Computational statistics, lecture 2

Frank Miller, Department of Computer and Information Science, Linköping University

frank.miller@liu.se

January 28, 2025

Today’s schedule

- Multivariate Optimization
  - Analytical opt.
  - Newton
  - Steepest ascent
  - Quasi-Newton
  - Nelder-Mead

Multivariate optimization – gradient and Hessian

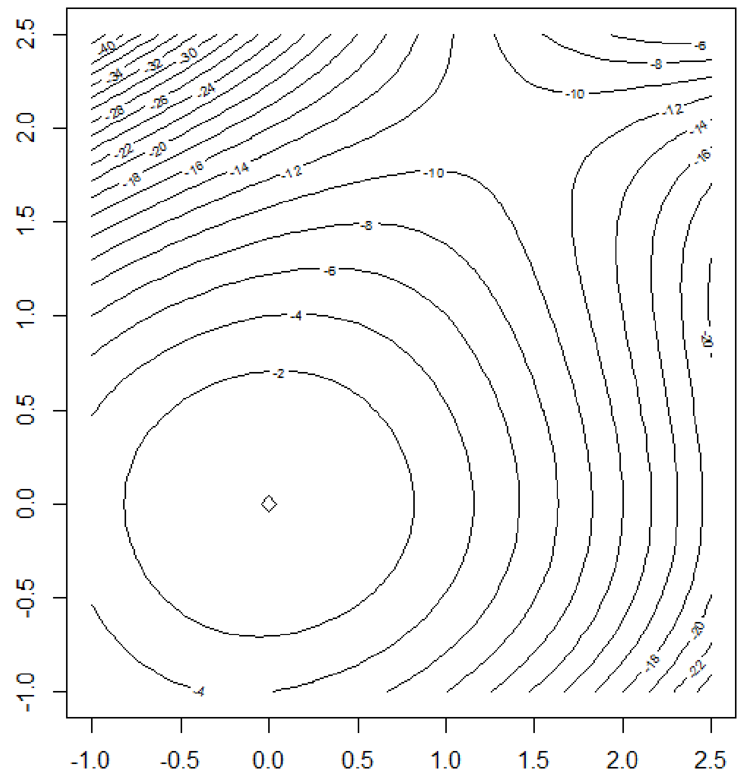
- $\boldsymbol{g}\begin{pmatrix} x_1 \\ \vdots \\ x_p \end{pmatrix}$ is a real-valued function

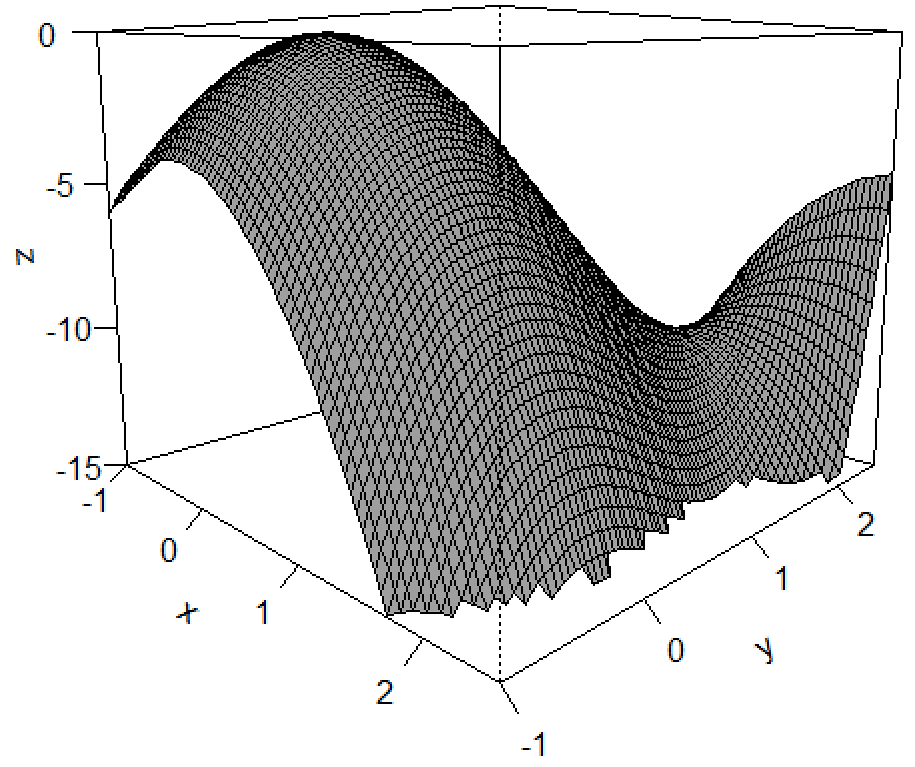
- $\boldsymbol{g}'\begin{pmatrix} x_1 \\ \vdots \\ x_p \end{pmatrix} = \begin{pmatrix} \frac{\partial g}{\partial x_1}(\boldsymbol{x}) \\ \vdots \\ \frac{\partial g}{\partial x_p}(\boldsymbol{x}) \end{pmatrix}$ is the gradient, $\boldsymbol{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_p \end{pmatrix}$

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

Bivariate optimization – visualization

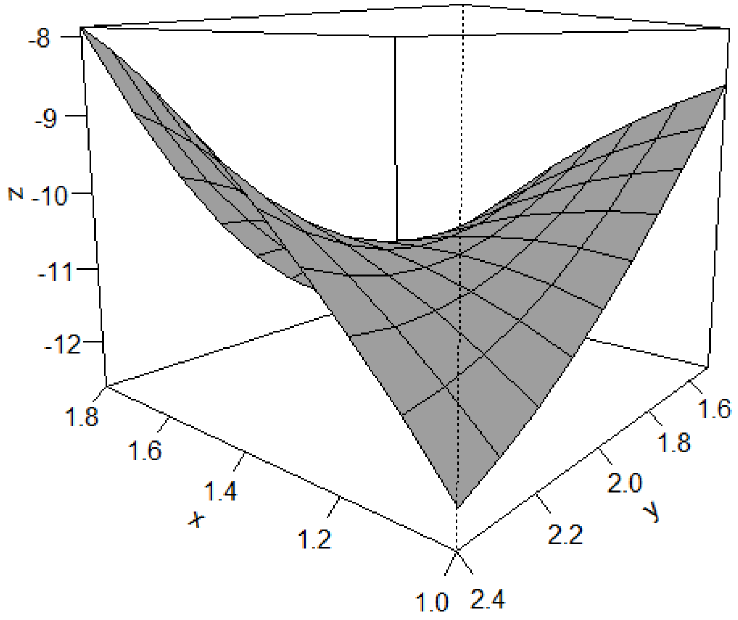
- $\boldsymbol{g}\binom{x}{y} = -3x^2 - 4y^2 + xy^3$





