# Matrices and Statistics with Applications
# A Numerical Point of View

L. Eldén

LiU

September 2020

# Large Data Sets

*"Gone are the days when statisticians used to work with fixed, laboriously compiled and labelled datasets."*

*Christos Dimitrakakis, Computer Science and Engineering, Chalmers, 2014*

# Large Data Sets

- Development of computers and computation
- Scientific research involves automated data collection
- Very large and highly inhomogeneous data sets
- Development of new statistical/computational methods for data analysis
- Necessary to use efficient and reliable numerical methods

# Computational Statistics

- Built on the mathematical theory and methods of statistics
- Includes
  - visualization
  - statistical computing
  - Monte Carlo methods
  - Exploratory methods
- Development of computationally-intensive methods for mining and visualization of large, non-homogeneous, multi-dimensional datasets to discover knowledge in the data
- Probability models
- Statements of confidence or of probability
- Model building and evaluation

# Research Areas I

- Techniques for discovering structure in data with emphasis on large-dimensional datasets
  - Exploratory or visual
  - estimation, clustering, or classification
- Statistical learning
- Methods for analysis of extremely large datasets (large number of observations or large number of dimensions)
- Computationally-intensive methods of analysis (Monte Carlo methods or resampling methods)
- Simulation methods
- Methods for statistical modeling
  - Classical statistical parametric models, semiparametric models or nonparametric models
  - Frequentist or Bayesian approach

# Research Areas II

- Explicitly or implicitly described models, e.g via differential equations, especially Stochastic Differential Equations.

- Numerical methods for statistical analysis (statistical computing)
- Methods for statistical problems that have a major "computer science" aspect (record matching, for example)

# Why does one need scientific computing?

- Large data sets: efficient, stable and robust algorithms (very often from linear algebra)
- Understanding of the influence of floating point round off errors
- Algorithms from state-of-the-art software libraries, often packaged in high level programming systems (MATLAB, R)

# Are floating point errors a major concern?

Answer, in general: NO!

## IEEE double precision arithmetic

- 64 bit format
- 52 bit mantissa $\Longrightarrow$ relative precision $\approx 10^{-16}$

# How fine is the IEEE floating-point number system?

Here is one way to see the answer[1]:

- According to molecular physics, there are approximately $3 \cdot 10^8$ molecules per meter in a gas at atmospheric conditions – essentially the cube root of Avogadro's number.
- The circumference of the earth is $4 \cdot 10^7$ meters, so in a circle around the earth, there are around $10^{16}$ molecules.
- In IEEE double precision arithmetic, there are $2^{52}$ numbers between 1 and 2, which is also about $10^{16}$.
- So if we put a giant circle around the earth with a distance coordinate ranging from 1 to 2, the spacing of the floating-point numbers along the circle will be about the same as the spacing of the air molecules.

[1] due to Nick Trefethen, Oxford

# IEEE double precision is usually sufficient

BUT: If one performs computations in a bad way, one can lose all the precision.

Example: Computation of variance using a one-pass method.

Warning: Sometimes computations for ultra-large problems (e.g. in AI) are done using reduced precision arithmetic (for speed). Many standard algorithms give sufficient accuracy, perhaps after some tweaking.

# Example: Regression

Let $y$ be a vector of observed quantities and a matrix $X$ of explaining variables.
Regression model:

$$y \sim X\beta + \epsilon,$$

where $\epsilon$ is a random variable.
Least squares method:
Determine the regression coefficients $\beta$ by solving

$$\min_{\beta} \|y - X\beta\|_2$$

# Regression: numerical aspects

Solve

$$\min_{\beta} \|y - X\beta\|_2$$

---

- Algorithms, efficiency and stability:
  1. Normal equations: solve $X^T X\beta = X^T y$
  2. QR decomposition: $X = QR$, where $Q$ is orthogonal and $R$ is triangular
  3. Singular value decomposition: $X = U\Sigma V^T$, where $U$ and $V$ are orthogonal and $\Sigma$ is diagonal
- What to do if the explaining variables (columns of $X$) are (almost) collinear (linearly dependent)?
- How to solve large problems? How large is large?
- How to solve large structured (sparse, Toeplitz, etc.) problems?

# How large is large?

Answer: Depends on the hardware.
Today (assume for simplicity that $X$ is $n \times n$)

- Small: $n \leq 1000$
  Takes less than 5 (say) seconds on any computer.
  Standard methods
- Medium: $1000 \leq 10000$
  Takes minutes.
  Standard methods
- Large: $n > 10^4$
  Takes minutes-hours-days. Often only possible to solve if the problem is structured (sparse, Toeplitz, etc.)
  Iterative methods, sometimes non-standard
- Ultralarge: $n \approx 10^9$ and larger
  Only special structures and special methods.

# How large is large? Example: SVD

Answer: Depends on the hardware.
Computation of SVD (Matlab on desktop computer):

| dimension | time (s) | time 2002 |
|---|---|---|
| 500 | 0.05 | 12 |
| 1000 | 0.5 | 129 |
| 5000 | 46 | |

R is based on the same subroutine library: LAPACK

# Programming environments

High quality algorithms packaged with graphics and other tools

- MATLAB (Statistics toolbox)
- R

**Two commandments of computational statistics:**

1. Always use a high quality programming environment!
2. Never write your own code for standard tasks like solving a linear system, QR, singular value decomposition, etc.!