# Optimisation algorithms in Statistics I, lecture 1

Frank Miller, Department of Statistics; Stockholm University

October 2, 2020

Stockholm University

# Course schedule

- Topic 1: **Gradient based algorithms**
  Lectures: October 2; Time 10-12, 13-15 (online, Zoom)
- Topic 2: **Stochastic gradient based algorithms**
  Lecture: October 13; Time: 9-12 (online, Zoom)
- Topic 3: **Gradient free algorithms**
  Lecture: October 23; Time 9-12 (online, Zoom)
- Topic 4: **Optimisation with restrictions**
  Lecture: November 6, Time 9-12 (online, Zoom)

Course homepage: http://gauss.stat.su.se/phd/oasi/

Includes reading material, lecture notes, assignments

# Optimisation in statistics

- Maximum Likelihood
- Minimising risk in (Bayesian) decision theory
- Minimising sum of squares (Least Squares Estimate)
- Maximising information in experimental design
- Machine learning

- Common problem in these examples:
  - $\boldsymbol{x}$ $p$-dimensional vector, $g$: $\mathbb{R}^p \to \mathbb{R}$ function
  - We search $\boldsymbol{x}^*$ with $g(\boldsymbol{x}^*) = \max g(\boldsymbol{x})$

# Optimisation in statistics

- Generic optimisation problem:
  - $\boldsymbol{x}$ $p$-dimensional vector, $g$: $\mathbb{R}^p \to \mathbb{R}$ function
  - We search $\boldsymbol{x}^*$ with $g(\boldsymbol{x}^*) = \max g(\boldsymbol{x})$

- Typical situation in machine learning situations, for Maximum Likelihood, and Least Squares problems: $g = \sum_{i=1}^{n} g_i$ with a (large) sample size $n$ where $g_i$: $\mathbb{R}^p \to \mathbb{R}$

- Note that a minimisation problem $g(\boldsymbol{x}^*) = \min g(\boldsymbol{x})$ can easily be solved by locking at $-g$ if an algorithm for maximisation is available

# Least squares estimation (LSE)

- We search a Least Squares estimate $\hat{\boldsymbol{\beta}}$ for **β** minimising the distance $g(\hat{\boldsymbol{\beta}}) = \|\hat{\boldsymbol{y}} - \boldsymbol{y}\|^2$ from $\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{\beta}}$ to **y=X β + ε**
- $g(\hat{\boldsymbol{\beta}}) = \left\|\boldsymbol{X}\hat{\boldsymbol{\beta}} - \boldsymbol{y}\right\|^2 = \left(\boldsymbol{X}\hat{\boldsymbol{\beta}} - \boldsymbol{y}\right)^T\left(\boldsymbol{X}\hat{\boldsymbol{\beta}} - \boldsymbol{y}\right)$
  $= \hat{\boldsymbol{\beta}}^T\boldsymbol{X}^T\boldsymbol{X}\hat{\boldsymbol{\beta}} - 2\hat{\boldsymbol{\beta}}^T\boldsymbol{X}^T\boldsymbol{y} + \boldsymbol{y}^T\boldsymbol{y}$
- Setting the derivative to 0, we get $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}$
  $(\frac{\partial f}{\partial \hat{\beta}} = 2\boldsymbol{X}^T\boldsymbol{X}\hat{\boldsymbol{\beta}} - 2\boldsymbol{X}^T\boldsymbol{y} = 0 \Rightarrow \boldsymbol{X}^T\boldsymbol{X}\hat{\boldsymbol{\beta}} = \boldsymbol{X}^T\boldsymbol{y})$
- Note that $g(\hat{\boldsymbol{\beta}}) = \left\|\boldsymbol{X}\hat{\boldsymbol{\beta}} - \boldsymbol{y}\right\|^2 = \sum_{i=1}^n \left(\boldsymbol{x}_i^T\hat{\boldsymbol{\beta}} - y_i\right)^2 = \sum_{i=1}^n g_i(\hat{\boldsymbol{\beta}})$
- Optimisation problem:
  - $\hat{\boldsymbol{\beta}}$ $p$-dimensional vector, $g$: $\mathbb{R}^p \to \mathbb{R}$ function
  - We search $\hat{\boldsymbol{\beta}}$ with $g(\hat{\boldsymbol{\beta}}) = \min g(\boldsymbol{b}) = \min \sum_{i=1}^n g_i(\boldsymbol{b})$
- Here, we do not need to iteratively compute this minimum since we have an algebraic solution $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}$

# Variations of least squares estimation

- While we have a solution for the LSE, we have no algebraic solution if we vary the problem
- Lasso estimate: $g(\hat{\boldsymbol{\beta}}) = \|\boldsymbol{X}\hat{\boldsymbol{\beta}} - \boldsymbol{y}\|^2 + \lambda\|\hat{\boldsymbol{\beta}}\|_1$
  $= \sum_{i=1}^{n}(\boldsymbol{x}_i\hat{\boldsymbol{\beta}} - y_i)^2 + \lambda\|\hat{\boldsymbol{\beta}}\|_1 = \sum_{i=1}^{n} g_i(\hat{\boldsymbol{\beta}})$
- $L_1$-estimation: $g(\hat{\boldsymbol{\beta}}) = \|\boldsymbol{X}\hat{\boldsymbol{\beta}} - \boldsymbol{y}\|_1 = \sum_{i=1}^{n}|\boldsymbol{x}_i\hat{\boldsymbol{\beta}} - y_i| = \sum_{i=1}^{n} g_i(\hat{\boldsymbol{\beta}})$
- Many further variations of estimates have been considered

- In all cases, we search $\hat{\boldsymbol{\beta}}$ with $g(\hat{\boldsymbol{\beta}}) = \min g(\boldsymbol{b}) = \min \sum_{i=1}^{n} g_i(\boldsymbol{b})$

- Recall: Norms for $\boldsymbol{x} = (x_1, \ldots, x_p)^T$: $\|\boldsymbol{x}\| = \|\boldsymbol{x}\|_2 = \sqrt{x_1^2 + \cdots + x_p^2}$ (Euklid),
  $\|\boldsymbol{x}\|_1 = |x_1| + \cdots + |x_p|$, $\|\boldsymbol{x}\|_\infty = \max\{|x_1|, \ldots, |x_p|\}$ (max-norm)

# Maximizing information of experimental designs

- Regression model $\mathbf{y}=\mathbf{X}\,\boldsymbol{\beta}+\boldsymbol{\varepsilon}$ (where $\boldsymbol{\varepsilon}$ has iid components)
- $\mathbf{X}$ design matrix (depends on choice of observational points)
- Covariance matrix of Least Squares estimate $\widehat{\boldsymbol{\beta}}$ is

$$\mathrm{Cov}\left(\widehat{\boldsymbol{\beta}}\right) = (\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X})^{-1}\cdot const$$

- Choose design of an experiment such that $\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X}$ "large"
- D-optimality: $g(\text{"}\mathbf{design}\text{"}) = \det(\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X})$
- We search $\mathbf{design}^{*}$ with $g(\mathbf{design}^{*}) = \max g(\mathbf{design})$

# Maximizing information of experimental designs

- Regression model $\mathbf{y}=\mathbf{X}\,\boldsymbol{\beta}+\boldsymbol{\varepsilon}$, $\mathrm{Cov}(\hat{\boldsymbol{\beta}}) = (\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X})^{-1}\cdot const$
- We search $\mathbf{design}^*$ with $g(\mathbf{design}^*) = \max g(\mathbf{design})$
- Example: cubic regression, $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \varepsilon$, *n* observations in each of following 4 points: $-1, -a, a, 1$. How should $a \in (0,1)$ be chosen?

$$\boldsymbol{X} = \begin{pmatrix} 1 & -1 & 1 & -1 \\ 1 & -a & a^2 & -a^3 \\ 1 & a & a^2 & a^3 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

$$g(a) = \det(\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X}) = \det(\boldsymbol{X}(a)^{\boldsymbol{T}}\boldsymbol{X}(a))$$