Figures can be drawn with **R**-core-functions **`contour`** and **`persp`** (see also **R**-code connected to lecture LM1 on homepage)

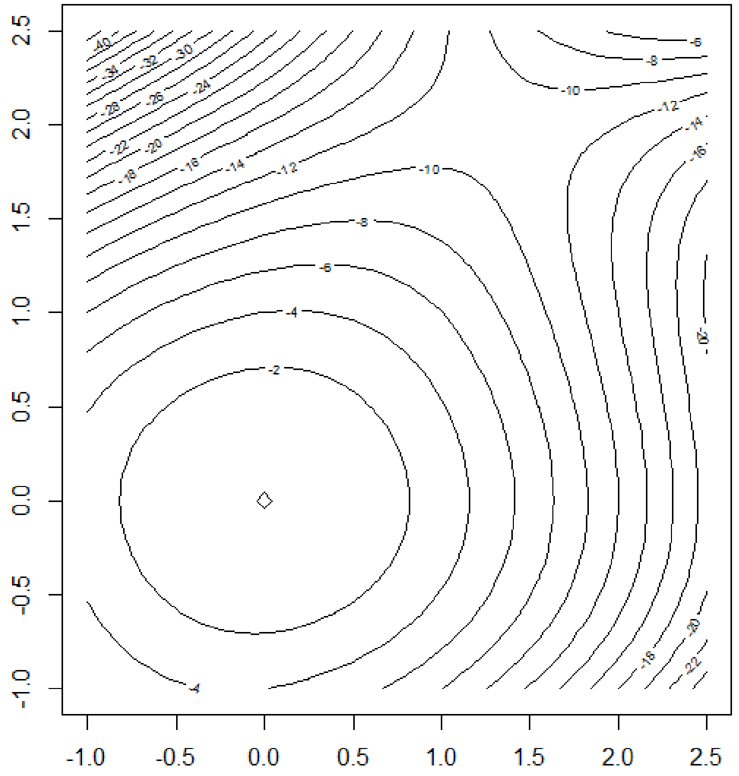
Multivariate optimization – saddle points



Multivariate optimization – analytical optimization

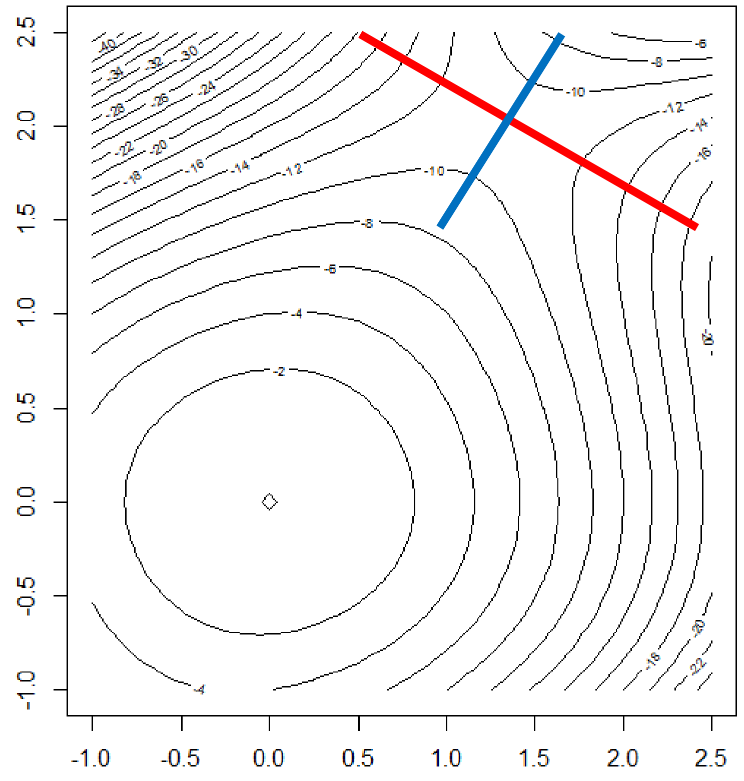
- $\boldsymbol{g}\begin{pmatrix}x\\y\end{pmatrix} = -3x^2 - 4y^2 + xy^3$
- $\boldsymbol{g}'\begin{pmatrix}x\\y\end{pmatrix} = \begin{pmatrix}-6x + y^3\\-8y + 3xy^2\end{pmatrix}$
- $\boldsymbol{g}''\begin{pmatrix}x\\y\end{pmatrix} = \begin{pmatrix}-6 & 3y^2\\3y^2 & -8 + 6xy\end{pmatrix}$

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



Saddle point and eigenvectors of the Hessian

- $\boldsymbol{g}\begin{pmatrix}x\\y\end{pmatrix} = -3x^2 - 4y^2 + xy^3$
- Saddle point at (4/3, 2)
- $\boldsymbol{g}'\begin{pmatrix}4/3\\2\end{pmatrix} = \begin{pmatrix}0\\0\end{pmatrix}$
- $\boldsymbol{g}''\begin{pmatrix}4/3\\2\end{pmatrix} = \begin{pmatrix}-6 & 12\\12 & 8\end{pmatrix}$



- Eigenvalues 14.89, -12.89; eigenvectors $\begin{pmatrix}0.498\\0.867\end{pmatrix}$, $\begin{pmatrix}-0.867\\0.498\end{pmatrix}$

Multivariate Newton

- $\boldsymbol{x}$ p–dimensional vector, g: $\mathbb{R}^p \to \mathbb{R}$ function
- We search $\boldsymbol{x}^*$ with $g(\boldsymbol{x}^*) = \max g(\boldsymbol{x})$
- Now, $\boldsymbol{g}'$ is p-dim. vector and $\boldsymbol{g}''$ is $p \times p$-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 Newton

