Machine Learning for Computational Biology

Nikolay Oskolkov, Lund University, NBIS SciLifeLab, Sweden

AFI course, 16.02.2026

@NikolayOskolkov

GitHub

github.com/NikolayOskolkov

Image generated at www.canva.com

Brief introduction: who am I

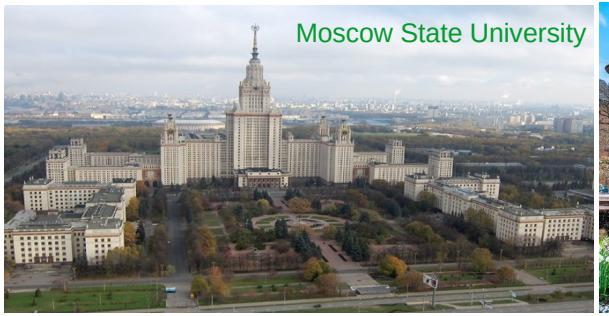
2007 PhD in theoretical physics in Moscow, Russia

2011 medical genomics at Lund University, Sweden

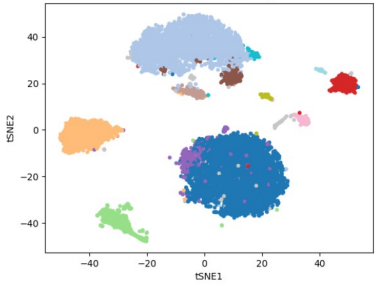
2016 bioinformatician at NBIS SciLifeLab, Sweden

2025 Metabolic Research Group leader, LIOS, Latvia

50+ publications; h-index = 25; 4,000+ citations



Single cell



Biomedical data integration

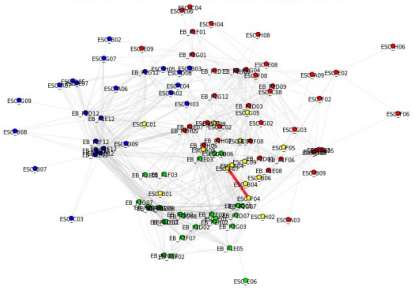
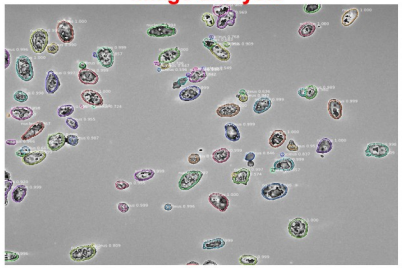
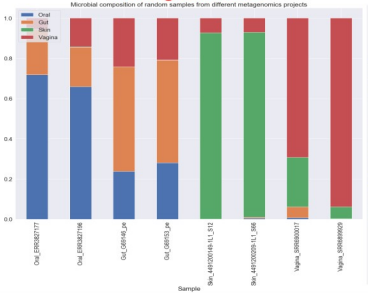


