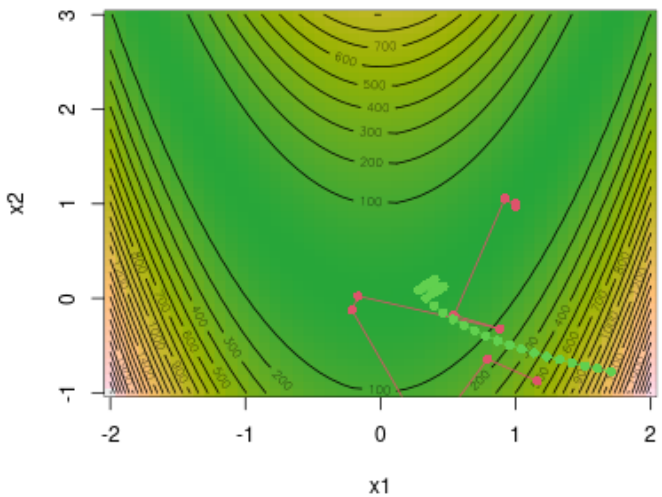
Optimization in Machine Learning

Second order methods: Newton-Raphson



Learning goals

- 1st vs. 2nd order methods
- Newton-Raphson

1ST AND 2ND ORDER PROCEDURES

Until now: **1st order methods**. They use gradient info, so 1st derivative.

Here: **2nd order methods**. Use Hessian, so 2nd derivative.

NEWTON-RAPHSON

Assumption: f twice differentiable with Hessian $\nabla^2 f(\boldsymbol{x})$

Aim: Find stationary point $\nabla f(\boldsymbol{x}) = \mathbf{0}$

Idea: Find root of Taylor approximation (1st order) of $\nabla f(\boldsymbol{x})$:

$$\begin{aligned} \nabla f(\mathbf{x}) \approx \nabla f(\mathbf{x}^{[t]}) + \nabla^2 f(\mathbf{x}^{[t]})(\mathbf{x} - \mathbf{x}^{[t]}) &= \mathbf{0} \\ \nabla^2 f(\mathbf{x}^{[t]})(\mathbf{x} - \mathbf{x}^{[t]}) &= -\nabla f(\mathbf{x}^{[t]}) \\ \mathbf{x}^{[t+1]} &= \mathbf{x}^{[t]} \underbrace{- \left(\nabla^2 f(\mathbf{x}^{[t]})\right)^{-1} \nabla f(\mathbf{x}^{[t]})}_{:= d^{[t]}} \end{aligned}$$

Update: $\boldsymbol{x}^{[t+1]} = \boldsymbol{x}^{[t]} + \boldsymbol{d}^{[t]}$ with $\boldsymbol{d}^{[t]} = \left(\nabla^2 f(\mathbf{x}^{[t]})\right)^{-1} \nabla f(\mathbf{x}^{[t]})$.
Or with step size: $\boldsymbol{x}^{[t+1]} = \boldsymbol{x}^{[t]} + \alpha \boldsymbol{d}^{[t]}$

Note: Numerically, we determine $\boldsymbol{d}^{[t]}$ by solving the LGS $\nabla^2 f(\mathbf{x}^{[t]}) \boldsymbol{d}^{[t]} = -\nabla f(\mathbf{x}^{[t]})$ with symmetic matrix $\nabla^2 f(\mathbf{x}^{[t]})$.

ANALYTICAL EXAMPLE ON QF

$$f(x, y) = \left(x^2 + \frac{y^2}{2}\right)$$

Update direction:

$$\mathbf{d}^{[t]} = -\left(\nabla^2 f(x^{[t]}, y^{[t]})\right)^{-1} \nabla f(x^{[t]}, y^{[t]})$$

$$\begin{aligned}
\nabla f(x, y) &= \begin{pmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{pmatrix} = \begin{pmatrix} 2x \\ y \end{pmatrix} \\
\nabla^2 f(x, y) &= \begin{pmatrix} \frac{\partial^2 f(x,y)}{\partial^2 x} & \frac{\partial^2 f(x,y)}{\partial x \partial y} \\ \frac{\partial^2 f(x,y)}{\partial y \partial x} & \frac{\partial^2 f(x,y)}{\partial^2 y} \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}
\end{aligned}$$