- We search $a^*$ with $g(a^*) = \max g(a)$

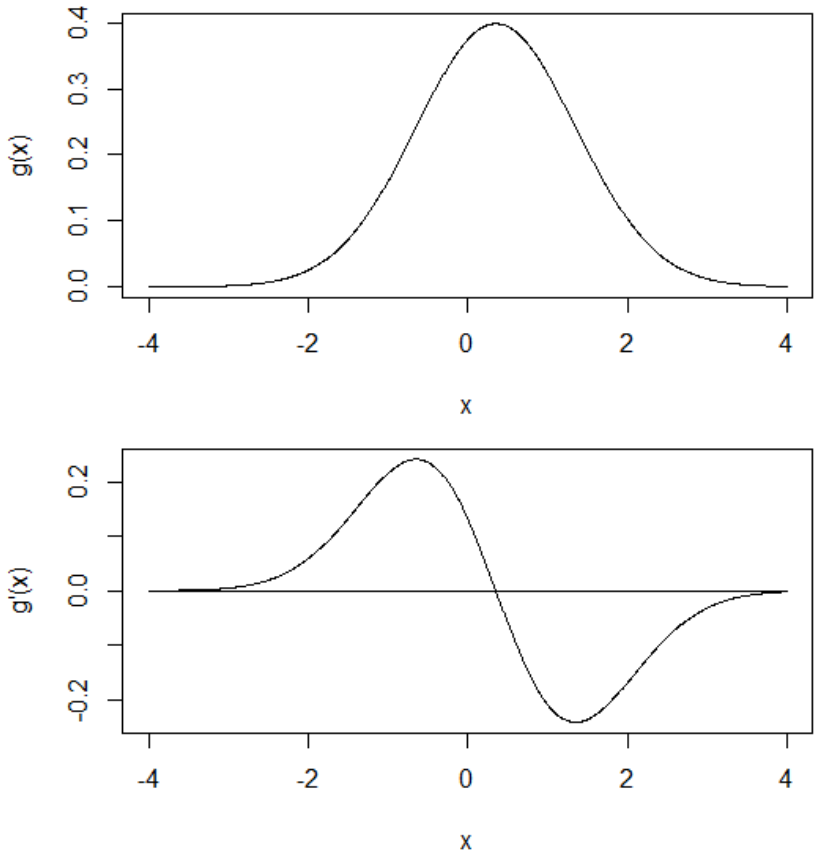
# Univariate optimisation

- $x$ real number, $g: \mathbb{R} \to \mathbb{R}$ continuously differentiable function
- We search $x^*$ with $g(x^*) = \max g(x)$

- Compute $g'(x)$ and search $x^*$ with $g'(x^*) = 0$
- One has then to check if the result is maximum, minimum, possibly local optimum…

# Univariate optimisation: bisection method

- Search $x^*$ with $g'(x^*) = 0$:
- See video on course homepage
- We iteratively improve approximations for $x^*$:
  $x^{(0)}$ -> $x^{(1)}$ -> $x^{(2)}$ -> …






Stockholm University

# Optimisation: convergence criterion

- Compare $x^{(t)}$ and $x^{(t+1)}$ and stop if they are "close enough"
- Absolute convergence criterion:

$$\left|x^{(t+1)} - x^{(t)}\right| < \epsilon$$

- Relative convergence criterion:

$$\frac{\left|x^{(t+1)} - x^{(t)}\right|}{|x^{(t)}|} < \epsilon$$

# Optimisation: Newton(-Raphson) method

- $x$ real number, $g: \mathbb{R} \to \mathbb{R}$ twice differentiable function
- Search $x^*$ with $g(x^*) = \max g(x)$ by searching $x^*$ with $g'(x^*) = 0$

- Taylor expansion around $x^*$ motivates:

$$0 = g'(x^*) \approx g'\left(x^{(t)}\right) + \left(x^* - x^{(t)}\right) g''\left(x^{(t)}\right)$$
$$-\left(x^* - x^{(t)}\right) g''\left(x^{(t)}\right) \approx g'\left(x^{(t)}\right)$$
$$x^* \approx x^{(t)} - g'\left(x^{(t)}\right) / g''(x^{(t)})$$

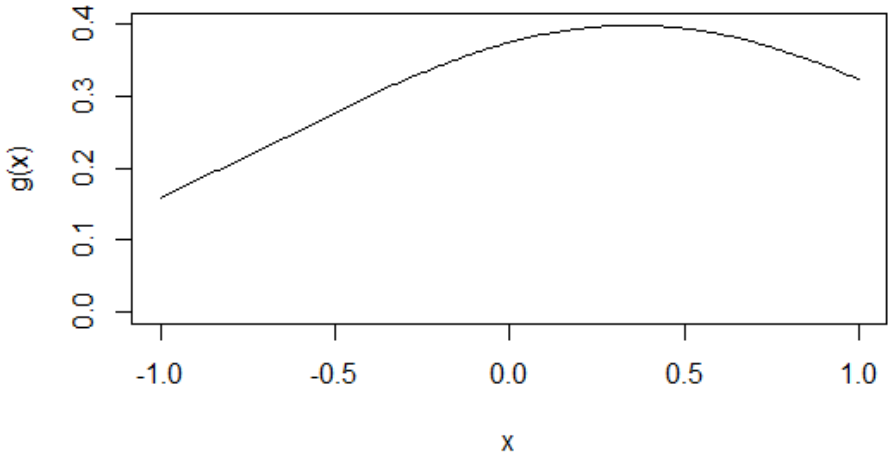
- Therefore, the Newton-iteration works as:

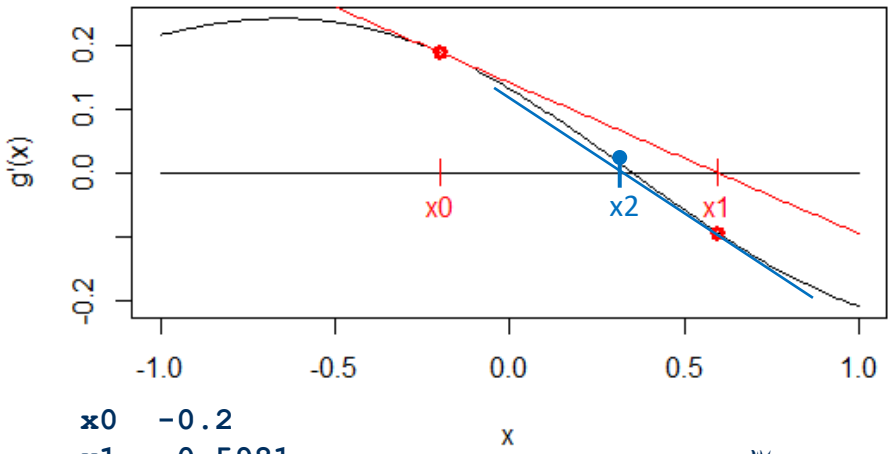
$$x^{(t+1)} = x^{(t)} - g'\left(x^{(t)}\right) / g''(x^{(t)})$$

# Optimisation: Newton(-Raphson) method

- $x^{(t+1)} = x^{(t)} - g'\left(x^{(t)}\right)/g''(x^{(t)})$
- Start with a $x^{(0)}$
- Tangent in $(x^{(0)}, g'(x^{(0)}))$ determines $x^{(1)}$
- Tangent in $(x^{(1)}, g'(x^{(1)}))$ determines $x^{(2)}$
- …
- until convergence crit. met

+ Newton method is fast
- Requires existence and computation of $g''$




```
x0  -0.2
x1   0.5981
x2   0.337996
x3   0.353557
x4   0.353553
x5   0.353553
STOP
```