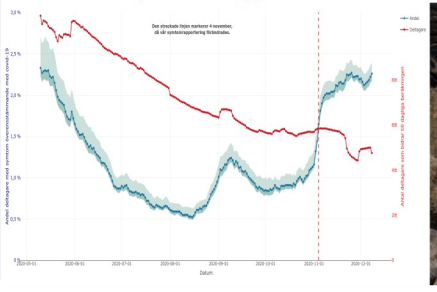
Image analysis



Metagenomics



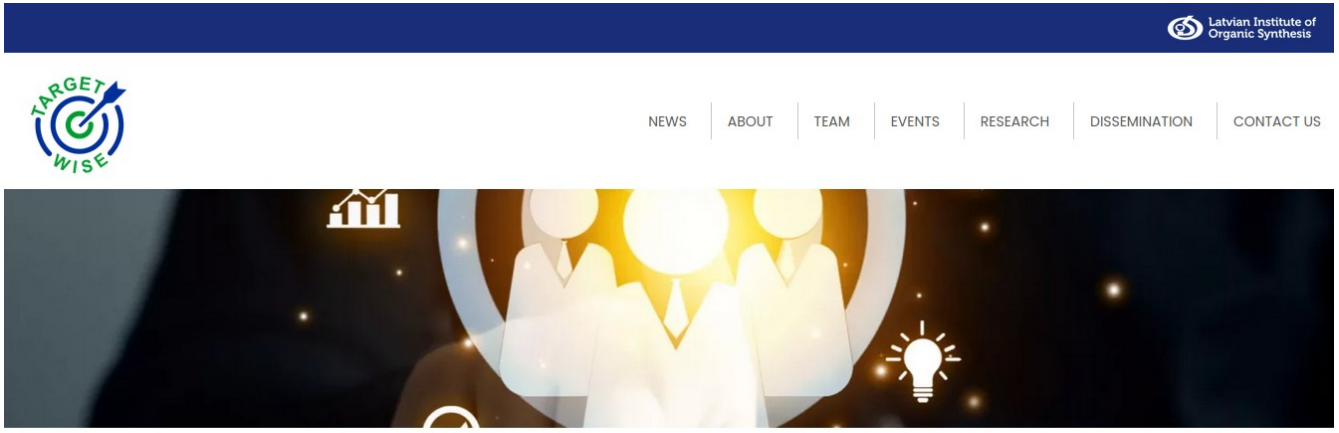
Epidemiology



Ancient microbiome

Metabolic Research Group at LIOS

Latvian Institute of Organic Synthesis



PhD. Nikolay Oskolkov
Group Leader (PI) of the Metabolic Research Group

1 more postdoctoral fellow and 1 PhD student to be hired

If you know anyone who might be interested, please contact me!

Kristina Grausa, PhD sudent in Metabolic Research Group

Daniel Rivas, MD, PhD in AI, postdoctoral fellow in Metabolic Research Group

Seminar overview

Plan for today's seminar:

14.00 - 15.20: Session1: Introduction to statistical analysis

15.20 – 15.30: Break

15.30 – 16.50: Session2: From statistics to machine learning

16.50 – 17.00: Break

17.00 - 18.20: Session3: From linear models to neural networks

18.20 – 18.30: Break

18.30 - 19.50: Session4: Decision tree-based machine learning

19.50 – 20.00: Questions + Discussion

Housekeeping rules

- To ask question please raise your hand, unmute yourself and ask. You can also ask questions in the zoom chat or Instats forum for this seminar.

- Please keep your camera on as much as possible for better contact and communication. Your face will not be recorded.

- The seminar is focused on lectures, however every session (except first one) will be accompanied by a ~20 mins practical (the notebooks are provided), where I will use html-versions of Rmarkdown and Jupyter notebooks to go through command lines with my explanations.

- The material is based on data and problems from computational biology, however the concepts discussed are general and can be applied for other types of data.

Machine Learning for Computational Biology

Nikolay Oskolkov, Lund University, NBIS SciLifeLab, Sweden

Instats seminar, 16.09.2024

Session 1: Introduction to Statistical Analysis in Computational Biology

@NikolayOskolkov

GitHub

github.com/NikolayOskolkov

Image generated at www.canva.com

Session content

Topics we'll cover in this session:

1) High-dimensional biological data and curse of dimensionality

2) Data-driven choice of statistical analysis

3) Basics of Frequentist statistics: pros and cons of p-value

4) LASSO regularization to overcome the curse of dimensionality

5) Basics of Bayesian statistics: priors as regularizations

Biological data are high dimensional

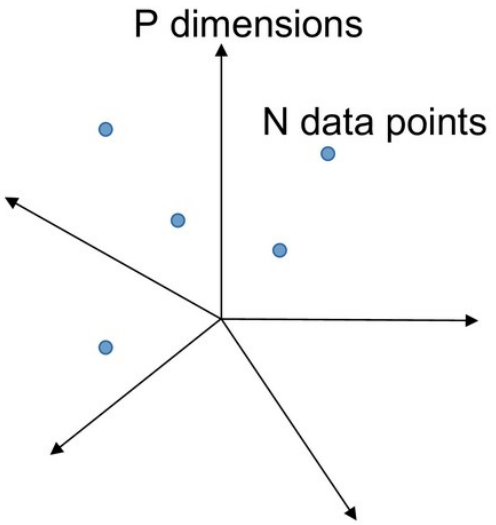
Statistical observations:
e.g. samples, cells etc.

N

P

0	3	1	0	2	3	8	1	1	3
1	1	0	0	7	1	2	2	3	3
1	2	2	0	0	6	7	1	2	2
1	2	3	10	0	4	6	1	0	5
3	2	2	1	4	3	2	1	6	0
7	4	4	5	3	9	6	1	6	1
7	1	1	5	2	8	9	1	3	6
5	0	1	6	2	0	0	0	1	5
1	6	3	3	4	6	2	0	1	1
1	2	2	4	1	1	3	0	8	2

Features: genes, proteins,
microbes, metabolites etc.