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

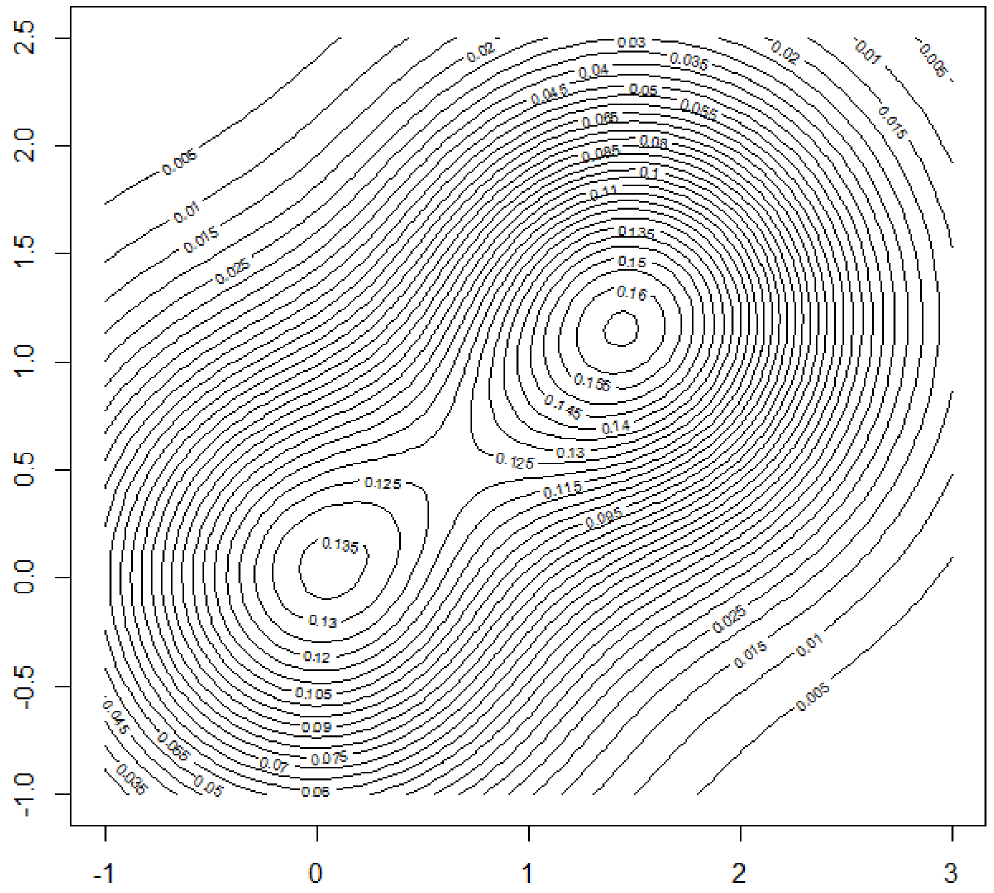
  Let g_1 be the density of $N\left(\begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}0.6 & 0\\0 & 0.6\end{pmatrix}\right)$, g_2 be density of $N\left(\begin{pmatrix}1.5\\1.2\end{pmatrix}, \begin{pmatrix}0.5 & 0\\0 & 0.5\end{pmatrix}\right)$, and $g = \frac{g_1+g_2}{2}$, i.e.

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

  (g is density of a normal mixture distribution).
- Compute point $\boldsymbol{x} = (x_1, x_2)$ with maximal density $g(\boldsymbol{x})$.

Multivariate Newton

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



Multivariate Newton

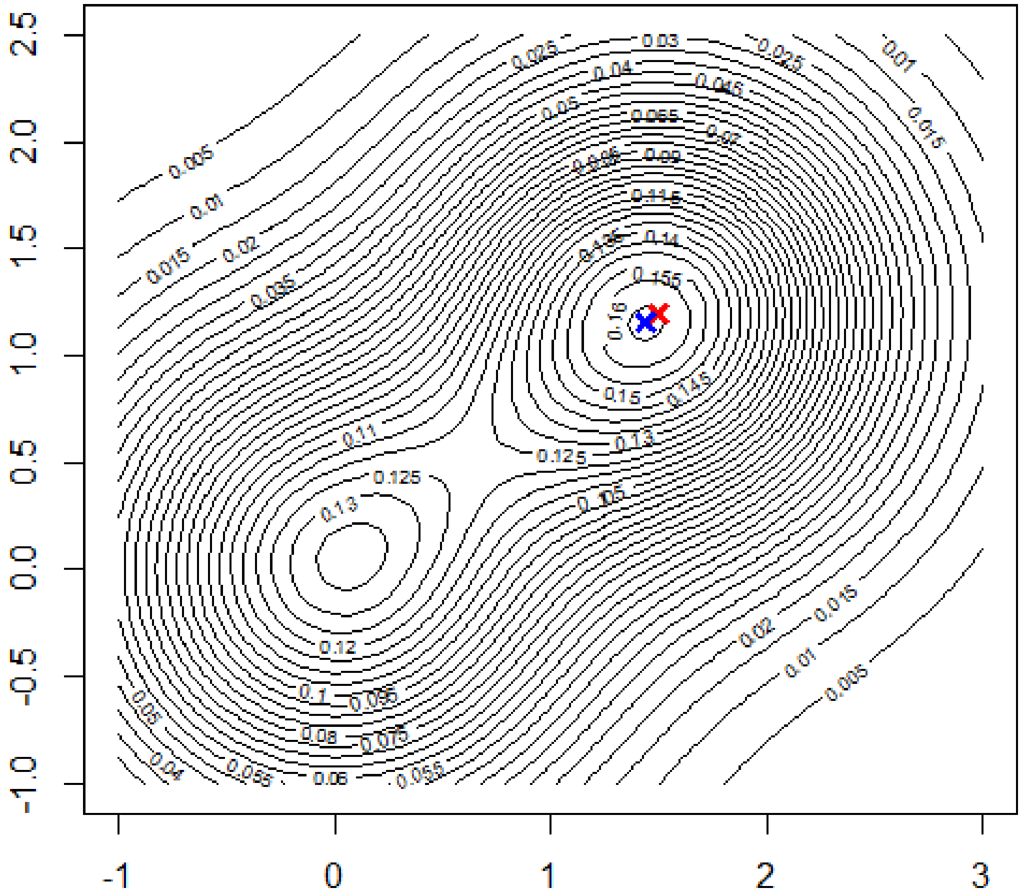
- $\boldsymbol{x}^{(\boldsymbol{t}+\mathbf{1})} = \boldsymbol{x}^{(\boldsymbol{t})} - \left(\boldsymbol{g}''\left(\boldsymbol{x}^{(\boldsymbol{t})}\right)\right)^{-1} \boldsymbol{g}'\left(\boldsymbol{x}^{(\boldsymbol{t})}\right)$
- We need ***g'*** and ***g''*** of

