The Foundations of (Machine) Learning

Stefan Kühn

codecentric AG

CSMLS Meetup Hamburg - February 26th, 2015

Contents

1. Supervised Machine Learning
2. Optimization Theory
3. Concrete Optimization Methods

Setting

Supervised Learning Approach

Use labeled training data in order to fit a given model to the data, i.e. to learn from the given data.

Typical Problems:

- Classification - discrete output
  - Logistic Regression
  - Neural Networks
  - Support Vector Machines
- Regression - continuous output
  - Linear Regression
  - Support Vector Regression
  - Generalized Linear/Additive Models

Training and Learning

Ingredients:

- Training Data Set
- Model, e.g. Logistic Regression
- Error Measure, e.g. Mean Squared Error

Learning Procedure:

- Derive objective function from Model and Error Measure
- Initialize Model parameters
- Find a good fit!
- Iterate with other initial parameters

What is Learning in this context?

Learning is nothing but the application of an algorithm for unconstrained optimization to the given objective function.

Unconstrained Optimization

Higher-order methods

- Newton's method (fast local convergence)

Gradient-based methods

- Gradient Descent / Steepest Descent (globally convergent)
- Conjugate Gradient (globally convergent)
- Gauß-Newton, Levenberg-Marquardt, Quasi-Newton
- Krylow Subspace methods

Derivative-free methods, direct search

- Secant method (locally convergent)
- Regula Falsi and successors (global convergence, typically slow)
- Nelder-Mead / Downhill-Simplex
  - unconventional method, creates a moving simplex
  - driven by reflection/contraction/expansion of the corner points
  - globally convergent for differentiable functions $f \in \mathcal{C}^1$

General Iterative Algorithmic Scheme

Goal: Minimize a given function f:

$$\min f(x),\ \ x \in \mathbb{R}^n$$

Iterative Algorithms

Starting from a given point an iterative algorithm tries to minimize the objective function step by step.

Preparation: $k = 0$

- Initialization: Choose initial points and parameters

Iterate until convergence: $k = 1, 2, 3, \ldots$

- Termination criterion: Check optimality of the current iterate
- Descent Direction: Find reasonable search direction
- Stepsize: Determine length of the step in the given direction

Termination criteria

Critical points x^*:

$$\nabla f(x^*) = 0$$

- Gradient: Should converge to zero

$$\|\nabla f(x^*)\| < tol$$

- Iterates: Distance between x^k and x^{k+1} should converge to zero

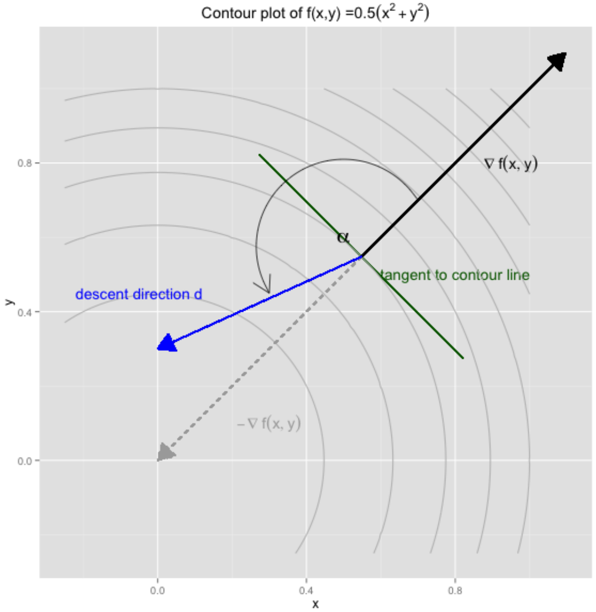
$$\left\|x^k - x^{k+1}\right\| < tol$$

- Function Values: Difference between $f(x^k)$ and $f(x^{k+1})$ should converge to zero

$$\left|f(x^k) - f(x^{k+1})\right| < tol$$

- Number of iterations: Terminate after *maxiter* iterations

Descent direction



Descent direction

Geometric interpretation

d is a descent direction if and only if the angle α between the gradient $\nabla f(x)$ and d is in a certain range:

