Numerical Methods for Computational Science and Engineering

Autumn Semester 2018, Week 14

Prof. Rima Alaifari, SAM, ETH Zurich

Gradient descent cont'd:

Gradient descent iteration:

$$x^{(k+1)} = x^{(k)} - t^{(k)} \nabla F(x^{(k)})$$

$t^{(k)}$: finding step size: problem in 1D

Algorithm:

- start with initial guess $x^{(0)}$
- while stopping criterion is not satisfied
  e.g. $\|\nabla F(x^{(k)})\|_2 > tol$

1.) take $g^{(k)}(t) = F(x^{(k)} - t \nabla F(x^{(k)}))$

2.) ideally: find step size t^* through a line search

e.g. $t^* = \underset{t \geq 0}{\operatorname{argmin}}\, g^{(k)}(t)$

3.) take $x^{(k+1)} = x^{(k)} - t^* \nabla F(x^{(k)})$

In each iteration $F(x^{(k)})$ decreases

we terminate when $\nabla F(x^{(k)}) \approx 0$

How to find t^* ?

- exact line search $\left(t^* = \underset{t \geq 0}{\operatorname{argmin}}\, g^{(k)}(t)\right)$
- backtracking line search

$$F(x - t\nabla F(x)) \approx F(x) - t\|\nabla F(x)\|^2$$

↑ for t small enough

$$< F(x) - \alpha t \|\nabla F(x)\|^2$$

↑ $\alpha \in (0,1)$

Idea: Start with $t = 1$, fixed $\alpha \in (0, 0.5)$:

decrease t (e.g. $t \leftarrow \frac{t}{2}$) until

$$F(x - t\nabla F(x)) < F(x) - \alpha t \|\nabla F(x)\|^2$$

Roughly: iterate until "good decrease" is reached

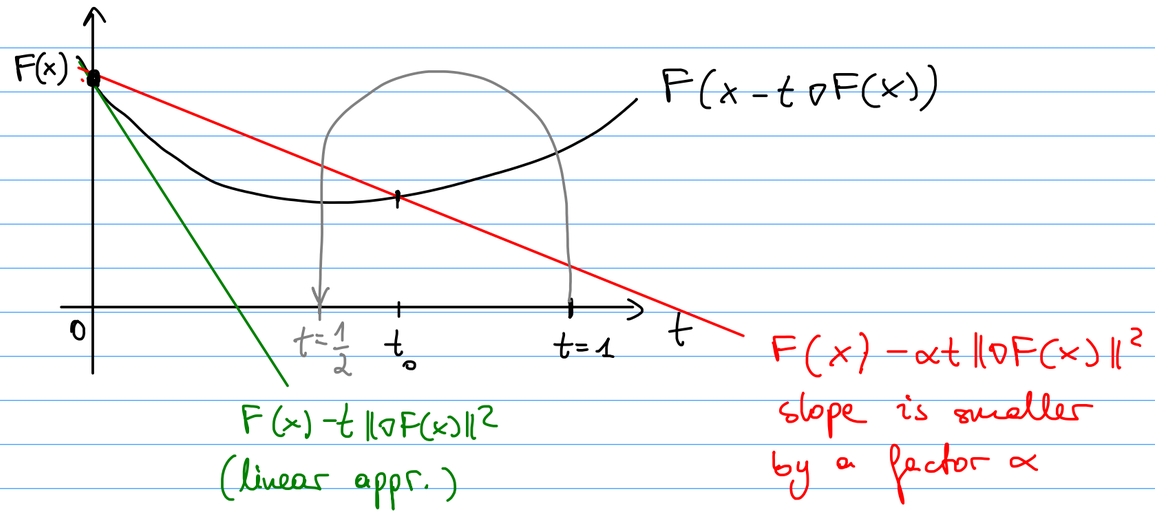
We iterate while: $F(x - t\nabla F(x)) > F(x) - \alpha t \|\nabla F(x)\|^2$

this guarantees:

$$F(x^{(k+1)}) < F(x^{(k)}) - \alpha t \|\nabla F(x^{(k)})\|^2$$

$$\Rightarrow \quad F(x^{(k)}) - F(x^{(k+1)}) > \alpha t \|\nabla F(x^{(k)})\|^2 > 0$$