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

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

Multivariate Newton

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

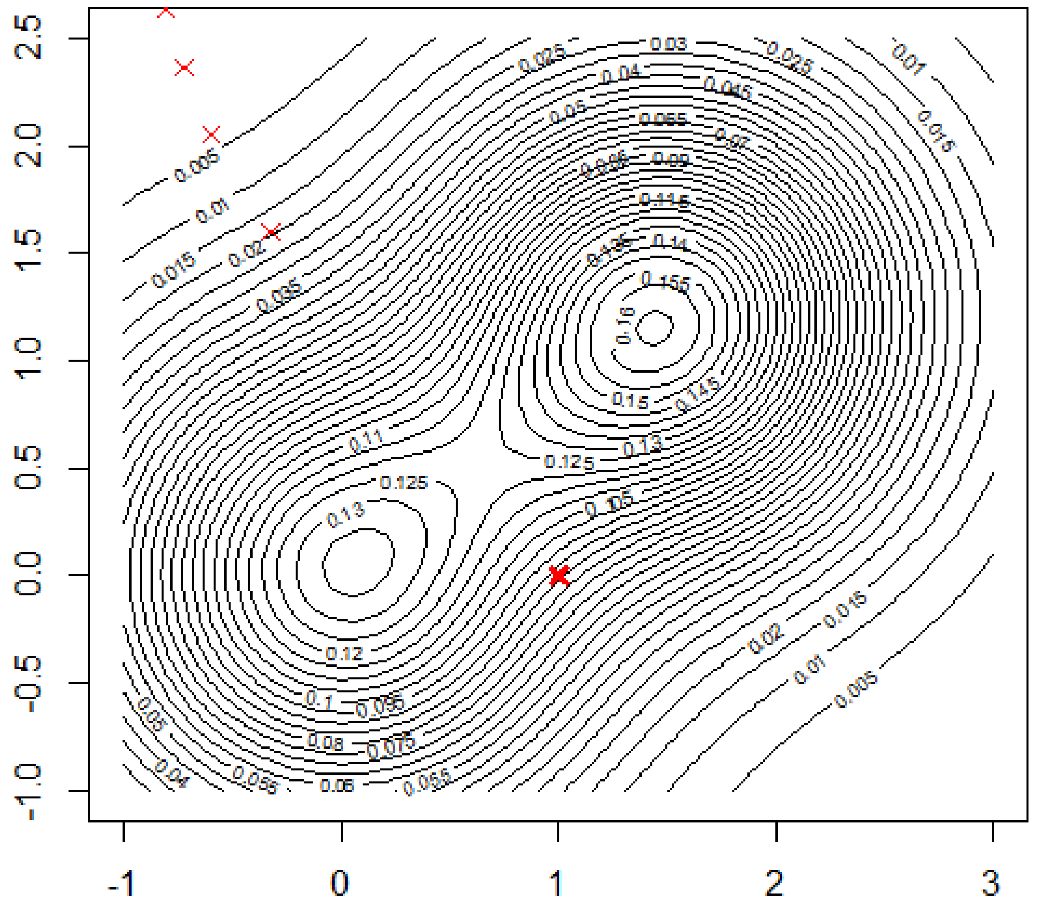


- Start with $\boldsymbol{x}^{(0)} = \begin{pmatrix} \mathbf{1.5} \\ \mathbf{1.2} \end{pmatrix}$
- $\boldsymbol{g}'(\boldsymbol{x}^{(0)}) = \begin{pmatrix} -0.0153 \\ -0.0123 \end{pmatrix}$
- $\boldsymbol{g}''(\boldsymbol{x}^{(0)}) = \begin{pmatrix} -0.2902 & 0.0306 \\ 0.0306 & -0.3040 \end{pmatrix}$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

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

Multivariate Newton

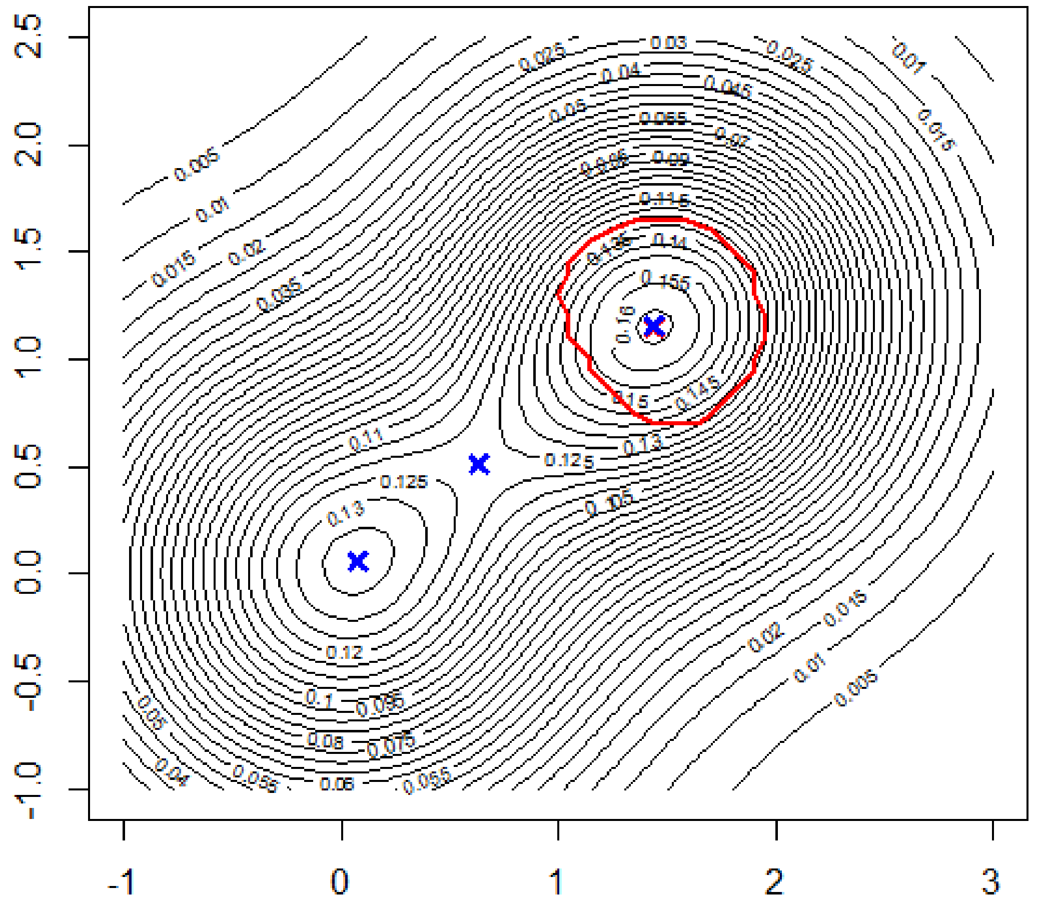
- $g(x_1, x_2) = \frac{1}{4\pi}\left(\frac{1}{0.6}e^{-({x_1}^2+{x_2}^2)/1.2} + \frac{1}{0.5}e^{-\left((x_1-1.5)^2+(x_2-1.2)^2\right)}\right)$



