LINKÖPING UNIVERSITY

# Advanced Computational Statistics – Spring 2025
# Assignment for Lecture 1

Frank Miller, frank.miller@liu.se,
Department of Computer and Information Science, Linköpings University

March 12, 2025

Perform the solutions individually and send your report **until March 31** by email to me. Try to keep this deadline. However, if you have problems with it, there will be a final deadline on September 30 for all assignments.

Please send me one **pdf-file** with your report (alternatively, Word is ok, too), and additionally, please send me your code in one separate **plain-text file** (an R-markdown, .rmd, is possible but not required).

## Problem 1.1

Let

$$g(x, y) = -x^2 + 10y - 2y^3 + \frac{1}{2}x^2y.$$

a. Plot the function with a contour plot or a 3-dimensional plot to visualise the function.

b. Compute the gradient analytically. Set the gradient to 0 and solve the equations analytically to identify candidates for maxima, minima, and saddle points.

c. Compute the Hessian matrix analytically. Determine if it is positive, negative, or indefinite in the candidate points (if you want to calculate eigenvalues, you can use software for it). What does this mean for your candidate points identified in b.?

## Problem 1.2

Three doses (0.1, 0.3, and 0.9 $g$) of a drug and placebo (0 $g$) are tested in a study. Afterward, a dose-dependent event is recorded. The data of $n = 10$ subjects is shown in Table 1; $x_i$ is the dose in gram; $\tilde{x}_i$ is the dose with unit changed to milligram; $y_i = 1$ if the event occurred, $y_i = 0$ otherwise.

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $x_i$ | in $g$ | 0 | 0 | 0 | 0.1 | 0.1 | 0.3 | 0.3 | 0.9 | 0.9 | 0.9 |
| $\tilde{x}_i$ | in $mg$ | 0 | 0 | 0 | 100 | 100 | 300 | 300 | 900 | 900 | 900 |
| $y_i$ | | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |

Table 1: Data for Problem 1.2

You should fit a simple logistic regression

$$p(x) = P(Y = 1|x) = \frac{1}{1 + \exp(-\beta_0 - \beta_1 x)}.$$

to the data, i.e. estimate $\beta_0$ and $\beta_1$. One can show that the log likelihood is

$$g((\beta_0, \beta_1)) = \sum_{i=1}^{n} \left[ y_i \log\{(1 + \exp(-\beta_0 - \beta_1 x_i))^{-1}\} + (1 - y_i) \log\{1 - (1 + \exp(-\beta_0 - \beta_1 x_i))^{-1}\} \right]$$

and the gradient is

$$\mathbf{g}'(\mathbf{b}) = \sum_{i=1}^{n} \left\{ y_i - \frac{1}{1+\exp(-\beta_0 - \beta_1 x_i)} \right\} \begin{pmatrix} 1 \\ x_i \end{pmatrix}.$$

a. Write a program for an ML-estimator for $(\beta_0, \beta_1)$ using the steepest ascent method with a step-size halving line search (back-tracking). Problems could turn up later, e.g. when the log is taken from a value close to 0. If this becomes a problem, try to handle this issue.

b. Plot the log-likelihood function (contour plot) for the observations in Table 1. Compute the ML-estimator with your function both using gram data $(x_i, y_i)$ and using milligram data $(\tilde{x}_i, y_i)$. How many iterations were used in each case? Discuss reasons for that result. Note: to check your ML-solution, you might use a standard statistical function like the function `glm` in `R`.

c. Choose the quasi-Newton with BFGS or the Newton algorithm. Write your own optimisation program with the chosen algorithm. Run your program for the dataset in Table 1, using both $g$-data and $mg$-data. Report the number of iterations used and compare with the results from b.. If you have chosen the Newton algorithm, you need the Hessian matrix; write how you obtained it or if you used sources for it, please cite appropriately.

# Problem 1.3

We consider the quadratic two-dimensional function

$$g(\mathbf{x}) = -\frac{1}{2}\mathbf{x}^\top A\mathbf{x}, \quad \mathbf{x} \in I\!R^2.$$

with a symmetric and positive definite $2 \times 2$-matrix $A$. The function $g$ has a maximum at $(0,0)^\top$ and the gradient is $\mathbf{g}'(\mathbf{x}) = -A\mathbf{x}$. For $A$, we consider two different matrices with

$$A_1 = \begin{pmatrix} 8 & 1 \\ 1 & 8 \end{pmatrix}, \quad A_2 = \begin{pmatrix} 1 & 0 \\ 0 & 100 \end{pmatrix}.$$

a. Program your own function for accelerated steepest ascent with Polyak's momentum with fixed step size $\alpha$ (without backtracking) and fixed momentum parameter $\beta$. These two parameters should be options in your function such that you can test different options. Choose a stopping criterion such that you have correct results up to around 6 digits. Your function should report the number of iterations used.

b. Plot a contour plot for the two cases and compute with your programming language the eigenvalues of the matrices $A_i$ and the condition number $\kappa$. For both the steepest ascent and the accelerated steepest ascent, compute the optimal parameters and the best convergence rate $\rho$ in the two cases.

c. Run the steepest ascent method (i.e. use $\beta = 0$ in your function) using a starting value $\mathbf{x}^{(0)} = (-4, -2)^\top$. Use the optimal value and several other $\alpha$-values in a grid of 0.01 or 0.02 around the optimal value. Report if the algorithm successfully found the maximum and how many iterations were needed for each parameter value. Is the performance best for the theoretically best $\alpha$-value?

d. Run the accelerated steepest ascent method with Polyak's momentum. Use the optimal $\alpha, \beta$ and some values around them and report convergence and number of iterations. How much does acceleration improve performance?