$\Rightarrow$ decrease in F



backtracking: start at $t = 1$ and stop when

$t \leq t_0$ for the first time

Note: By convexity:

$$F(x - t\nabla F(x)) < F(x) - t\|\nabla F(x)\|^2$$

can never happen!

i.e. the linear approximation $F(x) - t\|\nabla F(x)\|^2$ will always lie below the function graph

Newton's method for optimization

If F is twice diff.:

$$F(x) \approx F(x^{(k)}) + \nabla F(x^{(k)})^T (x - x^{(k)}) + \frac{1}{2}(x - x^{(k)})^T H_F(x^{(k)})(x - x^{(k)})$$

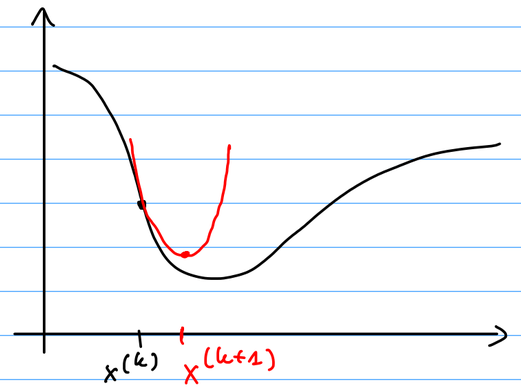
Differentiate RHS & set to zero

(minimum of quadr. approximation):

$$x^{(k+1)} = x^{(k)} - (H_F(x^{(k)}))^{-1} \nabla F(x^{(k)})$$

$\rightsquigarrow$ root finding for ∇F!

In 1D:



$$x^{(k+1)} = x^{(k)} - \frac{f'(x^{(k)})}{f''(x^{(k)})}$$

↑ vertex of parabola

Near minimum: quadratic convergence
[faster than gradient descent]

Gradient descent:

- line search in every step
- typically converges on a larger region

Newton's method:

- compute H_f & solve LSE in each step
- Newton's method typically needs fewer iteration

Note: both can get stuck in local minimum

BFGS method (Quasi-Newton method)

Instead of computing & solving for the Hessian $H_f(x^{(k)})$: approximate it by B_k s.t. B_{k+1} is obtained from a simple update of B_k!

Newton's method:

$$x^{(k+1)} = x^{(k)} - \left[H_F(x^{(k)})\right]^{-1} \nabla F(x^{(k)})$$

Approximation B_k of $H_F(x^{(k)})$:

secant-like condition as for Broyden's method:

$$B_{k+1}\underbrace{\left(x^{(k+1)} - x^{(k)}\right)}_{=:s^{(k)}} = \underbrace{\nabla F(x^{(k+1)}) - \nabla F(x^{(k)})}_{=:y^{(k)}}$$

$$\Leftrightarrow \quad B_{k+1} s^{(k)} = y^{(k)} \qquad (*)$$

Additionally: B_{k+1} should be s.p.d.

BFGS update:

$$B_{k+1} = B_k + \alpha u u^T + \beta v v^T \qquad \text{s.t. } (*) \text{ holds}$$

with choice: $u = y^{(k)}$, $v = B_k s^{(k)}$

$$\alpha = \frac{1}{y^{(k)T} s^{(k)}}, \quad \beta = -\frac{1}{s^{(k)T}\left(B_k s^{(k)}\right)}$$

$$\Rightarrow \quad \left(B_k + \alpha u u^T + \beta v v^T\right) s^{(k)} = y^{(k)}$$

BFGS update on B_k:

$$B_{k+1} = B_k + \frac{y^{(k)} y^{(k)T}}{y^{(k)T} s^{(k)}} - \frac{B_k s^{(k)} s^{(k)T} B_k^T}{s^{(k)T}\left(B_k s^{(k)}\right)}$$

With Sherman–Morrison–Woodbury formula:

$$B_{k+1}^{-1} = \left(I - \frac{s^{(k)} y^{(k)T}}{y^{(k)T} s^{(k)}} \right) B_k^{-1} \left(I - \frac{y^{(k)} s^{(k)T}}{y^{(k)T} s^{(k)}} \right) + \frac{s^{(k)} s^{(k)T}}{y^{(k)T} s^{(k)}}$$

[Last remark: L–BFGS variant
↑
limited memory
⇝ no storage of dense matrices B_k]