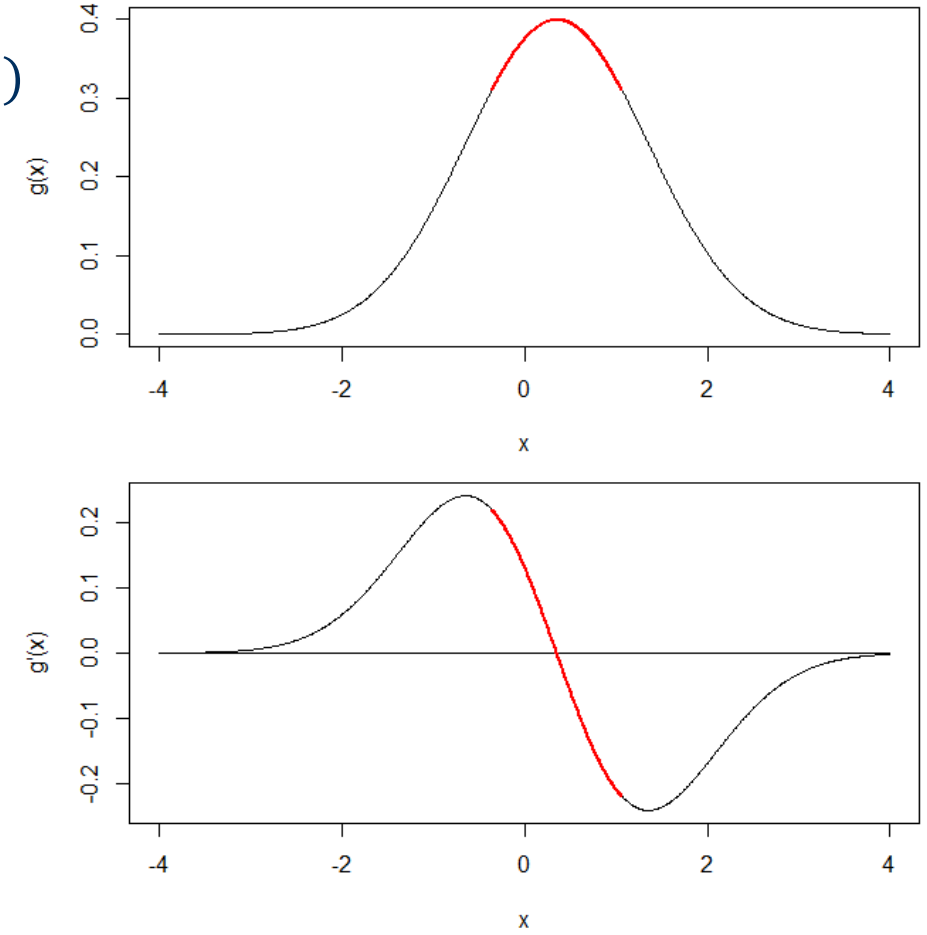


Stockholm University

# Optimisation: Newton(-Raphson) method

- $x^{(t+1)} = x^{(t)} - g'\left(x^{(t)}\right)/g''(x^{(t)})$

- What about the starting value $x^{(0)}$?






Stockholm University

# Optimisation: Secant method

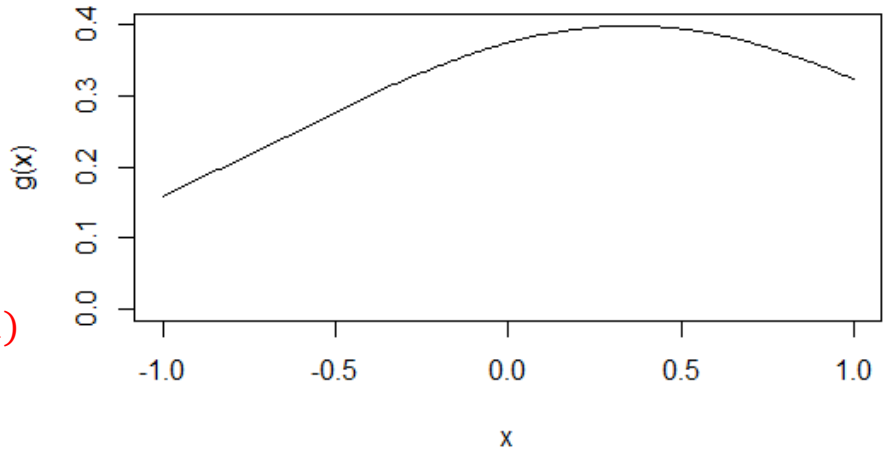
- $x$ real number, $g: \mathbb{R} \to \mathbb{R}$ once differentiable function
- Search $x^*$ with $g(x^*) = \max g(x)$ by searching $x^*$ with $g'(x^*) = 0$
- Recall: The Newton-iteration works as:

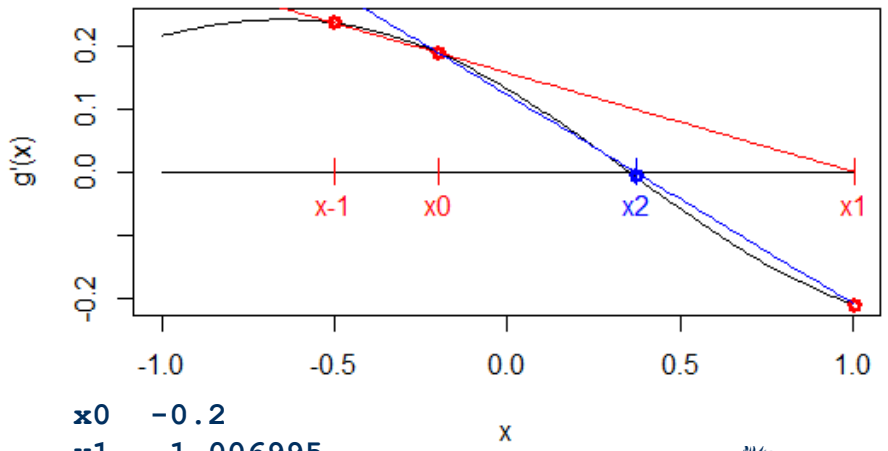
$$x^{(t+1)} = x^{(t)} - g'\left(x^{(t)}\right)/g''(x^{(t)})$$

- Need to compute $g''$ which might be difficult. Instead:
- Approximate $g''(x^{(t)})$ by $\left[g'\left(x^{(t)}\right) - g'(x^{(t-1)})\right]/\left(x^{(t)} - x^{(t-1)}\right)$

# Optimisation: Secant method

- $x^{(t+1)} = x^{(t)} - g'\left(x^{(t)}\right)\frac{x^{(t)}-x^{(t-1)}}{g'\left(x^{(t)}\right)-g'\left(x^{(t-1)}\right)}$
- Start with $x^{(0)}$ and $x^{(-1)}$
- Secant through $x^{(0)}$ and $x^{(-1)}$ determines $x^{(1)}$
- Secant through $x^{(1)}$ and $x^{(0)}$ determines $x^{(2)}$
- …
- until stopping crit. fulfilled




```
x0   -0.2
x1    1.006995
x2    0.371656
x3    0.349095
x4    0.353554
x5    0.353553
x6    0.353553
STOP
```

- Quite fast
- No 2nd derivative necessary



Stockholm University

# Optimisation: Convergence speed

- Convergence speed can be quantified by β and c as follows:
  - Let $\varepsilon^{(t)} = x^{(t)} - x^*$,
  - Find β and c such that $\lim_{t\to\infty} \varepsilon^{(t+1)}/(\varepsilon^{(t)})^{\beta} = c$
- ε = 1, 0.5, 0.25, 0.125, 0.063, 0.031, … ⇨ β=1, c=0.5,
- ε = 1, 0.1, 0.01, 0.001, 0.0001, … ⇨ β=1, c=0.1,
- If β=1, we say that convergence is ”linear”

- ε = 1, 0.5, 0.125, 0.008, 0.00003, … ⇨ β=2, c=0.5.
- If β=2, we say that convergence is ”quadratic”

# Optimisation: Comparison of methods

| Bisection | Secant | Newton |
|---|---|---|
| g’ required | g’ required | g’’ required |
| finds always an optimum between $a_0$ and $b_0$ (but could be local) | converges only when the two starting values ”close” to optimum | converges only when starting value ”close” to optimum |
| slow $\beta=1$ | quite fast $\beta = \frac{1+\sqrt{5}}{2} = 1.62$ | fast $\beta=2$ |

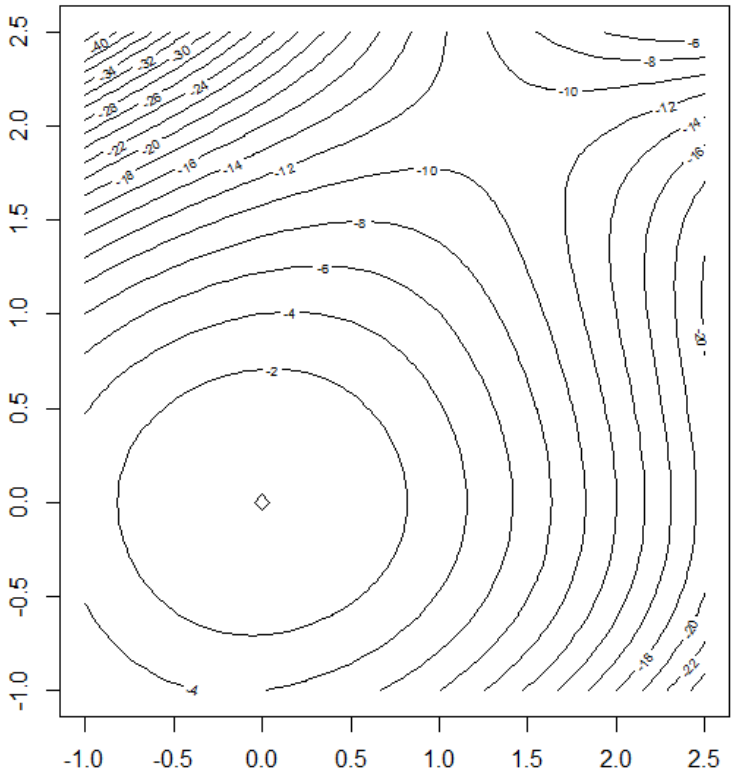
- $\lim_{t\to\infty} \varepsilon^{(t+1)}/(\varepsilon^{(t)})^{\beta} = c$

