Computational Statistics
Optimisation

Luca Bortolussi

Department of Mathematics and Geosciences
University of Trieste

Office 238, third floor, H2bis
luca@dmi.units.it

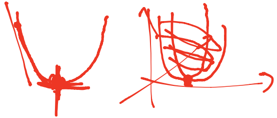
Trieste, Winter Semester 2015/2016

Outline

1. Gradient-based Methods
2. Newton's Methods

BASICS

- Consider a function $f(\mathbf{x})$, from $\mathbb{R}^n$ to $\mathbb{R}$, twice differentiable. Their minima are points such that $\nabla f(\mathbf{x}) = 0$.
- At a minimum $\mathbf{x}^*$ of f, the Hessian matrix $H_f(\mathbf{x}^*)$ is positive semidefinite, i.e. $\boxed{\mathbf{v}^T H_f \mathbf{v} \geq 0.}$
- If a point $\mathbf{x}^*$ is such that (a) $\nabla f(\mathbf{x}) = 0$ and (b) $H_f(\mathbf{x}^*)$ is positive definite, then $\mathbf{x}^*$ is a minimum of f.
- For a quadratic function $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x} + c$ the condition $\nabla f(\mathbf{x}) = 0$ reads $\boxed{A\mathbf{x} - \mathbf{b} = 0.}$
- If A is invertible and positive definite, then the point $\mathbf{x}^* = A^{-1}\mathbf{b}$ is the unique minimum of f, as f is a convex function.



GRADIENT DESCENT

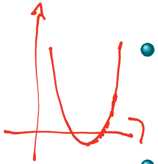
FIX $x_0 \in \mathbb{R}^n$ (often $x_0 = 0$)

- Notation. $\mathbf{x}_k$ denotes the sequence of points of the descent. $\mathbf{g}_k = \boxed{\nabla f(\mathbf{x}_k)}$. The update is in the direction $\mathbf{p}_k$:

$$\boxed{\mathbf{x}_{k+1} = \mathbf{x}_k + \boxed{\eta_k \mathbf{p}_k}}$$

η_k - LEARNING RATE

- In gradient descent, at a point $\mathbf{x}$, take a step towards $-\nabla f(\mathbf{x})$, hence in the update rule becomes we set $\boxed{\mathbf{p}_k = -\mathbf{g}_k.}$



- In the simplest case, $\eta_k = \eta$. If η is not small enough, we can step over the minimum. If η is very small this usually not happens, but convergence is very slow.
- An improvement of convergence is to set $\mathbf{p}_k = -\mathbf{g}_k - \beta \mathbf{g}_{k-1}$, with $0 < \beta < 1$ the momentum coefficient.
- For a quadratic function, we have that $\mathbf{p}_k = -A\mathbf{x}_k + \mathbf{b}$.

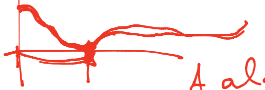
STOCHASTIC GRADIENT DESCENT

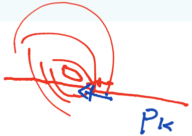
If $N \gg 1$, compute ∇f is costly

- If the function to minimise is of the form $\boxed{f(\mathbf{x}) = \sum_{i=1}^{N} f_i(\mathbf{x}),}$ as is the case for ML problems, then we can use stochastic gradient descent, which instead of taking a step along $\mathbf{g}_k$, it steps along the direction $-\nabla f_i(\mathbf{x_k})$.
- The algorithm iterates over the dataset one or more times, typically permuting it each time.
- The learning rate η_k can be takes as constant or be decreased every (m) iterations, to improve convergence.
- Alternatively to one single observations, small batches of n_t observations can be used to improve the method.

i.e. a small FIXED number $\ll N$

GRADIENT DESCENT WITH LINE SEARCH





f along p_k

- One possibility to improve gradient descent is to take the best step possible, i.e. set η_k to a value minimising the function $\boxed{f(\mathbf{x}_k + \lambda \mathbf{p}_k)}$ along the line with direction $\mathbf{p}_k$.
- The minimum is obtained by solving for λ the equation

$$\boxed{\nabla f(\mathbf{x}_k + \lambda \mathbf{p}_k)^T \mathbf{p}_k = \mathbf{g}_{k+1}^T \mathbf{p}_k = 0}$$

  and setting η_k to this solution.
- for a quadratic function, we have that the best learning rate is given by

$$\boxed{\eta_k = \frac{(\mathbf{b} - A\mathbf{x}_k)^T \mathbf{p}_k}{\mathbf{p}_k^T A \mathbf{p}_k}}$$

