=+Inf
- $10^{400}/10^{400}$=Inf/Inf=NaN
- $10^{-200}/10^{200}$=0
- $0*10^{400}$=?
- If x:=x+1, the cycle will NOT converge to +Inf !

# Operations on real numbers and floats

- Since floats are not the same as real numbers, usual mathematical laws may break down.

**Ex**: 1/3+1/3=2/3 where in computer
0.33333+0.33333≠0.66667

- However, most of computer systems are designed to make arithmetic operations as correct as possible

# Operations on real numbers and floats

## More problems with floats:

- Results of X*Y and X+Y do not result in a true value (overflow for ex.)
- A+X=B+X but B ≠A
- A+X=X but A+Y ≠Y
- A+X=X but X-X ≠a
- -> BE CAREFUL WHEN YOU COMPARE NUMBERS IN COMPUTER!
- Associativity and distributivity may not hold

# Summation problem

- Recall x:=x+1, similar problem may occur when summing arbitrary data series.

Solution A:

1. Sort the numbers ascending
2. Sum up numbers in this order

Solution B (similar magnitude):

1. Sum numbers pairwise, having n numbers obtain n/2 numbers
2. Continue until you have 1 number

# Cancellation

- If computing exponent using Taylor series

$e^x=1+x+x^2/2+x^3/6+\ldots$

- If x=20, the formula works fine
- If x=-20, the error is almost 100%

Main reason: varying sign of the terms

Cancellation = adding two numbers almost equal magnitude, opposite sign

Here, effects of small cancellations accumulated

# Computer arithmetics in matrix computations

- Very often, one needs to solve (ex: regression)

$$\mathbf{Ax=b}$$

**A** matrix
**X** unknown vector
**b** vector of scalars

**Requirement**

- The algorithm solving the problem should be numerically stable

# Linear regression models

Minimize

$$S(\beta_0, \beta_1, \ldots, \beta_p) = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_{1i} - \ldots - \beta_p x_{pi})^2$$

Solve the equation system $\frac{\partial S}{\partial \beta_0} = \ldots = \frac{\partial S}{\partial \beta_p} = 0$ that can be written $X^T X \beta = X^T Y$

where $X = \begin{pmatrix} 1 & x_{11} & . & . & x_{p1} \\ 1 & x_{12} & . & . & x_{p2} \\ . & . & . & . & . \\ . & . & . & . & . \\ 1 & x_{1n} & . & . & x_{pn} \end{pmatrix}$ is a matrix of observed x-variables

# Smoothing splines

- Minimize

$$S(\beta_0, \beta_1, ..., \beta_p) = \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \int \{f''(x)\}^2 dx$$

- Solution is $f(x) = \sum_{i=1}^{n} N_j(x)\theta_j$

where θ is found from $\left(N^T N + \lambda \Omega_N\right)\theta = N^T Y$

# Solving system of linear equations

- Important to be able to solve **Ax=b** numerically
- Aware of computer arithmetics! (recall a+x=x)

- Condition number
  - Original system $Ax = b$
  - Perturbed system $A\tilde{x} = \tilde{b}$ $\tilde{x} = x + \delta x$ $\tilde{b} = b + \delta b$

- Solution is good if small perturbation of *b* causes small perturbation of *x*, and since

$$\frac{||\delta x||}{||x||} \leq ||A||\,||A^{-1}||\frac{||\delta b||}{||b||}$$

# Solving system of linear equations

Condition number

$$\kappa(A) = \|A\| \, \|A^{-1}\|$$

Properties:

- Large condition number is a bad signal, but does not imply ill-conditioning
- If norm is $L_2$ then k is ratio of max.eigenvalue and min.eigenvalue
- Since $\kappa_2(A^{\mathrm{T}}A) = \kappa_2^2(A) \geq \kappa_2(A)$ problems in regression fitting may appear

# Solving system of linear equations

## Resolving ill-conditioning

- Rescaling the variables (columns)

- Using decompositions
  - QR, Cholesky, SVD,…

Example: $Ax = b \ \rightarrow LL^T x = b$

- Solve $Ly = b$
- Solve $L^T x = y$