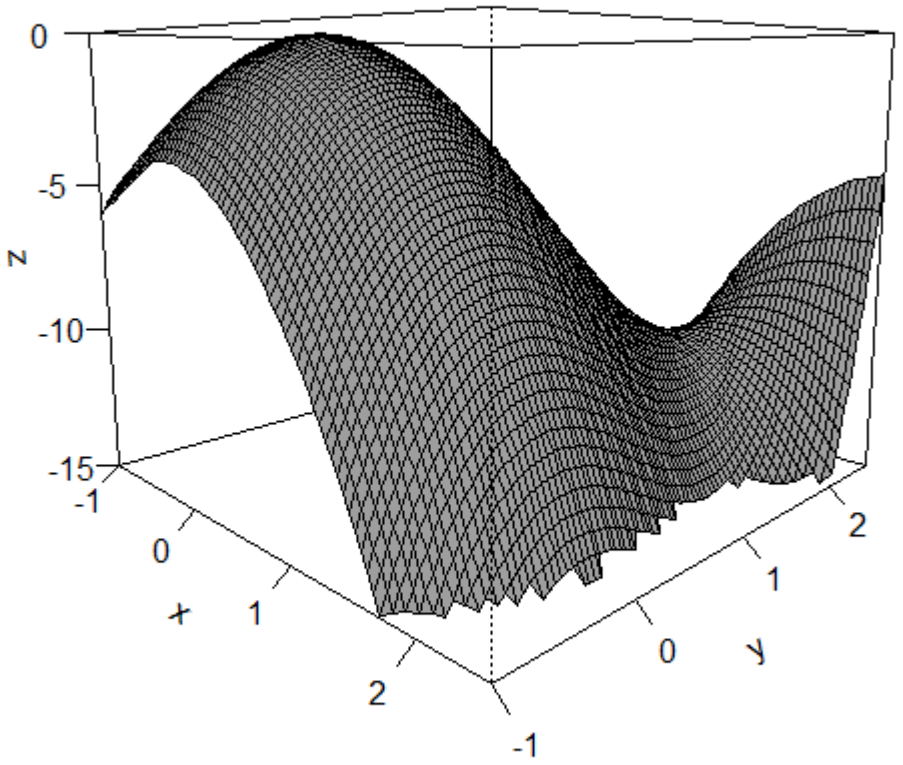
# Multivariate optimisation – gradient and Hessian

- $g\begin{pmatrix} x_1 \\ \vdots \\ x_p \end{pmatrix}$ is a real-valued function
- $g'\begin{pmatrix} x_1 \\ \vdots \\ x_p \end{pmatrix} = \begin{pmatrix} \frac{\partial g}{\partial x_1}(\boldsymbol{x}) \\ \vdots \\ \frac{\partial g}{\partial x_p}(\boldsymbol{x}) \end{pmatrix}$ is the gradient, $\boldsymbol{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_p \end{pmatrix}$
- $g''\begin{pmatrix} x_1 \\ \vdots \\ x_p \end{pmatrix} = \begin{pmatrix} \frac{\partial g}{\partial x_1 \partial x_1}(\boldsymbol{x}) & \cdots & \frac{\partial g}{\partial x_1 \partial x_p}(\boldsymbol{x}) \\ \vdots & & \vdots \\ \frac{\partial g}{\partial x_1 \partial x_p}(\boldsymbol{x}) & \cdots & \frac{\partial g}{\partial x_p \partial x_p}(\boldsymbol{x}) \end{pmatrix}$ is the Hessian matrix

# Bivariate optimisation – visualisation

- $g\begin{pmatrix} x \\ y \end{pmatrix} = -3x^2 - 4y^2 + xy^3$



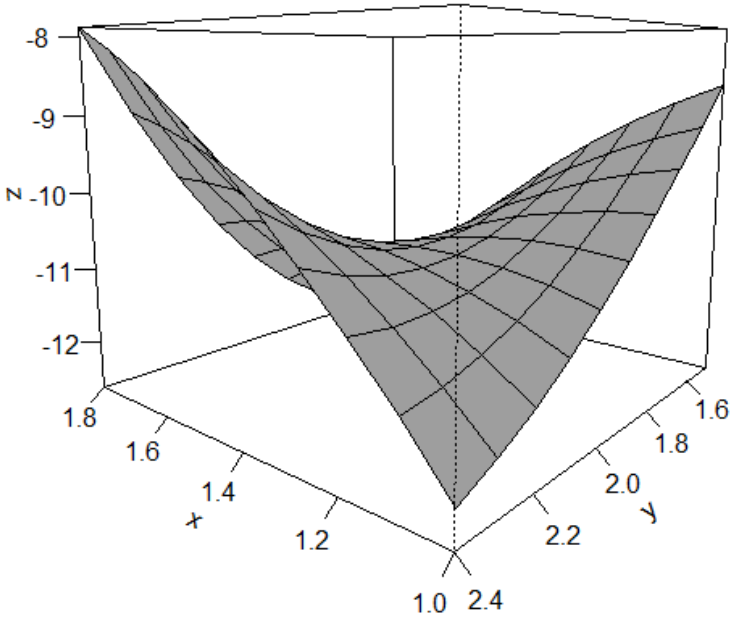




Figures can be drawn using R-core-functions **`contour`** and **`persp`**



Stockholm University

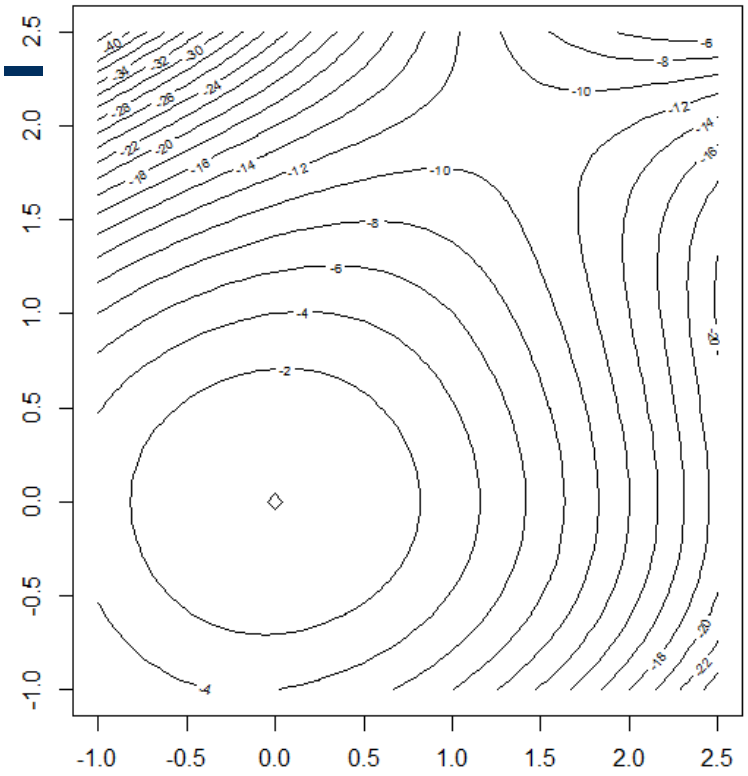
# Multivariate optimisation – saddle points






Stockholm University

# Multivariate optimisation – analytical optimisation




- $g\begin{pmatrix} x \\ y \end{pmatrix} = -3x^2 - 4y^2 + xy^3$
- $g'\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -6x + y^3 \\ -8y + 3xy^2 \end{pmatrix}$
- $g''\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -6 & 3y^2 \\ 3y^2 & -8 + 6xy \end{pmatrix}$

- See calculation in following document: AnalytOpt_OASI2020.pdf
- Maximum at (0,0), saddle point at (4/3,2)



Stockholm University

# Multivariate optimisation – Newton meth.

- $\boldsymbol{x}$ $p-$dimensional vector, $g$: $\mathbb{R}^p \rightarrow \mathbb{R}$ function
- We search $\boldsymbol{x}^*$ with $g(\boldsymbol{x}^*) = \max g(\boldsymbol{x})$
- Now, $\boldsymbol{g}'$ is $p$-dim. vector and $\boldsymbol{g}''$ is $p$x$p$-matrix ("Hessian")