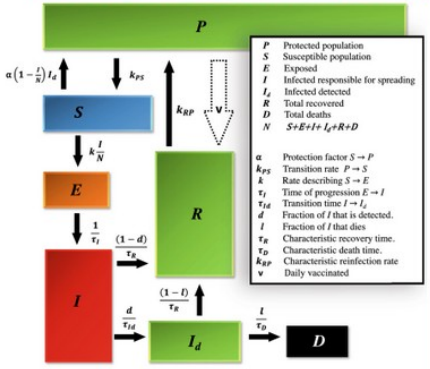
High Dimensional Data:
P >> N

For a robust statistical analysis, one should properly “sample” the P-dimensional space, hence large sample size is required, N >> P

Types of data analysis

Biology / Biomedicine

Mathematical modeling



$N \longrightarrow 0$

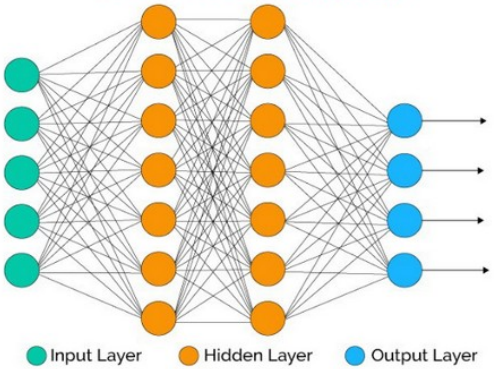
Bayesianism

$N << P$

Frequentism

$N \approx P$

Machine Learning



$N >> P$

Hypothesis-driven → **Data-driven**

Amount of Data

The Curse of Dimensionality

Ex.1

$$Y = \alpha + \beta X$$

$$\beta = (X^T X)^{-1} X^T Y$$

$$(X^T X)^{-1} \sim \frac{1}{\det(X^T X)} \cdots \to \infty, \quad n << p$$

Ex.2 $$E[\hat{\sigma}^2] = \frac{n-p}{n}\sigma^2$$

Biased Maximum Likelihood variance estimator at n << p

Some peculiarities of Frequentist statistics

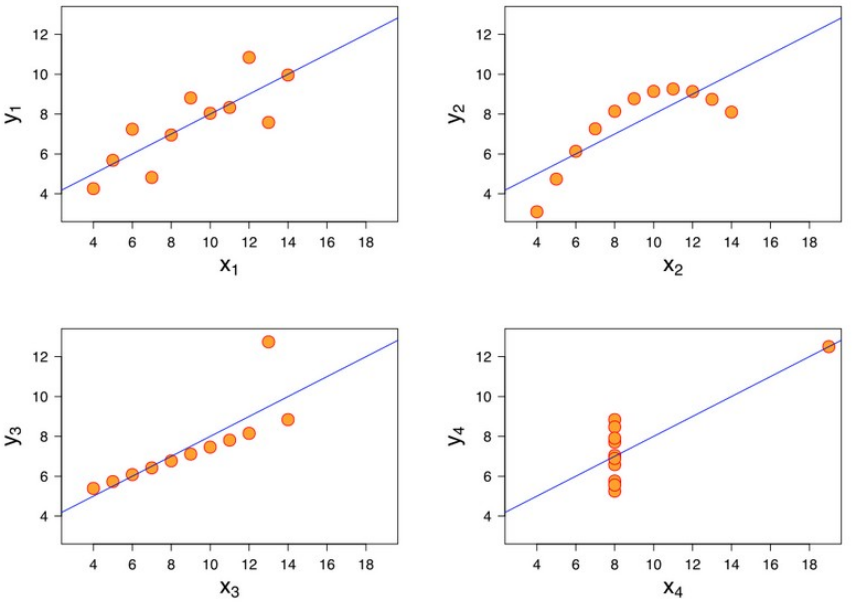
- based on Maximum Likelihood principle
- focus ~~too much~~ on summary statistics

$$\mathrm{L}\left(\mathrm{x_i} \mid \mu, \sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp^{-\frac{\sum_{\mathrm{i}=1}^{\mathrm{N}} (\mathrm{x_i} - \mu)^2}{2\sigma^2}}$$

$$\frac{\partial \mathrm{L}\left(\mathrm{x_i} \mid \mu, \sigma^2\right)}{\partial \mu} = 0; \quad \frac{\partial \mathrm{L}\left(\mathrm{x_i} \mid \mu, \sigma^2\right)}{\partial \sigma^2} = 0$$