- Start with $\boldsymbol{x}^{(0)} = \begin{pmatrix} \mathbf{1} \\ \mathbf{0} \end{pmatrix}$
- $\boldsymbol{g}'(\boldsymbol{x}^{(0)}) = \begin{pmatrix} -0.0667 \\ +0.0705 \end{pmatrix}$
- $\boldsymbol{g}''(\boldsymbol{x}^{(0)}) = \begin{pmatrix} 0.0347 & 0.0705 \\ 0.0705 & 0.0144 \end{pmatrix}$
- $\left(\boldsymbol{g}''(\boldsymbol{x}^{(0)})\right)^{-1} \boldsymbol{g}'(\boldsymbol{x}^{(0)}) = \begin{pmatrix} \mathbf{1.33} \\ \mathbf{-1.60} \end{pmatrix}$
- $\boldsymbol{x}^{(1)} = \begin{pmatrix} \mathbf{1} \\ \mathbf{0} \end{pmatrix} - \begin{pmatrix} \mathbf{1.33} \\ \mathbf{-1.6} \end{pmatrix} = \begin{pmatrix} \mathbf{-0.33} \\ \mathbf{1.6} \end{pmatrix}$

Multivariate Newton

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



- 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

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

Today's schedule

- Multivariate Optimization
  - Analytical opt.
  - Newton
  - Steepest ascent
  - Quasi-Newton
  - Nelder-Mead

Steepest ascent method

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

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

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

- Other choices instead ***I*** in formula above possible
- We know that *g* will increase for small *α*

Backtracking line search (for steepest ascent)

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

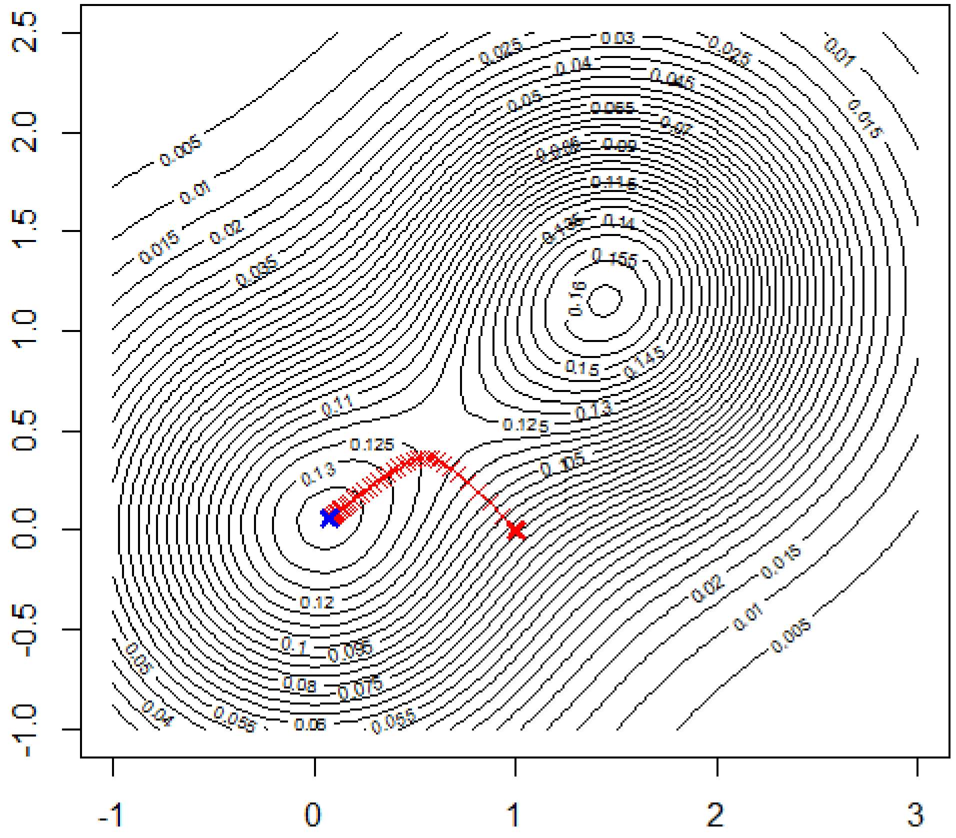
- We know that g will increase for small α

- Try $\alpha^{(t)} = \alpha_0$ first; α_0 could be set to 1
- If g decreases, half $\alpha^{(t)}$ until $g(\boldsymbol{x}^{(\boldsymbol{t}+\mathbf{1})})$ increases (backtracking)
- For the next iteration, either set $\alpha^{(t+1)} = \alpha_0$, or use the reduced α, $\alpha^{(t+1)} = \alpha^{(t)}$ and check again if backtracking is necessary