- The multivariate version of the Newton method is motivated by the multivariate Taylor expansion

$$0 = \boldsymbol{g}'(\boldsymbol{x}^*) \approx \boldsymbol{g}'\left(\boldsymbol{x}^{(t)}\right) + \boldsymbol{g}''\left(\boldsymbol{x}^{(t)}\right)\left(\boldsymbol{x}^* - \boldsymbol{x}^{(t)}\right)$$

- The Newton-iteration works as:

$$\boldsymbol{x}^{(t+1)} = \boldsymbol{x}^{(t)} - \left(\boldsymbol{g}''\left(\boldsymbol{x}^{(t)}\right)\right)^{-1} \boldsymbol{g}'\left(\boldsymbol{x}^{(t)}\right)$$

# Multivariate optimisation – Newton meth.

- $\boldsymbol{x}^{(t+1)} = \boldsymbol{x}^{(t)} - \left(\boldsymbol{g}''\left(\boldsymbol{x}^{(t)}\right)\right)^{-1} \boldsymbol{g}'\left(\boldsymbol{x}^{(t)}\right)$
- Example:

  Let $g_1$ be the density of $N\left(\begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}0.6 & 0\\0 & 0.6\end{pmatrix}\right)$,

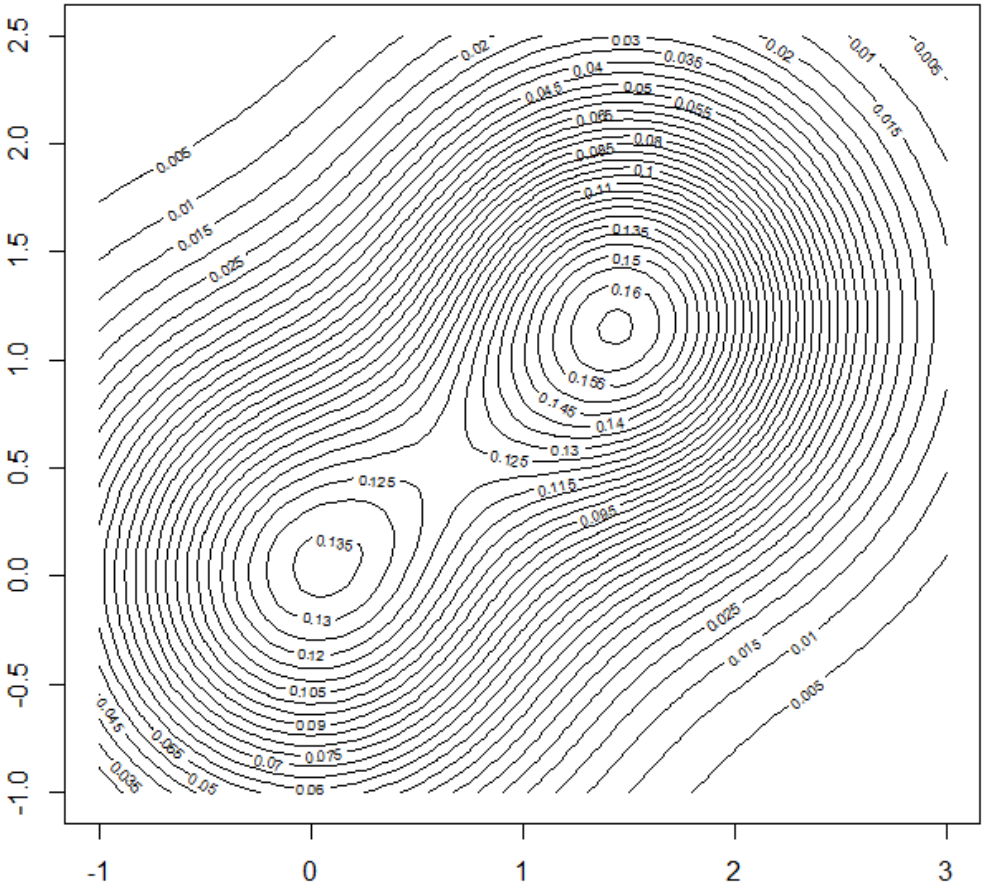
  $g_2$ be density of $N\left(\begin{pmatrix}1.5\\1.2\end{pmatrix}, \begin{pmatrix}0.5 & 0\\0 & 0.5\end{pmatrix}\right)$, and $g=(g_1+g_2)/2$, i.e.

  $$g(x_1, x_2) = \frac{1}{4\pi}\left(\frac{1}{0.6} e^{-\left({x_1}^2+{x_2}^2\right)/1.2} + \frac{1}{0.5} e^{-\left((x_1-1.5)^2+(x_2-1.2)^2\right)}\right)$$

  (g is density of a normal mixture distribution).
- Compute point $\mathbf{x}=(x_1,x_2)$ where density $g(\mathbf{x})$ maximal.
- Do you have a guess?

# Multivariate optimisation – Newton meth.

- $g(x_1, x_2) = \frac{1}{4\pi}\left(\frac{1}{0.6}e^{-({x_1}^2+{x_2}^2)/1.2} + \frac{1}{0.5}e^{-\left((x_1-1.5)^2+(x_2-1.2)^2\right)}\right)$






Stockholm University

# Multivariate optimisation – Newton meth.

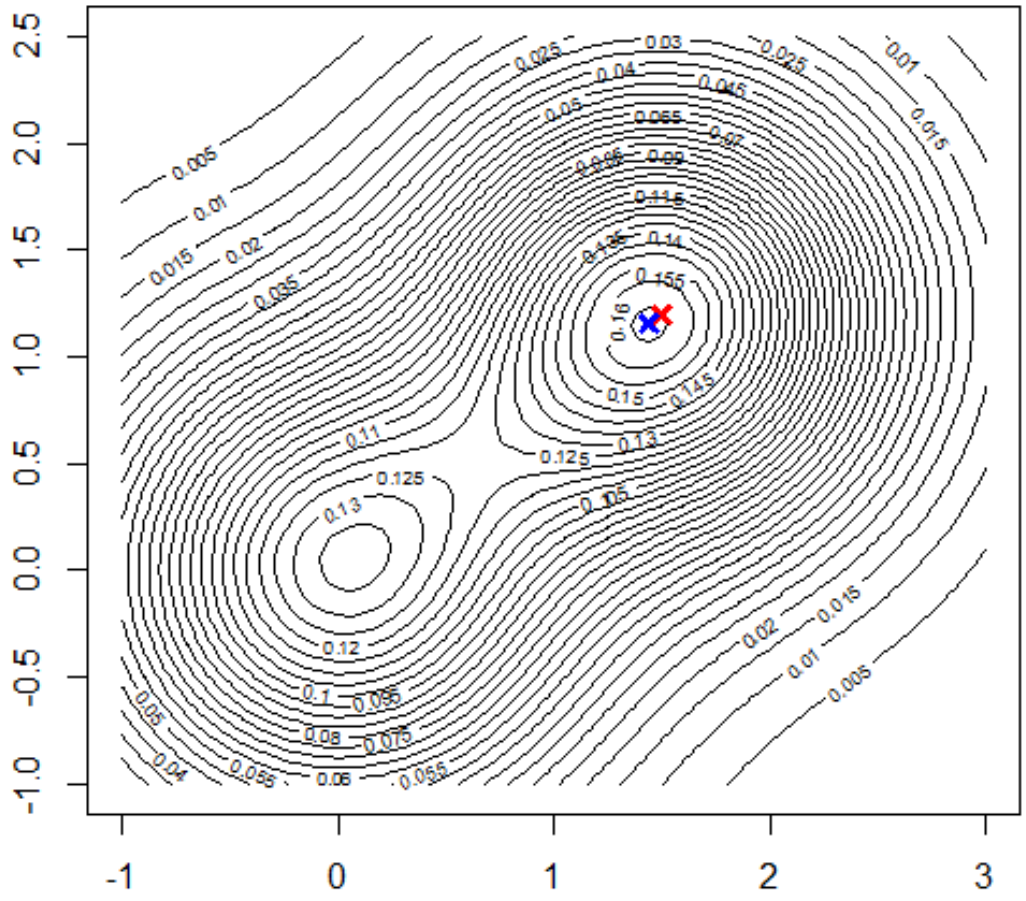
- $\boldsymbol{x}^{(t+1)} = \boldsymbol{x}^{(t)} - \left(\boldsymbol{g}''\left(\boldsymbol{x}^{(t)}\right)\right)^{-1} \boldsymbol{g}'\left(\boldsymbol{x}^{(t)}\right)$
- We need $\boldsymbol{g'}$ and $\boldsymbol{g''}$ of

$$g(x_1, x_2) = \frac{1}{4\pi}\left(\frac{1}{0.6} e^{-\left({x_1}^2+{x_2}^2\right)/(2\cdot 0.6)} + \frac{1}{0.5} e^{-\left((x_1-1.5)^2+(x_2-1.2)^2\right)}\right)$$