$$\mu = \frac{1}{N} \sum_{i=0}^{N} x_i \; - \text{mean estimator}$$

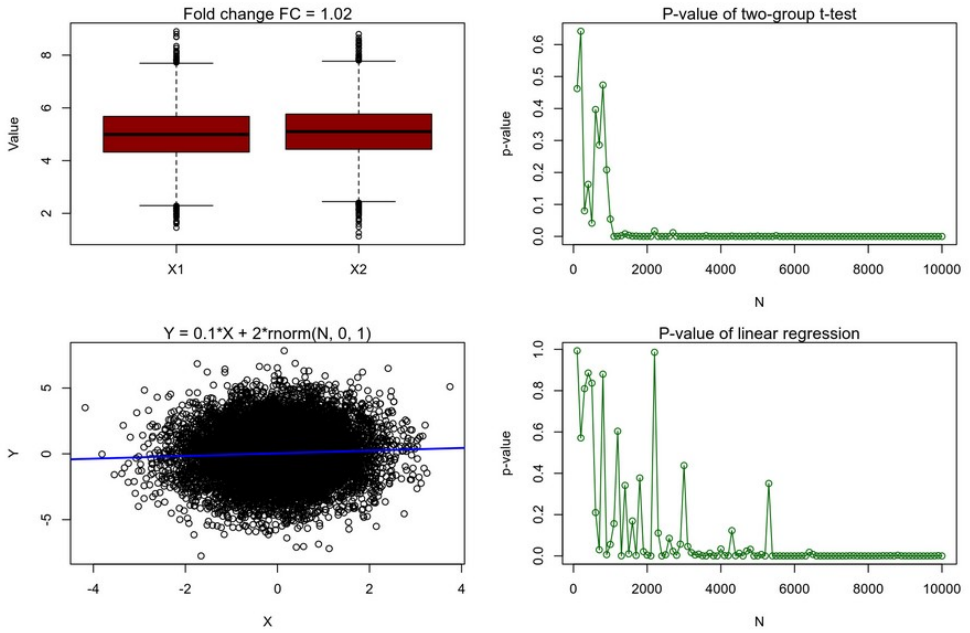
$$\sigma^2 = \frac{1}{N} \sum_{i=0}^{N} (x_i - \mu)^2 \; - \text{variance estimator}$$



Summary statistics do not always reasonbly describe data (example: Anscombes quartet)

Frequentist statistics: focus ~~to much~~ on p-values

```r
FC<-1.02; x_mean<-5; x_sd<-1; N_vector<-seq(from=100,to=10000,by=100); pvalue_t<-vector(); pvalue_lm<-vector()
for(N in N_vector)
{
  x1 <- rnorm(N, x_mean, x_sd); x2 <- rnorm(N, x_mean*FC, x_sd)
  t_test_res<-t.test(x1, x2); pvalue_t <- append(pvalue_t, t_test_res$p.value)

  x <- rnorm(N, 0, 1); y <- 0.1*x+2*rnorm(N, 0, 1)
  lm_res <- summary(lm(y~x)); pvalue_lm <- append(pvalue_lm, lm_res$coefficients[2,4])
}
par(mfrow=c(2,2)); par(mar = c(5, 5, 1, 1))
boxplot(x1, x2, names=c("X1","X2"), ylab="Value", col="darkred"); mtext("Fold change FC = 1.02")
plot(pvalue_t~N_vector,type='o',xlab="N",ylab="p-value",col="darkgreen"); mtext("P-value of two-group t-test")
plot(y~x, xlab="X", ylab="Y"); abline(lm(y~x), col="blue", lwd=2); mtext("Y = 0.1*X + 2*rnorm(N, 0, 1)")
plot(pvalue_lm~N_vector,type='o',xlab="N",ylab="p-value",col="darkgreen"); mtext("P-value of linear regression")
```



Questionable whether p-value is a best metric for ranking features (biomarkers)

Frequentist statistics struggles with high-dimensional data

```r
n <- 20 # number of samples
p <- 2  # number of features / dimensions
Y <- rnorm(n)
X <- matrix(rnorm(n * p), n, p)
summary(lm(Y ~ X))
```

```
Call:
lm(formula = Y ~ X)

Residuals:
    Min      1Q  Median      3Q     Max
-2.0522 -0.6380  0.1451  0.3911  1.8829

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.14950    0.22949   0.651    0.523
X1          -0.09405    0.28245  -0.333    0.743
X2          -0.11919    0.24486  -0.487    0.633

Residual standard error: 1.017 on 17 degrees of freedom
Multiple R-squared:  0.02204,   Adjusted R-squared:  -0.09301
F-statistic: 0.1916 on 2 and 17 DF,  p-value: 0.8274
```