- Searching α such that g becomes maximal is a more sophisticated possibility

Steepest ascent

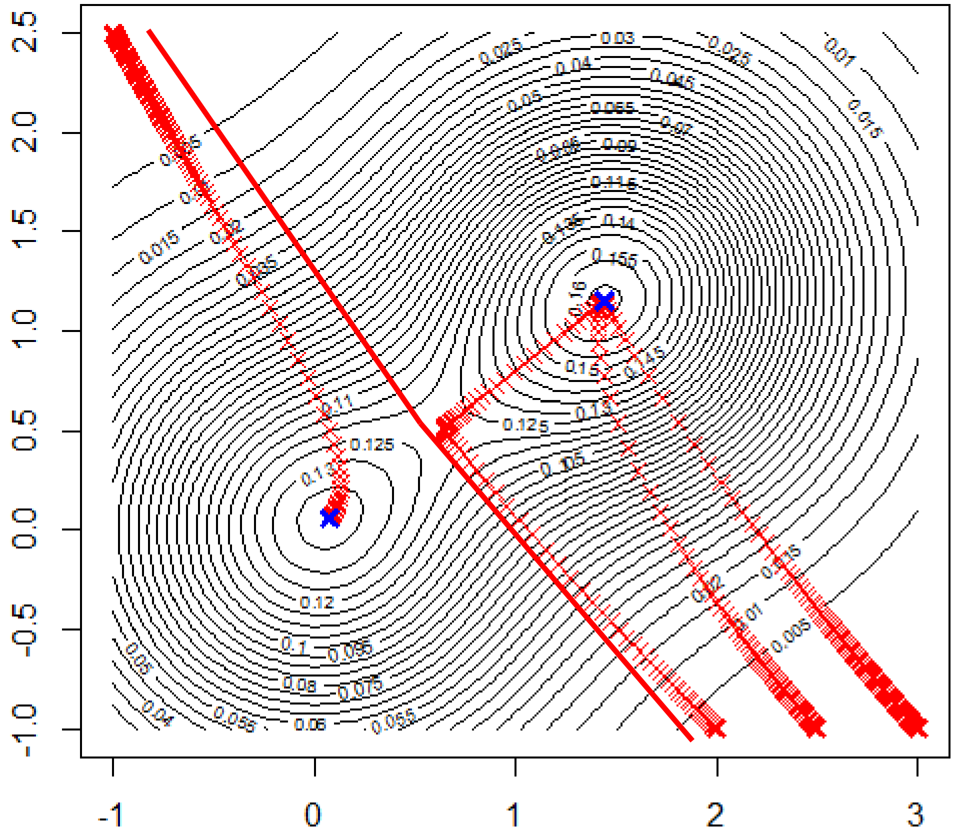
- $g(x_1, x_2) = \frac{1}{4\pi}\left(\frac{1}{0.6}e^{-({x_1}^2+{x_2}^2)/1.2} + \frac{1}{0.5}e^{-\left((x_1-1.5)^2+(x_2-1.2)^2\right)}\right)$



- Start with $\boldsymbol{x}^{(0)} = \begin{pmatrix}\mathbf{1}\\\mathbf{0}\end{pmatrix}$
- $\boldsymbol{g}'\left(\boldsymbol{x}^{(0)}\right) = \begin{pmatrix}-0.0667\\+0.0705\end{pmatrix}$
- $\boldsymbol{x}^{(1)} = \begin{pmatrix}\mathbf{1}\\\mathbf{0}\end{pmatrix}+\alpha^{(0)}\begin{pmatrix}-0.0667\\+0.0705\end{pmatrix} = \begin{pmatrix}\mathbf{0.9333}\\\mathbf{0.0705}\end{pmatrix}$

Steepest ascent

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



- 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 backtracking rule, convergence not always guaranteed

Today's schedule

- Multivariate Optimization
  - Analytical opt.
  - Newton
  - Steepest ascent
  - Quasi-Newton
  - Nelder-Mead

Quasi-Newton

- Steepest ascent and Newton method have iteration
$$\boldsymbol{x}^{(\boldsymbol{t}+\mathbf{1})} = \boldsymbol{x}^{(\boldsymbol{t})} - \left(\boldsymbol{M}^{(t)}\right)^{-1} \boldsymbol{g}'\left(\boldsymbol{x}^{(\boldsymbol{t})}\right)$$
with $\boldsymbol{M}^{(\boldsymbol{t})} = \boldsymbol{g}''\left(\boldsymbol{x}^{(\boldsymbol{t})}\right)$ for the Newton method and
with $\left(\boldsymbol{M}^{(\boldsymbol{t})}\right)^{-\mathbf{1}} = -\alpha^{(t)}\boldsymbol{I}$ for the steepest ascent method
- Disadvantage of Newton: Need to calculate Hessian $\boldsymbol{g}''\left(\boldsymbol{x}^{(\boldsymbol{t})}\right)$ in each iteration
- Disadvantage of steepest ascent: No information about curvature used
- We can monitor the computed gradients $\boldsymbol{g}'\left(\boldsymbol{x}^{(\boldsymbol{t})}\right)$ and their change gives information about the curvature of g