- $\frac{\partial g}{\partial x_1}(x_1, x_2) = \frac{1}{4\pi}\left(\frac{-2x_1}{1.2\cdot 0.6} e^{-\left({x_1}^2+{x_2}^2\right)/1.2} + \frac{-2(x_1-1.5)}{0.5} e^{-\left((x_1-1.5)^2+(x_2-1.2)^2\right)}\right)$
- $\frac{\partial g}{\partial x_2}(x_1, x_2) = \frac{1}{4\pi}\left(\frac{-2x_2}{1.2\cdot 0.6} e^{-\left({x_1}^2+{x_2}^2\right)/1.2} + \frac{-2(x_2-1.2)}{0.5} e^{-\left((x_1-1.5)^2+(x_2-1.2)^2\right)}\right)$
- $\boldsymbol{g}'(x_1, x_2) = \begin{pmatrix} \frac{\partial g}{\partial x_1}(x_1, x_2) \\ \frac{\partial g}{\partial x_2}(x_1, x_2) \end{pmatrix}$
- $\frac{\partial^2 g}{\partial^2 x_1}(x_1, x_2)$=…; $\frac{\partial^2 g}{\partial x_1 \partial x_2}(x_1, x_2)$=…; $\frac{\partial^2 g}{\partial^2 x_2}(x_1, x_2)$=… give $\boldsymbol{g''}$

# Multivariate optimisation – Newton meth.

- $g(x_1, x_2) = \frac{1}{4\pi}\left(\frac{1}{0.6}e^{-({x_1}^2+{x_2}^2)/1.2} + \frac{1}{0.5}e^{-\left((x_1-1.5)^2+(x_2-1.2)^2\right)}\right)$




- Start with $\boldsymbol{x^{(0)}} = \begin{pmatrix} \mathbf{1.5} \\ \mathbf{1.2} \end{pmatrix}$
- $\boldsymbol{g'(x^{(0)})} = \begin{pmatrix} -0.0153 \\ -0.0123 \end{pmatrix}$
- $\boldsymbol{g''(x^{(0)})} = \begin{pmatrix} -0.2902 & 0.0306 \\ 0.0306 & -0.3040 \end{pmatrix}$
- $\boldsymbol{\left(g''(x^{(0)})\right)^{-1} g'(x^{(0)})} = \begin{pmatrix} \mathbf{0.058} \\ \mathbf{0.046} \end{pmatrix}$
- $\boldsymbol{x^{(1)}} = \begin{pmatrix} \mathbf{1.5} \\ \mathbf{1.2} \end{pmatrix} - \begin{pmatrix} \mathbf{0.058} \\ \mathbf{0.046} \end{pmatrix} = \begin{pmatrix} \mathbf{1.442} \\ \mathbf{1.154} \end{pmatrix}$

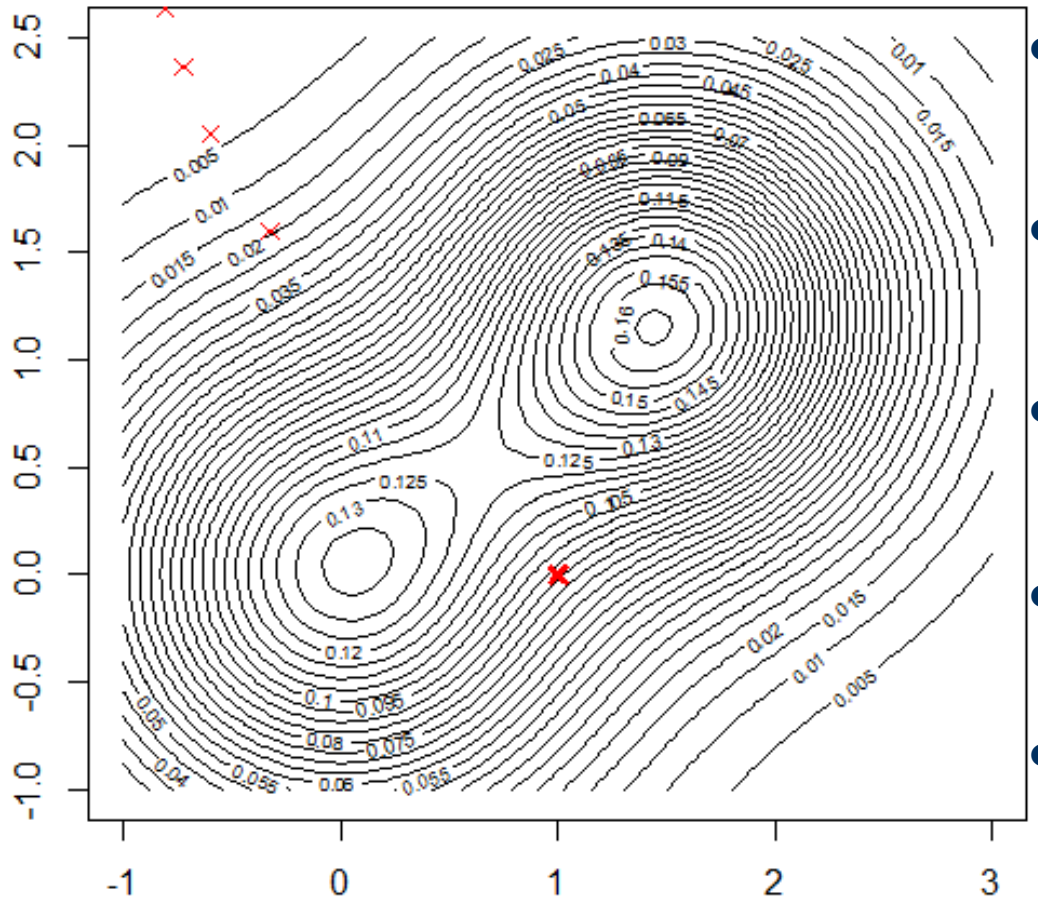
$$\boldsymbol{x^{(2)} = x^*} = \begin{pmatrix} \mathbf{1.441} \\ \mathbf{1.153} \end{pmatrix}$$



Stockholm University

# Multivariate optimisation – Newton meth.

- $g(x_1, x_2) = \frac{1}{4\pi}\left(\frac{1}{0.6}e^{-({x_1}^2+{x_2}^2)/1.2} + \frac{1}{0.5}e^{-\left((x_1-1.5)^2+(x_2-1.2)^2\right)}\right)$




- Start with $\boldsymbol{x^{(0)}} = \begin{pmatrix}\mathbf{1}\\ \mathbf{0}\end{pmatrix}$
- $\boldsymbol{g'(x^{(0)})} = \begin{pmatrix}-0.0667\\ +0.0705\end{pmatrix}$
- $\boldsymbol{g''(x^{(0)})} = \begin{pmatrix}0.0347 & 0.0705\\ 0.0705 & 0.0144\end{pmatrix}$
- $\left(\boldsymbol{g''(x^{(0)})}\right)^{-1}\boldsymbol{g'(x^{(0)})} = \begin{pmatrix}\mathbf{1.33}\\ \mathbf{-1.60}\end{pmatrix}$
- $\boldsymbol{x^{(1)}} = \begin{pmatrix}\mathbf{1}\\ \mathbf{0}\end{pmatrix} - \begin{pmatrix}\mathbf{1.33}\\ \mathbf{-1.6}\end{pmatrix}$