Going to higher dimensions →

```r
n <- 20 # number of samples
p <- 10 # number of features / dimensions
Y <- rnorm(n)
X <- matrix(rnorm(n * p), n, p)
summary(lm(Y ~ X))
```

```
Call:
lm(formula = Y ~ X)

Residuals:
    Min      1Q  Median      3Q     Max
-1.0255 -0.4320  0.1056  0.4493  1.0617

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.54916    0.26472   2.075   0.0679 .
X1           0.30013    0.21690   1.384   0.1998
X2           0.68053    0.27693   2.457   0.0363 *
X3          -0.10675    0.26010  -0.410   0.6911
X4          -0.21367    0.33690  -0.634   0.5417
X5          -0.19123    0.31881  -0.600   0.5634
X6           0.81074    0.25221   3.214   0.0106 *
X7           0.09634    0.24143   0.399   0.6992
X8          -0.29864    0.19004  -1.571   0.1505
X9          -0.78175    0.35408  -2.208   0.0546 .
X10          0.83736    0.36936   2.267   0.0496 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.8692 on 9 degrees of freedom
Multiple R-squared:  0.6592,    Adjusted R-squared:  0.2805
F-statistic: 1.741 on 10 and 9 DF,  p-value: 0.2089
```

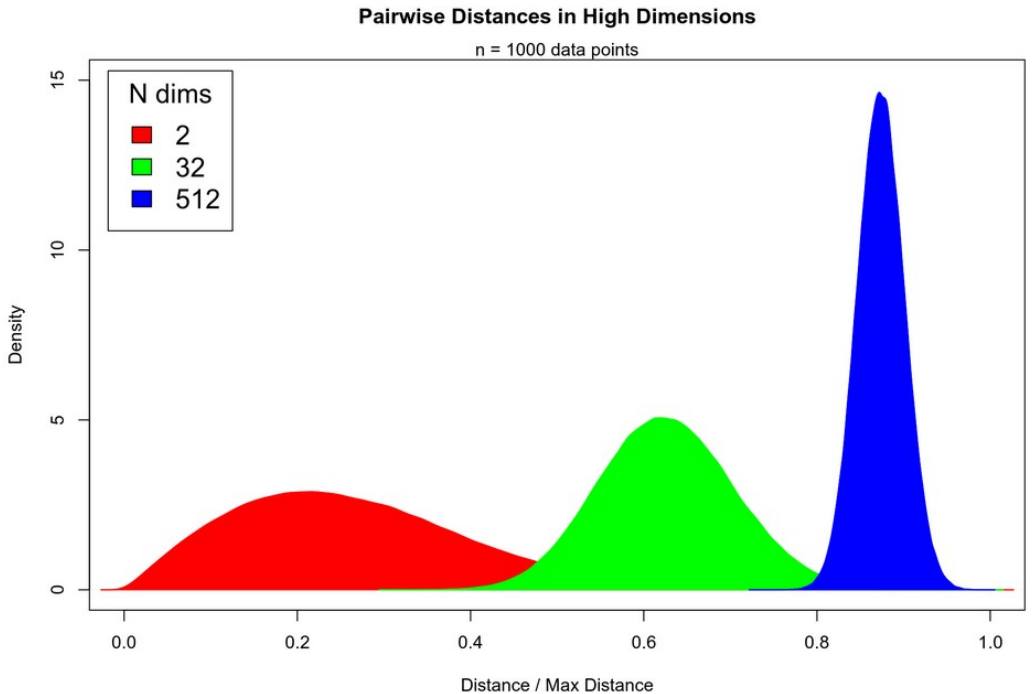
Going to even higher dimensions →

```r
n <- 20 # number of samples
p <- 20 # number of features / dimensions
Y <- rnorm(n)
X <- matrix(rnorm(n * p), n, p)
summary(lm(Y ~ X))
```

```
Call:
lm(formula = Y ~ X)

Residuals:
ALL 20 residuals are 0: no residual degrees of freedom!

Coefficients: (1 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.34889        NaN     NaN      NaN
X1           0.66218        NaN     NaN      NaN
X2           0.76212        NaN     NaN      NaN
X3          -1.35033        NaN     NaN      NaN
X4          -0.57487        NaN     NaN      NaN
X5           0.02142        NaN     NaN      NaN
X6           0.40290        NaN     NaN      NaN
X7           0.03313        NaN     NaN      NaN
X8          -0.31983        NaN     NaN      NaN
X9          -0.92833        NaN     NaN      NaN
X10          0.18091        NaN     NaN      NaN
X11         -1.37618        NaN     NaN      NaN
X12          2.11438        NaN     NaN      NaN
X13         -1.75103        NaN     NaN      NaN
X14         -1.55073        NaN     NaN      NaN
X15          0.01112        NaN     NaN      NaN
X16         -0.50943        NaN     NaN      NaN
X17         -0.47576        NaN     NaN      NaN
X18          0.31793        NaN     NaN      NaN
X19          1.43615        NaN     NaN      NaN
X20               NA         NA      NA       NA

Residual standard error: NaN on 0 degrees of freedom
Multiple R-squared:      1, Adjusted R-squared:    NaN
F-statistic:   NaN on 19 and 0 DF,  p-value: NA
```