CONJUGATE GRADIENTS

$\left| \sum_i a_i x_i^2 - b_i x_i + a \right|$

$f(x) = x^T A x - b^T x + c$

- Consider a quadratic minimisation problem. If the matrix A would be diagonal, we could solve separately n different 1-dimensional optimisation problems.
- We can change coordinates by an orthogonal matrix P that diagonalises the matrix A. By letting $\mathbf{x} = P\mathbf{y}$, we can rewrite the function $f(\mathbf{x})$ as

$$f(\mathbf{y}) = \frac{1}{2}\mathbf{y}^T P^T A P \mathbf{y} - \mathbf{b}^T P \mathbf{y} + c$$

- The columns of P are called conjugate vectors and satisfy $\mathbf{p}_i^T A \mathbf{p}_j = 0$ and $\mathbf{p}_i^T A \mathbf{p}_i > 0$. They are linearly independent and are very good directions to follows in a descent method.

Conjugate gradients

- To construct conjugate vectors, we can use the Gram-Schmidt orthogonalisation procedure: if $\mathbf{v}$ is linearly independent of $\mathbf{p}_1, \ldots, \mathbf{p}_k$, then

$$\boxed{\mathbf{p}_{k+1} = \mathbf{v} - \sum_{j=1}^{k} \frac{\mathbf{p}_j^T A \mathbf{v}}{\mathbf{p}_j^T A \mathbf{p}_j} \mathbf{p}_j}$$

- We can start from a basis and construct the conjugate vectors $\mathbf{p}_1, \ldots, \mathbf{p}_n$.
- In the conjugate vectors algorithm, we take step $k+1$ along $\mathbf{p}_{k+1}$. The best η_k, according to line search, is

$$\eta_k = \frac{-\mathbf{p}_k^T \mathbf{g}_k}{\mathbf{p}_k^T A \mathbf{p}_k}$$

- It holds that $\nabla f(\mathbf{x}_{k+1})^T \mathbf{p}_i = 0$ for all $i = 1, \ldots, k$ (Lunenberg expanding subspace theorem).

Conjugate gradients

- The conjugate gradients method constructs $\mathbf{p}_k$'s on the fly. Works well also for non-quadratic problems. For quadratic problems converges in at most n steps.
- A good choice for a linearly independent vector $\mathbf{v}$ at step $k+1$ to construct $\mathbf{p}_{k+1}$ is thus $\boxed{\nabla f(\mathbf{x}_{k+1})}$.
- In this case, after some algebra, we can compute:

$$\alpha_{k+1} = \boxed{\frac{\mathbf{g}_{k+1}^T \mathbf{g}_{k+1}}{\mathbf{p}_{k+1}^T A \mathbf{p}_{k+1}}}$$

$$\boxed{\mathbf{p}_{k+1} = -\mathbf{g}_{k+1} + \beta_k \mathbf{p}_k}$$

with

$$\beta_k = \frac{\mathbf{g}_{k+1}^T \mathbf{g}_{k+1}}{\mathbf{g}_k^T \mathbf{g}_k} \quad \text{or} \quad \boxed{\beta_k = \frac{\mathbf{g}_{k+1}^T (\mathbf{g}_{k+1} - \mathbf{g}_k)}{\mathbf{g}_k^T \mathbf{g}_k}}$$

known as the Fletcher-Reeves or Polak-Ribière (preferrable for non-quadratic problems) formulae .

Outline

NEWTON-RAPSON METHOD

- As an alternative optimisation for small n, we can use the Newton-Rapson method, which has better convergence properties than gradient descent.
- By Taylor expansion

$$f(\mathbf{x}+\Delta) \approx \boxed{f(\mathbf{x}) + \Delta^T \nabla f(\mathbf{x}) + \frac{1}{2}\Delta^T H_f(\mathbf{x})\Delta}$$

  where $\mathbf{H}_f$ is the Hessian of $f(\mathbf{x})$.
- Differentiating w.r.t. Δ, the minimum of the r.h.s. is when $\boxed{\nabla f(\mathbf{x}) = -H_f(\mathbf{x})\Delta,}$ hence for $\boxed{\Delta = -\mathbf{H}_f^{-1}(\mathbf{x})\nabla f(\mathbf{x})}$
- Thus we obtain the update rule:

$$\boxed{\mathbf{x}_{k+1} = \mathbf{x}_k - \eta \mathbf{H}_f^{-1}(\mathbf{x}_k)\nabla f(\mathbf{x}_k)}$$

  with $0 < \eta < 1$ to improve convergence.
- Compute the update for a quadratic problem