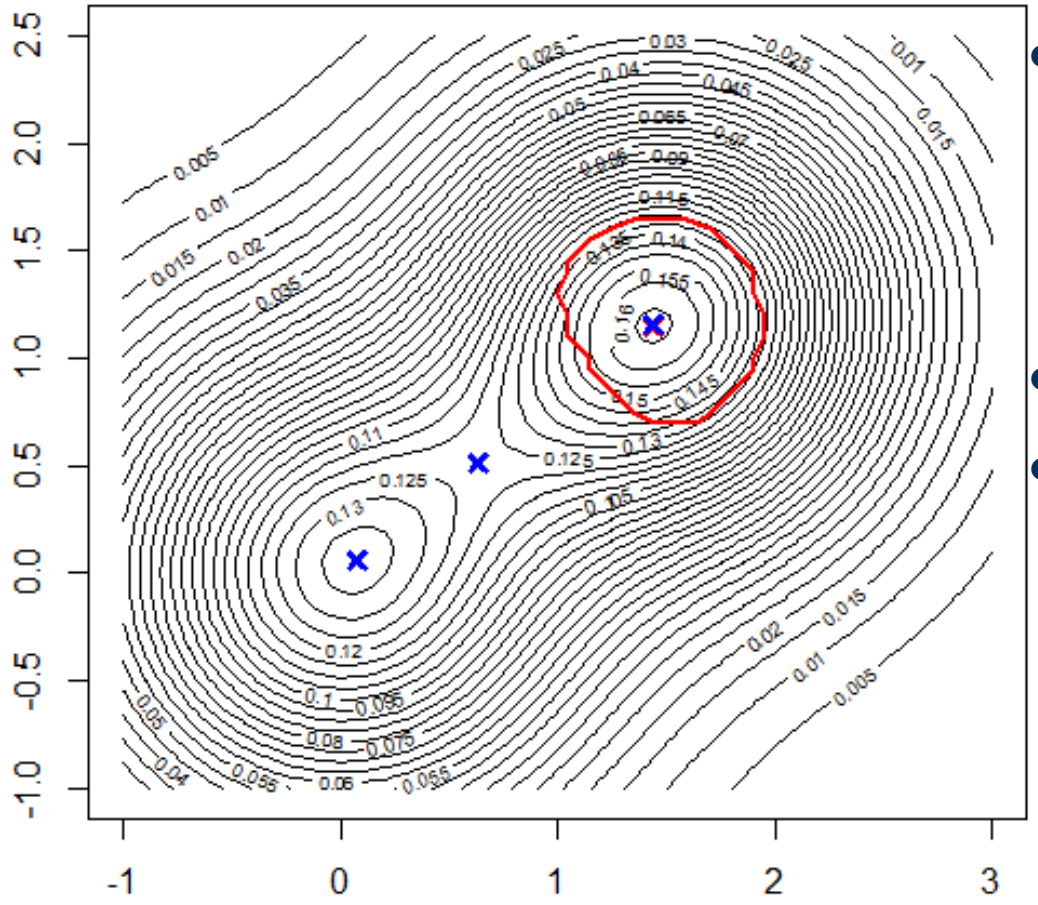
$= \begin{pmatrix}\mathbf{-0.33}\\ \mathbf{1.6}\end{pmatrix}$



Stockholm University

# Multivariate optimisation – Newton meth.

- $g(x_1, x_2) = \frac{1}{4\pi}\left(\frac{1}{0.6}e^{-({x_1}^2+{x_2}^2)/1.2} + \frac{1}{0.5}e^{-\left((x_1-1.5)^2+(x_2-1.2)^2\right)}\right)$




- Only starting values within the red-marked area converge to the right global maximum
- Convergence very quick
- Other starting values converge to the local maximum or saddle point (both blue-marked) or diverge while searching for a minimum



Stockholm University

# Multivariate optimisation – Steepest ascent method

- When using Newton method, it is not guaranteed that g(x) increases in each step
- To compute the Hessian $\boldsymbol{g''}$ can be difficult
- A method forcing improvements in each step is the steepest ascent method

$$\boldsymbol{x}^{(t+1)} = \boldsymbol{x}^{(t)} - \left(\boldsymbol{g''}\left(\boldsymbol{x}^{(t)}\right)\right)^{-1} \boldsymbol{g'}\left(\boldsymbol{x}^{(t)}\right)$$

$$\boldsymbol{x}^{(t+1)} = \boldsymbol{x}^{(t)} + \alpha^{(t)} \boldsymbol{I}\, \boldsymbol{g'}\left(\boldsymbol{x}^{(t)}\right)$$

- Other choices instead $\boldsymbol{I}$ in formula above possible
- We know that $g$ will increase for small $\alpha$

# Multivariate optimisation – Steepest ascent method

$$\boldsymbol{x}^{(\boldsymbol{t+1})} = \boldsymbol{x}^{(\boldsymbol{t})} + \alpha^{(t)} \boldsymbol{I}\, \boldsymbol{g}'\left(\boldsymbol{x}^{(\boldsymbol{t})}\right)$$

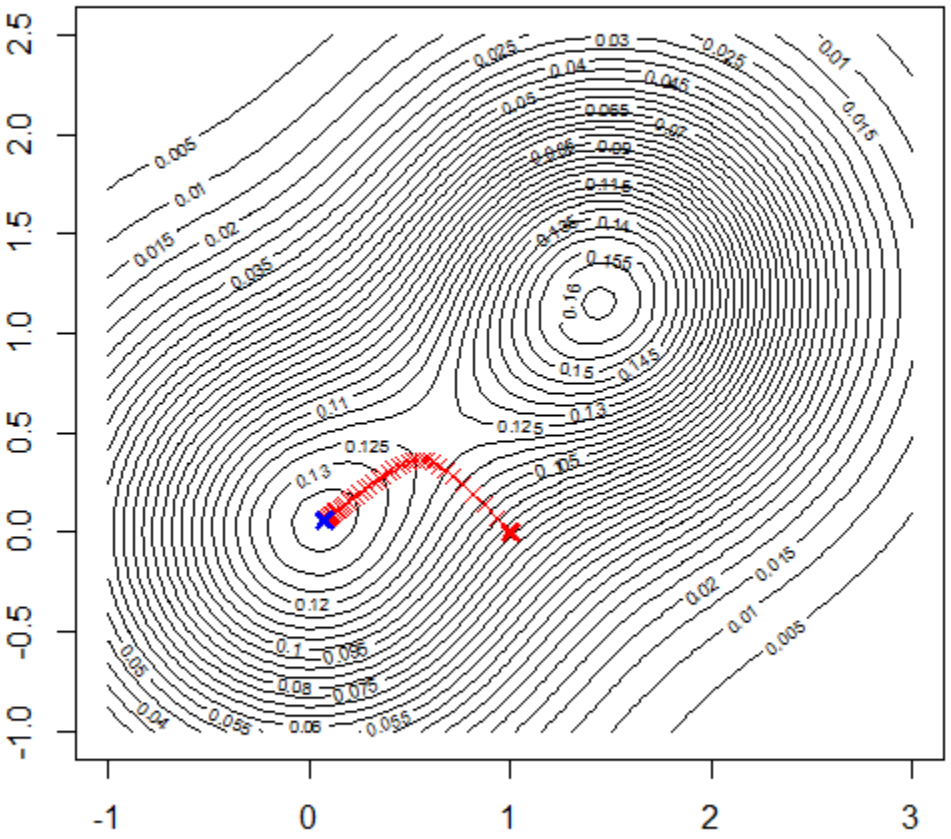
- Other choices instead $\boldsymbol{I}$ in formula above possible
- We know that *g* will increase for small $\alpha$

- Try $\alpha^{(t)} = 1$ first
- If *g* decreases, half $\alpha^{(t)}$ until $\mathrm{g}(\boldsymbol{x}^{(\boldsymbol{t+1})})$ increases

- More sophisticated is to search $\alpha$ such that *g* becomes maximal (”line search method”)

# Multivariate optimisation – Steepest ascent method

- $g(x_1, x_2) = \frac{1}{4\pi}\left(\frac{1}{0.6}e^{-({x_1}^2+{x_2}^2)/1.2} + \frac{1}{0.5}e^{-\left((x_1-1.5)^2+(x_2-1.2)^2\right)}\right)$




- Start with $\boldsymbol{x}^{(0)} = \begin{pmatrix}\mathbf{1}\\ \mathbf{0}\end{pmatrix}$
- $\boldsymbol{g}'(\boldsymbol{x}^{(0)}) = \begin{pmatrix}-0.0667\\ +0.0705\end{pmatrix}$
- $\boldsymbol{x}^{(1)} = \begin{pmatrix}\mathbf{1}\\ \mathbf{0}\end{pmatrix} + \alpha^{(0)}\begin{pmatrix}-0.0667\\ +0.0705\end{pmatrix}$