Equidistant points in high dimensions

```r
n <- 1000; p <- c(2, 32, 512); pair_dist <- list()
for(i in 1:length(p)) {
  X <- matrix(rnorm(n * p[i]), n, p[i])
  pair_dist[[i]] <- as.vector(dist(X));
  pair_dist[[i]] <- pair_dist[[i]] / max(pair_dist[[i]])
}
```



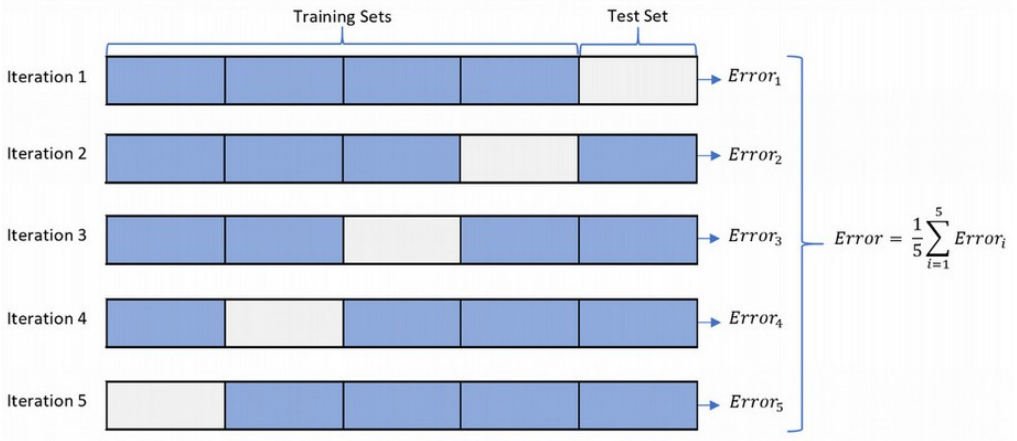
- Data points in high dimensions:
  - move **away** from each other
  - become **equidistant** and similar
- Impossible to see differences between cases and controls

Regularizations: LASSO

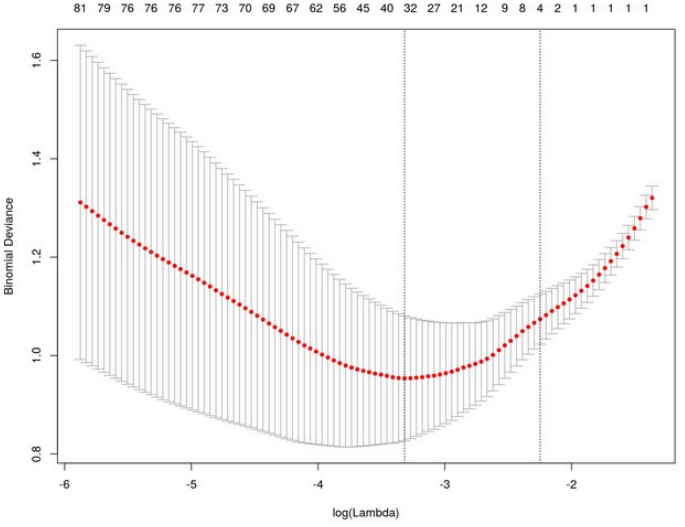
$$Y = \beta_1 X_1 + \beta_2 X_2 + \epsilon$$

$$\text{OLS} = (Y - \beta_1 X_1 - \beta_2 X_2)^2$$

$$\text{Penalized OLS} = (Y - \beta_1 X_1 - \beta_2 X_2)^2 + \lambda(|\beta_1| + |\beta_2|)$$



Cross-validation is a standard way to tune model hyperparameters such as λ in LASSO



Regularizations are priors in Bayesian statistics

$$Y = \beta_1 X_1 + \beta_2 X_2 + \epsilon; \quad Y \sim N(\, \beta_1 X_1 + \beta_2 X_2, \sigma^2\,) \equiv \mathrm{L}\,(\,\mathrm{Y} \mid \beta_1, \beta_2\,)$$

- **Maximum Likelihood** principle: maximize probability to observe data given parameters:

$$\mathrm{L}\,(\,\mathrm{Y} \mid \beta_1, \beta_2\,) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp^{-\frac{(\mathrm{Y} - \beta_1 \mathrm{X}_1 - \beta_2 \mathrm{X}_2)^2}{2\sigma^2}}$$