Quasi-Newton

- Steepest ascent and Newton method have iteration $\boldsymbol{x}^{(t+1)} = \boldsymbol{x}^{(t)} - \left(\boldsymbol{M}^{(t)}\right)^{-1}\boldsymbol{g}'\left(\boldsymbol{x}^{(t)}\right)$
- Newton ($\boldsymbol{M}^{(t)} = \boldsymbol{g}''\left(\boldsymbol{x}^{(t)}\right)$) was motivated with the multidimensional Taylor expansion
$$\boldsymbol{g}'(\boldsymbol{x}^*) \approx \boldsymbol{g}'\left(\boldsymbol{x}^{(t)}\right) + \boldsymbol{g}''\left(\boldsymbol{x}^{(t)}\right)\left(\boldsymbol{x}^* - \boldsymbol{x}^{(t)}\right)$$
or
$$\boldsymbol{g}'(\boldsymbol{x}^*) - \boldsymbol{g}'\left(\boldsymbol{x}^{(t)}\right) \approx \boldsymbol{g}''\left(\boldsymbol{x}^{(t)}\right)\left(\boldsymbol{x}^* - \boldsymbol{x}^{(t)}\right)$$
- Use approximations $\boldsymbol{M}^{(t+1)}$ to $\boldsymbol{g}''\left(\boldsymbol{x}^{(t)}\right)$ fulfilling this when $\boldsymbol{x}^*$ replaced by $\boldsymbol{x}^{(t+1)}$:
$$\underbrace{\boldsymbol{g}'\left(\boldsymbol{x}^{(t+1)}\right) - \boldsymbol{g}'\left(\boldsymbol{x}^{(t)}\right)}_{\boldsymbol{y}^{(t)}} = \boldsymbol{M}^{(t+1)}\underbrace{\left(\boldsymbol{x}^{(t+1)} - \boldsymbol{x}^{(t)}\right)}_{\boldsymbol{z}^{(t)}}$$
- This condition is called **secant condition**; there are multiple solutions to this condition; Broyden, Fletcher, Goldfarb, and Shanno (**BFGS**) solution:
$$\boldsymbol{M}^{(t+1)} = \boldsymbol{M}^{(t)} - \frac{\boldsymbol{M}^{(t)}\boldsymbol{z}^{(t)}\left(\boldsymbol{M}^{(t)}\boldsymbol{z}^{(t)}\right)^T}{\boldsymbol{z}^{(t)T}\boldsymbol{M}^{(t)}\boldsymbol{z}^{(t)}} + \frac{\boldsymbol{y}^{(t)}\boldsymbol{y}^{(t)T}}{\boldsymbol{y}^{(t)T}\boldsymbol{z}^{(t)}}$$

Quasi-Newton

- BFGS (quasi-Newton) method has iteration $\boldsymbol{x}^{(t+1)} = \boldsymbol{x}^{(t)} - \left(\boldsymbol{M}^{(t)}\right)^{-1} \boldsymbol{g}'\left(\boldsymbol{x}^{(t)}\right)$ with $\boldsymbol{M}^{(t+1)} = \boldsymbol{M}^{(t)} - \frac{\boldsymbol{M}^{(t)}\boldsymbol{z}^{(t)}\left(\boldsymbol{M}^{(t)}\boldsymbol{z}^{(t)}\right)^T}{\boldsymbol{z}^{(t)T}\boldsymbol{M}^{(t)}\boldsymbol{z}^{(t)}} + \frac{\boldsymbol{y}^{(t)}\boldsymbol{y}^{(t)T}}{\boldsymbol{y}^{(t)T}\boldsymbol{z}^{(t)}}$
- Initial $\boldsymbol{M}^{(1)}$ can be set e.g. to the identity matrix
- Ascent not ensured but backtracking can be used to ensure it:

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

- The **`R`** function **`optim`** includes the quasi-Newton BFGS
- Convergence of quasi-Newton methods usually faster than linear but slower than quadratic

Convergence order for deterministic algorithms

- Recall: Convergence order and convergence rate

$$\frac{\{g(\boldsymbol{x}^{(t+1)}) - g(\boldsymbol{x}^*)\}}{\{g(\boldsymbol{x}^{(t)}) - g(\boldsymbol{x}^*)\}^q} \rightarrow c \quad (\text{for } t \rightarrow \infty)$$

- q is convergence order (q=1, 0<c<1 linear; q=2, 0<c<1 quadratic)
- c is convergence rate
- Under certain assumption, we have following orders:

Uni-dimensional	**Bisection** order = roughly 1*		**Secant** order = $(1+\sqrt{5})/2$	**Newton** order = 2
Multi-dimensional		**Steepest ascent** order = 1	**Quasi-Newton** order > 1**	**Newton** order = 2

*strictly, the above criterion cannot be proven for bisection
**criterion above fulfilled for q=1 and c=0; "superlinear"

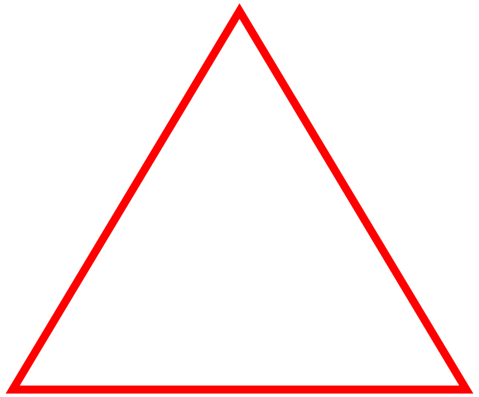
Convergence speed for an example function

- The convergence of BFGS and Newton can be extremely fast in praxis compared to steepest ascent/descent
- Example from Nocedal and Wright (2006), chapter 6: Rosenbrock function $g(\boldsymbol{x}) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$, starting point (-1.2, 1), minimum at (1,1).
  #iterations until error < 10^{-5}:

• Steepest descent	5264
• BFGS	34
• Newton	21

Today's schedule

- Multivariate Optimization
  - Analytical opt.
  - Newton
  - Steepest ascent
  - Quasi-Newton
  - Nelder-Mead



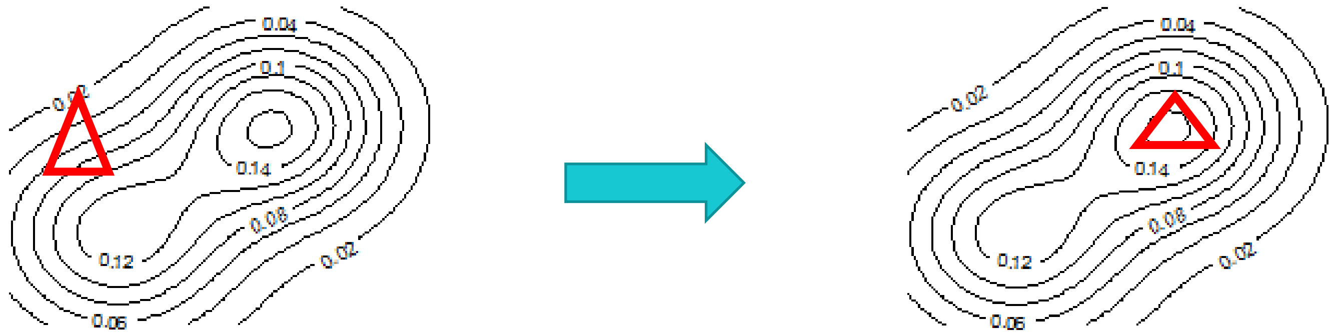
Nelder-Mead algorithm

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

- Nelder-Mead method is heuristic method for p-dimensional optimization problem (default in **`R`**-function **`optim`**)
- Advantage/disadvantages:
  - + No computation of derivatives necessary
  - - No theoretical guarantee for convergence (counter examples exist)
  - - Might be slow
- Works often well, especially if p not too large