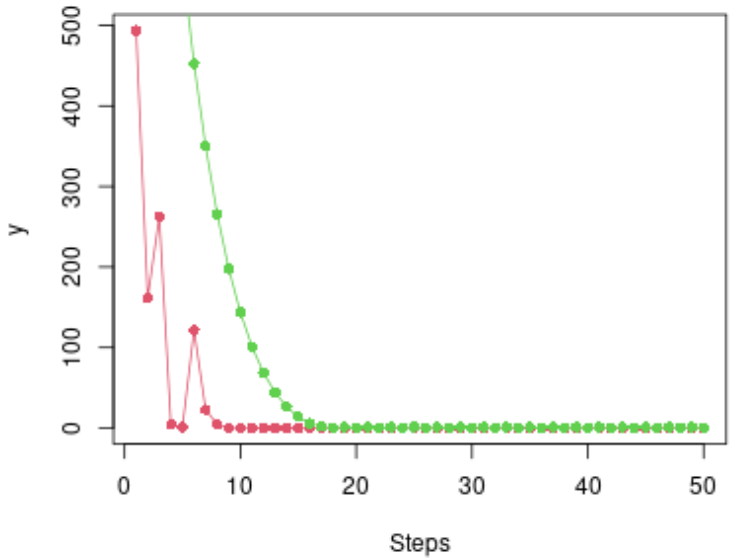
ANALYTICAL EXAMPLE ON QF

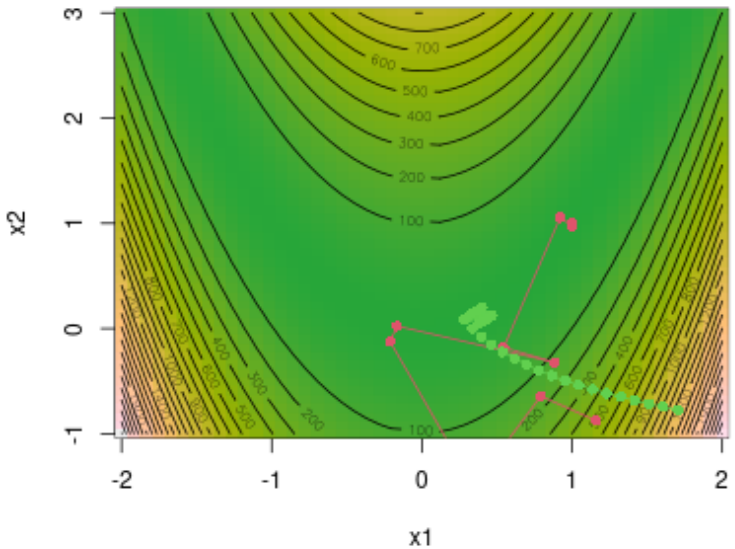
`t = 1:`

$$\begin{aligned}
\begin{pmatrix} x^{[1]} \\ y^{[1]} \end{pmatrix} &= \begin{pmatrix} x^{[0]} \\ y^{[0]} \end{pmatrix} + \mathbf{d}^{[0]} = \begin{pmatrix} x^{[0]} \\ y^{[0]} \end{pmatrix} - \begin{pmatrix} 0.5 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 2x^{[0]} \\ y^{[0]} \end{pmatrix} \\
&= \begin{pmatrix} x^{[0]} \\ y^{[0]} \end{pmatrix} + \begin{pmatrix} -x^{[0]} \\ -y^{[0]} \end{pmatrix} \\
&= \mathbf{0}
\end{aligned}$$

NR only needs one iteration to solve!

NR VS GD ON BRANIN





Red=NR; Green=GD

NR has much better convergence speed here.

DISCUSSION

Advantages:

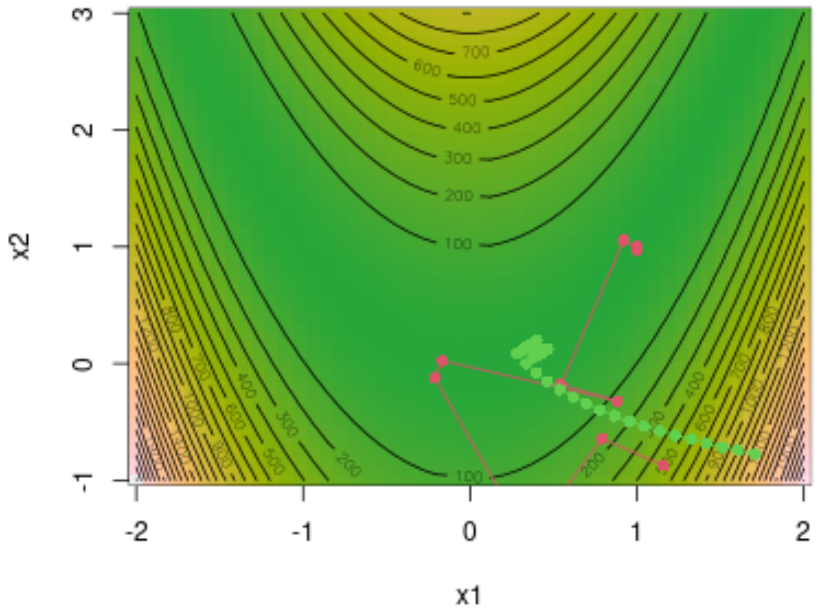
- If f sufficiently smooth, NR converges quadratically locally (i.e. if starting point is close enough to optimum)

Disadvantages

- At “bad” starting points, NR may not converge at all
- Hessian must be calculated and the direction of descent determined by solving a system of equations

LIMITATIONS

Problem 1: Update is generally not a direction of descent.



But: If Hessian is positive definite, it is necessarily descent direction:

$$(\mathbf{d}^{[t]})^\top \nabla f(\mathbf{x}^{[t]}) = -\left(\nabla f(\mathbf{x}^{[t]})\right)^\top \left(\nabla^2 f(\mathbf{x}^{[t]})\right)^{-1} \nabla f(\mathbf{x}^{[t]}) < 0.$$

Near the minimum, Hessian is p.d.. especially at beginning Hessian is often not p.d. and the Newton-Raphson update direction is not sensible.

LIMITATIONS

Problem 2: The calculation of the Hessian can be **expensive** and the calculation of the descent direction by solving the system of equations

$$\left(\nabla^2 f(\mathbf{x}^{[t]})\right) \mathbf{d}^{[t]} = -\nabla f(\mathbf{x}^{[t]})$$

can be numerically unstable.

Aim: Find methods that can be applied without the Hessian matrix

- Quasi-Newton method
- Gauss-Newton algorithm (for least squares)