- **Bayes theorem**: maximize posterior probability of observing parameters given data:

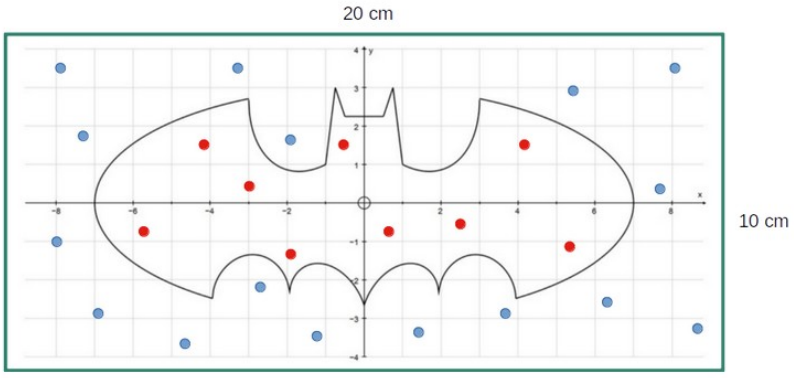
$$\mathrm{Posterior}(\mathrm{params} \mid \mathrm{data}) = \frac{\mathrm{L}(\mathrm{data} \mid \mathrm{params}) * \mathrm{Prior}(\mathrm{params})}{\int \mathrm{L}(\mathrm{data} \mid \mathrm{params}) * \mathrm{Prior}(\mathrm{params})\, \mathrm{d}(\mathrm{params})}$$

$$\mathrm{Posterior}(\,\beta_1, \beta_2 \mid \mathrm{Y}\,) \sim \mathrm{L}\,(\,\mathrm{Y} \mid \beta_1, \beta_2\,) * \mathrm{Prior}(\beta_1, \beta_2) \sim \exp^{-\frac{(\mathrm{Y} - \beta_1 \mathrm{X}_1 - \beta_2 \mathrm{X}_2)^2}{2\sigma^2}} * \exp^{-\lambda(|\beta_1| + |\beta_2|)}$$

$$-\log\left[\mathrm{Posterior}(\,\beta_1, \beta_2 \mid \mathrm{Y}\,)\right] \sim (\mathrm{Y} - \beta_1 \mathrm{X}_1 - \beta_2 \mathrm{X}_2)^2 + \lambda(|\beta_1| + |\beta_2|)$$

Markov Chain Monte Carlo (MCMC): introduction

- Integration via Monte Carlo sampling

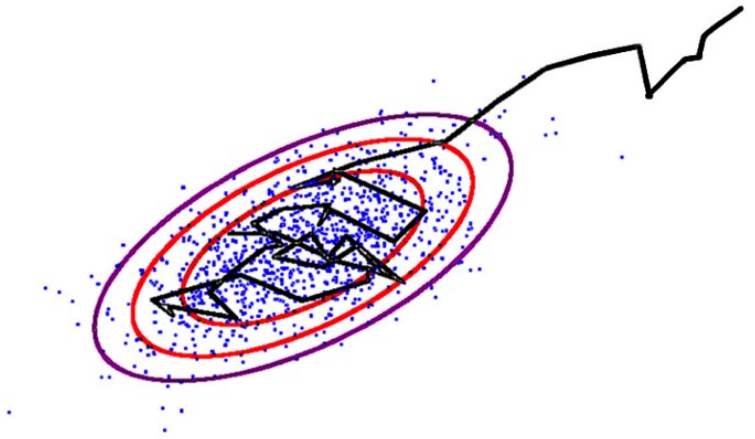


$$I = 2\int_{2}^{4} x dx = 2\frac{x^2}{2}\Big|_{2}^{4} = 16 - 4 = 12$$

```r
f <- function(x){return(2*x)}; a <- 2; b <- 4; N <- 10000; count <- 0
x <- seq(from = a, to = b, by = (b-a) / N); y_max <- max(f(x))
for(i in 1:N)
{
  x_sample <- runif(1, a, b); y_sample <- runif(1, 0, y_max)
  if(y_sample <= f(x_sample)){count <- count + 1}
}
paste0("Integral by Monte Carlo: I = ", (count / N) * (b - a) * y_max)
```

```
[1] "Integral by Monte Carlo: I = 11.9248"
```

- Markov Chain Monte Carlo (MCMC)