$$\frac{\pi}{2} = 90^\circ < \alpha < 270^\circ = \frac{3\pi}{2}$$

Algebraic equivalent

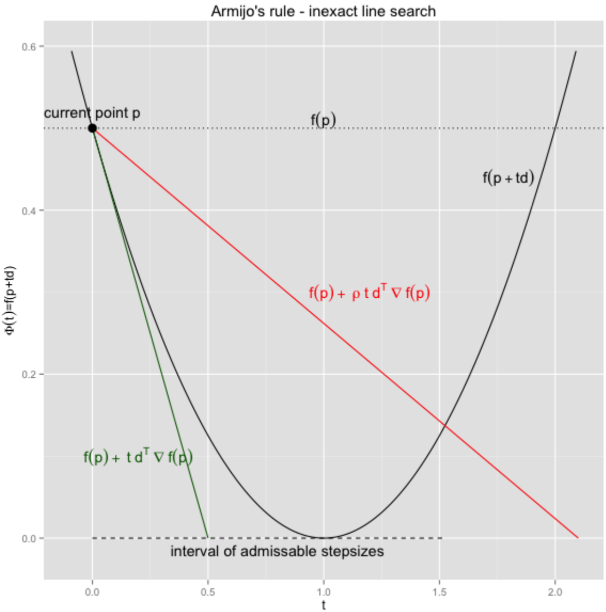
The sign of the scalar product between two vectors a and b is determined by the cosine of the angle α between a and b:

$$\langle a, b \rangle = a^T b = \|a\| \|b\| \cos \alpha(a, b)$$

d is a descent direction if and only if:

$$d^T \nabla f(x) < 0$$

Stepsize


Armijo's rule - inexact line search
current point p
f(p)
f(p + td)
f(p) + ρ t d^T ∇ f(p)
f(p) + t d^T ∇ f(p)
interval of admissable stepsizes
Φ(t)=f(p+td)
t
0.0
0.2
0.4
0.6
0.0
0.5
1.0
1.5
2.0

Stepsize

Armijo's rule

Takes two parameters $0 < \sigma < 1$, and $0 < \rho < 0.5$

For $\ell = 0, 1, 2, ...$ test *Armijo condition*:

$$f(p + \sigma^\ell d) < f(p) + \rho\sigma^\ell d^T \nabla f(p)$$

Accepted stepsize

First ℓ that passes this test determines the accepted stepsize

$$t = \sigma^\ell$$

Standard Armijo implies, that for the accepted stepsize t always holds $t <= 1$, only *semi-efficient*.

Technical detail: Widening

Test whether some $t > 0$ satisfy Armijo condition, i.e. check $\ell = -1, -2, \ldots$ as well, ensures *efficiency*.

Gradient Descent

Descent direction

Direction of *Steepest Descent*, the negative gradient:

$$d = -\nabla f(x)$$

Motivation:

- corresponds to $\alpha = 180^\circ = \pi$
- obvious choice, always a descent direction, no test needed
- guarantees the quickest win locally
- works with inexact line search, e.g. Armijo' s rule
- works for functions $f \in \mathcal{C}^1$
- always solves auxiliary optimization problem

$$\min \; s^T \nabla f(x), \quad s \in \mathbb{R}^n, \; \|s\| = 1$$

Conjugate Gradient

Motivation: Quadratic Model Problem, minimize

$$f(x) = \|Ax - b\|^2$$

Optimality condition:

$$\nabla f(x^*) = 2A^T (Ax^* - b) = 0$$

Obvious approach: Solve system of linear equations

$$A^T Ax = A^T b$$

Descent direction

Consecutive directions $d_i, \dots, d_{i+k}$ satisfy certain orthogonality or *conjugacy* conditions, $M = A^T A$ symmetric positive definite:

$$d_i^T M d_j = 0, \ i \neq j$$

Nonlinear Conjugate Gradient

Initial Steps:

- start at point x_0 with $d_0 = -\nabla f(x_0)$
- perform exact line search, find

$$t_0 = \arg\min f(x_0 + td_0), \ \ t > 0$$

- set $x_1 = x_0 + t_0 d_0$.

Iteration:

- set $\Delta_k = -\nabla f(x_k)$
- compute β_k via one of the available formulas (next slide)
- update conjugate search direction $d_k = \Delta_k + \beta_k d_{k-1}$
- perform exact line search, find

$$t_k = \arg\min f(x_k + td_k), \ \ t > 0$$

- set $x_{k+1} = x_k + t_k d_k$

Nonlinear Conjugate Gradient

Formulas for β_k:

Fletcher-Reeves
$$\beta_k^{FR} = \frac{\Delta_k^T \Delta_k}{\Delta_{k-1}^T \Delta_{k-1}}$$

Polak-Ribière
$$\beta_k^{PR} = \frac{\Delta_k^T \left(\Delta_k - \Delta_{k-1}\right)}{\Delta_{k-1}^T \Delta_{k-1}}$$

Hestenes-Stiefel
$$\beta_k^{HS} = -\frac{\Delta_k^T \left(\Delta_k - \Delta_{k-1}\right)}{s_{k-1}^T \left(\Delta_k - \Delta_{k-1}\right)}$$

Dai-Yuan
$$\beta_k^{DY} = -\frac{\Delta_k^T \Delta_k}{s_{k-1}^T \left(\Delta_k - \Delta_{k-1}\right)}$$

Reasonable choice with automatic direction reset:

$$\beta = \max\left\{0, \beta^{PR}\right\}$$