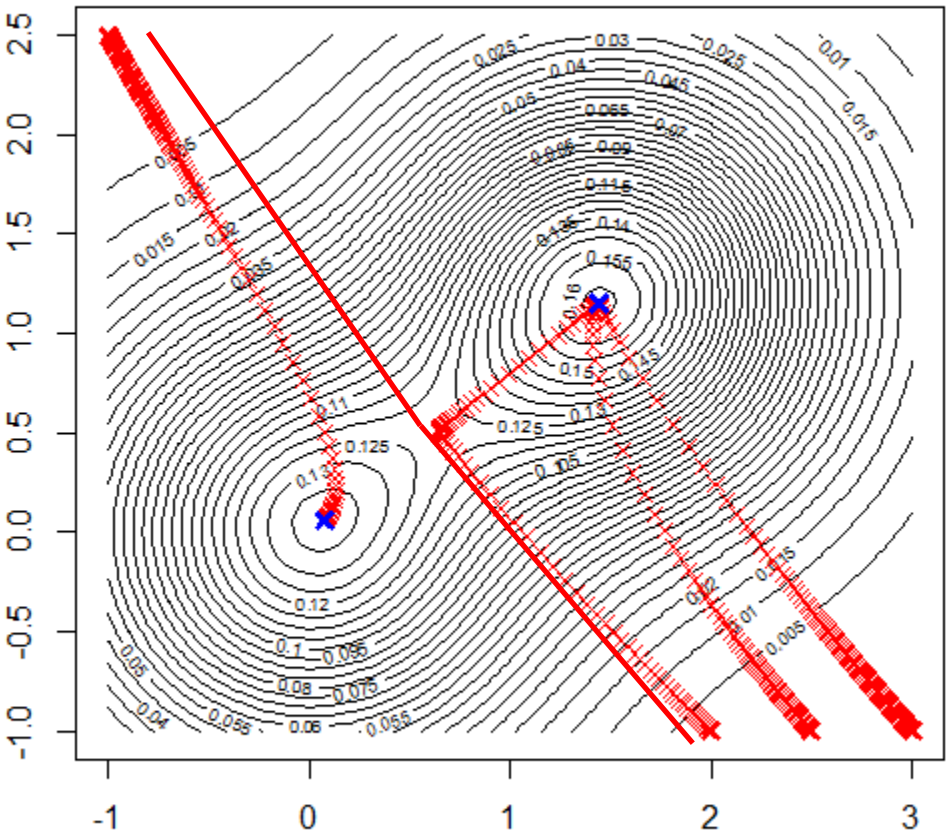
  $= \begin{pmatrix}\mathbf{0.9333}\\ \mathbf{0.0705}\end{pmatrix}$



Stockholm University

# Multivariate optimisation – Steepest ascent method

- $g(x_1, x_2) = \frac{1}{4\pi}\left(\frac{1}{0.6}e^{-({x_1}^2+{x_2}^2)/1.2} + \frac{1}{0.5}e^{-\left((x_1-1.5)^2+(x_2-1.2)^2\right)}\right)$




- Start with $\boldsymbol{x}^{(0)} = \begin{pmatrix}-1\\2.5\end{pmatrix}, \begin{pmatrix}2\\-1\end{pmatrix}, \begin{pmatrix}2.5\\-1\end{pmatrix}, \begin{pmatrix}3\\-1\end{pmatrix}$
- All these paths converge to either the global or local maximum
- Convergence is much slower than for Newton
- Depending on convergence criterion and alpha-rule, convergence not always guaranteed



Stockholm University

# Stopping criteria

- Stopping criterion e.g. $\left(\boldsymbol{x}^{(t+1)} - \boldsymbol{x}^{(t)}\right)^T \left(\boldsymbol{x}^{(t+1)} - \boldsymbol{x}^{(t)}\right) < \epsilon$
- Other stopping criteria:
  - Absolut stopping criterion, $\left\|\boldsymbol{x}^{(t+1)} - \boldsymbol{x}^{(t)}\right\| < \epsilon$,
  - Relative stopping criterion, $\left\|\boldsymbol{x}^{(t+1)} - \boldsymbol{x}^{(t)}\right\| / \left\|\boldsymbol{x}^{(t+1)}\right\| < \epsilon$,
  - Modified rel. stopping crit., $\frac{\left\|\boldsymbol{x}^{(t+1)} - \boldsymbol{x}^{(t)}\right\|}{\left\|\boldsymbol{x}^{(t+1)}\right\| + \varepsilon} < \varepsilon$
  - Different norms $\|\cdot\|$ can be used

- Recall: Norms for $\boldsymbol{x} = (x_1, \ldots, x_p)^T$: $\|\boldsymbol{x}\| = \|\boldsymbol{x}\|_2 = \sqrt{x_1^2 + \cdots + x_p^2}$ (Euklid), $\|\boldsymbol{x}\|_1 = |x_1| + \cdots + |x_p|$, $\|\boldsymbol{x}\|_\infty = \max\{|x_1|, \ldots, |x_p|\}$ (max-norm)

# Convergence rate of optimisation algorithms

- Recall the definition for convergence speed/rate (e.g. bisection is approximately linear, Newton is quadratic)
- To check convergence speed in an optimisation-run, you can calculate

$$D^{(t)} = \frac{\left\| \boldsymbol{x}^{(t)} - \boldsymbol{x}^* \right\|}{\left\| \boldsymbol{x}^{(t-1)} - \boldsymbol{x}^* \right\|}$$

  (after the algorithm has converged for the result $\boldsymbol{x}^*$), see GH, page 101/102

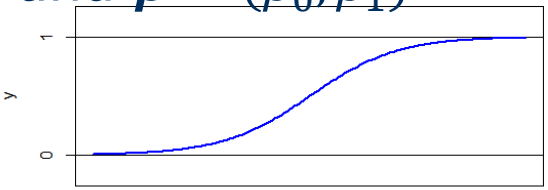
- If $D^{(t)} \to 1$, there is not even linear convergence (bad),
- If $D^{(t)} \to c \in (0,1)$, linear convergence,
- If $D^{(t)} \to 0$, better than linear convergence.

# Maximum likelihood estimator (MLE)

- The MLE is solution of $g(\widehat{\boldsymbol{\beta}}) = \max g(\boldsymbol{b})$ with $g(\widehat{\boldsymbol{\beta}}) = \text{log}-\text{likelihood}(\widehat{\boldsymbol{\beta}}, \boldsymbol{X}, \boldsymbol{y}) = \sum_{i=1}^{n} \text{log}-\text{likelihood}(\widehat{\boldsymbol{\beta}}, \boldsymbol{x_i}, y_i)$ (the latter equation requires independence of observations)
- In the simple case of normally distributed observations, MLE=LSE and we have an algebraic solution
- Otherwise, we need usually iterative methods to compute the MLE

# MLE for logistic regression

- The MLE is solution of $g(\widehat{\boldsymbol{\beta}}) = \max g(\boldsymbol{b})$ with $g(\widehat{\boldsymbol{\beta}}) = \text{log}-\text{likelihood}(\widehat{\boldsymbol{\beta}}, \boldsymbol{X}, \boldsymbol{y}) = \sum_{i=1}^{n} \text{log}-\text{likelihood}(\widehat{\boldsymbol{\beta}}, \boldsymbol{x_i}, y_i)$
- Logistic regression model: $$p(\boldsymbol{x_i}) = P(Y = 1|\boldsymbol{x_i}) = \frac{1}{1 + \exp(-\boldsymbol{x_i^T\beta})} = \frac{\exp(\boldsymbol{x_i^T\beta})}{1 + \exp(\boldsymbol{x_i^T\beta})}$$
- For simple logistic regression: $\boldsymbol{x_i^T} = (1, x_{i1})$ and $\boldsymbol{\beta} = (\beta_0, \beta_1)^{\boldsymbol{T}}$
- For multiple logistic regression ($p$ var.): $\boldsymbol{x_i^T} = (1, x_{i1}, \dots, x_{ip})$ and $\boldsymbol{\beta} = (\beta_0, \beta_1, \dots, \beta_p)^{\boldsymbol{T}}$




- Log likelihood: $g(\boldsymbol{b}) = \sum_{i=1}^{n} \log(\hat{y}_i) y_i + \log(1 - \hat{y}_i)(1 - y_i)$ where $\hat{y}_i = 1/(1 + \exp(-\boldsymbol{x_i^T b}))$
- We can write each term in the sum alternatively as follows: $\log(1 - \hat{y}_i) + (\log(\hat{y}_i) - \log(1 - \hat{y}_i)) y_i = -\log(1 + \exp(\boldsymbol{x_i^T b})) + \boldsymbol{x_i^T b} y_i$
- Derivatives with respect to $b_j, j = 0, \dots, p$, necessary

# Assignments

- Topic 1: October 2 until October 12
- Topic 2: October 13 until October 22
- Topic 3: October 23 until November 5
- Topic 4: November 6 until November 27
- Final assignment and second chance for Topic 1-4: November 13 until **January 25 (no extension!)**

- Some Problems (e.g. 1.3) will be discussed in upcoming lectures and developed further. If you do not solve them until the first deadline, I will provide a different problem as replacement to be done until January 25