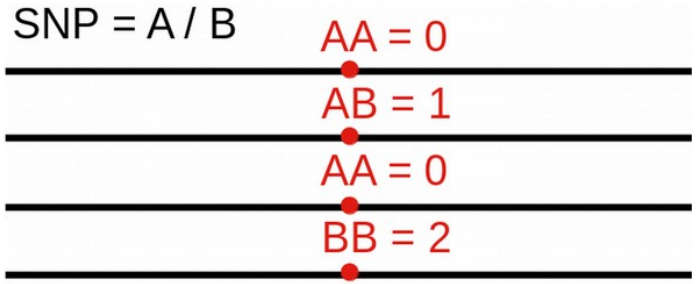


$$\text{Hastings ratio} = \frac{\text{Posterior}\left(\text{params}_{\text{next}} \mid \text{data}\right)}{\text{Posterior}\left(\text{params}_{\text{previous}} \mid \text{data}\right)}$$

- If Hastings ratio > *u* [0, 1], then **accept**, else **reject**
- Hastings ratio does not contain the intractable integral from Bayes theorem

Markov Chain Monte Carlo (MCMC) from scratch in R

- Example from population genetics

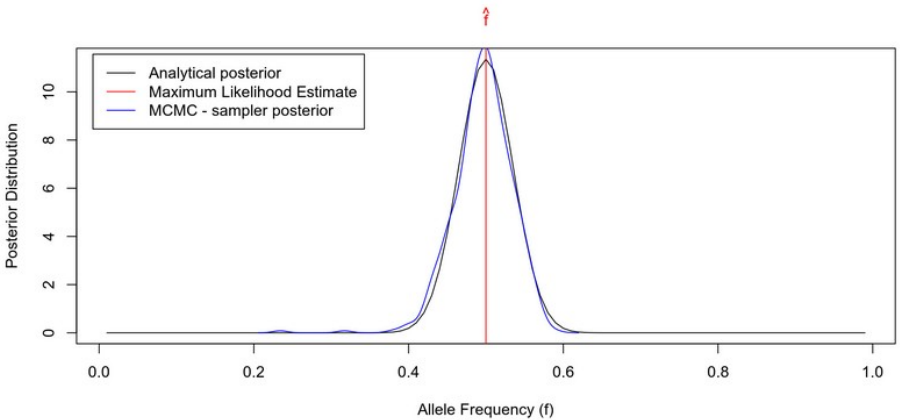



```r
N <- 100; n <- c(25, 50, 25) # Observed genotype data for N individuals
f_MLE <- (n[2] + 2*n[3]) / (2 * sum(n)) # MLE of allele frequency

# Define log-likelihood function (log-binomial distribution)
LL <- function(n, f){return((n[2] + 2*n[3])*log(f) + (n[2] + 2*n[1])*log(1-f))}
# Define log-prior function (log-beta distribution)
LP <- function(f, alpha, beta){return(dbeta(f, alpha, beta, log = TRUE))}

# Run MCMC Metropolis - Hastings sampler
f_poster <- vector(); alpha <- 0.5; beta <- 0.5; f_cur <- 0.1 # initialization
for(i in 1:1000)
{
  f_next <- abs(rnorm(1, f_cur, 0.1)) # make random step for allele frequency

  LL_cur <- LL(n, f_cur); LL_next <- LL(n, f_next)
  LP_cur <- LP(f_cur, alpha, beta); LP_next <- LP(f_next, alpha, beta)
  hastings_ratio <- LL_next + LP_next - LL_cur - LP_cur

  if(hastings_ratio > log(runif(1))){f_cur <- f_next}; f_poster[i] <- f_cur
}
```

$$L(n \mid f) = \prod_g \left[\binom{2}{g} f^g (1-f)^{2-g} \right]^{n_g}$$

$$\frac{\partial \log[L(n|f)]}{\partial f} = 0 \Rightarrow \hat{f} = \frac{n_1 + 2n_2}{2(n_0 + n_1 + n_2)}$$

$$\text{Prior}(\text{f}, \alpha, \beta) = \frac{1}{\text{B}(\alpha, \beta)} \text{f}^{\alpha-1} (1-\text{f})^{\beta-1}$$



Session summary

Take home messages of the session:

1) Biological data are high dimensional and this should be taken into account when performing statistical analyses

2) The choice of analysis (e.g. Frequentist, Bayesian or machine learning) is driven by particular data type

3) LASSO and Bayesian statistics are recommended for analysis of high-dimensional biological data

4) Penalized regression can be viewed as a bridge from traditional statistics to machine learning

National Bioinformatics Infrastructure Sweden (NBIS)

Knut och Alice Wallenbergs Stiftelse

Vetenskapsrådet

Lunds Universitet