Nelder-Mead algorithm

- Idea: Work with simplex of p+1 points; i.e., for two-dimensional cases: triangle
- Aim that triangle includes maximum
- Choose arbitrary starting triangle
- Change vertices to ”move the triangle upwards”



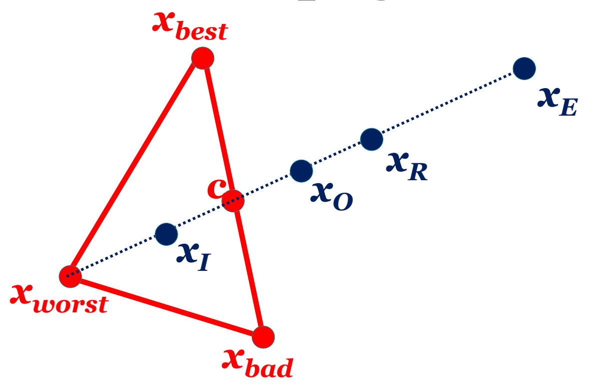
- Two animations:
  - https://upload.wikimedia.org/wikipedia/commons/9/96/Nelder_Mead2.gif
  - https://www.youtube.com/watch?v=KEGSLQ6TlBM

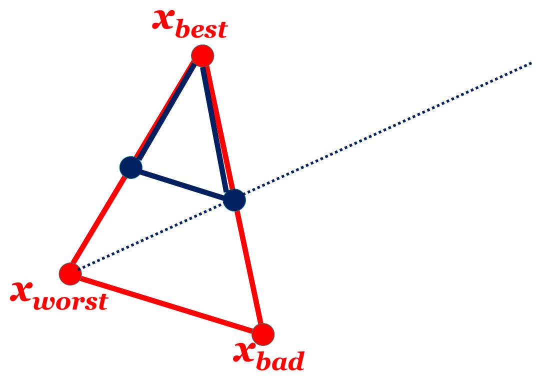
Nelder-Mead algorithm

- Identify worst vertex $\boldsymbol{x_{worst}}$ ($g(\boldsymbol{x_{worst}})$ minimal among all vertices) and compute average $\boldsymbol{c}$ of remaining vertices
- Let $\boldsymbol{x_{best}}$ be best and $\boldsymbol{x_{bad}}$ be second worst vertex
- Rules for
  - Reflection
  - Expansion
  - Outer contraction
  - Inner contraction
  - Shrinkage

Nelder-Mead algorithm

- Replace $\boldsymbol{x}_{worst}$ with one of $\boldsymbol{x}_I$, $\boldsymbol{x}_O$, $\boldsymbol{x}_R$, $\boldsymbol{x}_E$ (rule depends on values for $g(\boldsymbol{x}_{worst})$, $g(\boldsymbol{x}_{bad})$, $g(\boldsymbol{x}_{best})$, $g(\boldsymbol{x}_I)$, $g(\boldsymbol{x}_O)$, $g(\boldsymbol{x}_R)$, $g(\boldsymbol{x}_E)$; see Givens and Hoeting, page 47-48; Gentle, page 273) and create new simplex/triangle

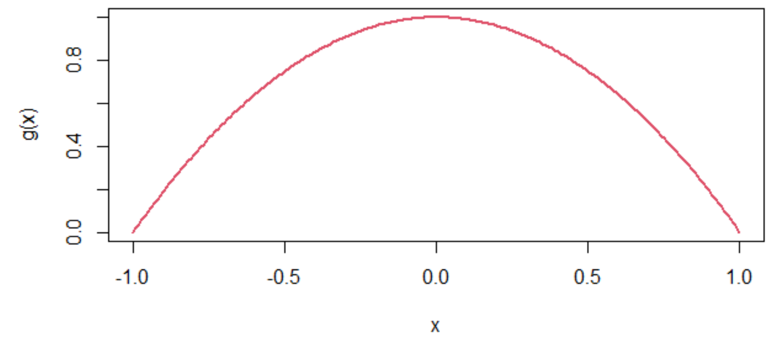




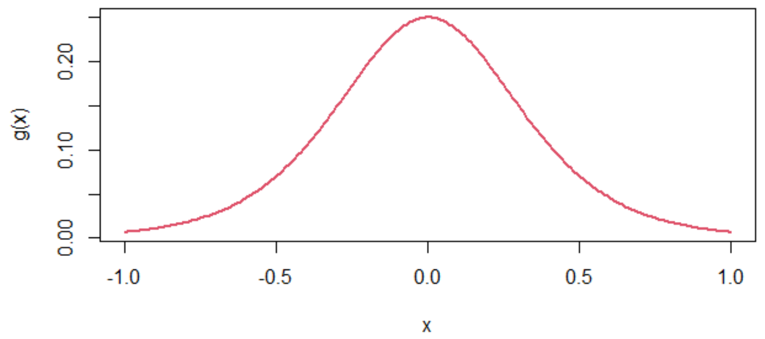
- Or in specific cases: Shrink (keep $\boldsymbol{x}_{best}$ and move all other vertices towards it)
- Another animation: https://www.youtube.com/watch?v=j2gcuRVbwR0

Convexity / Concavity and log likelihood

- Function g concave, if $g((\boldsymbol{x}+\boldsymbol{y})/2) \geq (g(\boldsymbol{x})+g(\boldsymbol{y}))/2$ for all $\boldsymbol{x}, \boldsymbol{y}$



concave



non-concave

- If g is concave, a local maximum is a global maximum
- Log likelihood for exponential families is concave
- Log likelihoods can be non-concave (e.g., Cauchy-distribution)
- Deep learning optimization problems are often non-concave / non-convex and have multiple local extrema