Pattern Recognition (PR)

Prof. Dr.-Ing. Andreas Maier
Pattern Recognition Lab (CS 5), Friedrich-Alexander-Universität Erlangen-Nürnberg
Winter Term 2020/21

This is a printable version of the slides of the lecture

Pattern Recognition (PR)
Winter term 2020/21
Friedrich-Alexander University of Erlangen-Nuremberg.

Erlangen, January 8, 2021
Prof. Dr.-Ing. Andreas Maier

Optimization

Motivation

- Optimization is crucial for many solutions in pattern recognition, pattern analysis, machine learning, artificial intelligence, etc.
- Optimization has many faces:
  - discrete optimization,
  - combinatorial optimization,
  - genetic algorithms,
  - gradient descent,
  - unconstrained and constrained optimization,
  - linear programming,
  - convex optimization, etc.
- There is no lecture on pattern recognition without a refresher course on optimization techniques.
- Each researcher has his own favorite optimization algorithm.

Convexity

Definition

A function $f : \mathbb{R}^d \to \mathbb{R}$ is *convex* if the domain $\text{dom}(f)$ of f is a convex set and if $\forall \boldsymbol{x}, \boldsymbol{y} \in \text{dom}(f)$, and θ with $0 \leq \theta \leq 1$, we have

$$f(\theta \boldsymbol{x} + (1-\theta)\boldsymbol{y}) \leq \theta f(\boldsymbol{x}) + (1-\theta) f(\boldsymbol{y})$$

A function $f : \mathbb{R}^d \to \mathbb{R}$ is *concave* if $-f$ is convex.

Geometric interpretation:

The line segment between $(\boldsymbol{x}, f(\boldsymbol{x}))$ and $(\boldsymbol{y}, f(\boldsymbol{y}))$ lies above the graph of f.

Unconstrained Optimization

Let us assume in the following that we have to compute the minimum of a convex function

$$f : \mathbb{R}^d \to \mathbb{R}$$

that is twice differentiable.

The unconstrained optimization problem is just the solution of the minimization problem

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\text{argmin}}\, f(\boldsymbol{x})$$

where $\boldsymbol{x}^*$ denotes the optimal point.

Unconstrained Optimization (cont.)

For this particular family of functions, a necessary and sufficient condition for the minumum are the zero-crossings of the function's gradient:

$$\nabla f(\boldsymbol{x}^*) = 0.$$

Unconstrained Optimization (cont.)

Most methods follow an iterative scheme:

$$\begin{array}{rl} \text{initialization} & \boldsymbol{x}^{(0)} \\ \text{iteration step} & \boldsymbol{x}^{(k+1)} = g(\boldsymbol{x}^{(k)}) \end{array}$$

where $g : \mathbb{R}^d \rightarrow \mathbb{R}^d$ is the update function.

The iterations terminate, if

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\| < \varepsilon,$$

i. e. no further significant change.

Descent Methods

We now consider iteration schemes that produce a sequence of estimates according to the update function

$$\boldsymbol{x}^{(k+1)} = g(\boldsymbol{x}^{(k)}) = \boldsymbol{x}^{(k)} + t^{(k)} \Delta \boldsymbol{x}^{(k)}.$$

where

$\Delta \boldsymbol{x}^{(k)} \in \mathbb{R}^d$:	is the search direction in the k-th iteration
$t^{(k)} \in \mathbb{R}$:	denotes the step length in the k-th iteration

and where we expect

$$f(\boldsymbol{x}^{(k+1)}) < f(\boldsymbol{x}^{(k)}), \quad \text{i.e.} \ \nabla f(\boldsymbol{x}^{(k)})^T \Delta \boldsymbol{x}^{(k)} < 0$$

except $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} = \boldsymbol{x}^*$.

Taylor Approximation

For many problems it is always good to know the second order Taylor approximation:

$$f(\boldsymbol{x} + t \cdot \Delta \boldsymbol{x}) \approx f(\boldsymbol{x}) + t \cdot \nabla f(\boldsymbol{x})^T \Delta \boldsymbol{x} + \frac{1}{2} t^2 \cdot \Delta \boldsymbol{x}^T \nabla^2 f(\boldsymbol{x}) \Delta \boldsymbol{x}$$

Descent Methods (cont.)

Input: function f, initial estimate $\boldsymbol{x}^{(0)}$
Initialize: $k := 0$
repeat
Select (or compute) descent direction
Line search (1-D optimization):

$$t^{(k)} = \underset{t \geq 0}{\operatorname{argmin}} f(\boldsymbol{x}^{(k)} + t \cdot \Delta \boldsymbol{x}^{(k)})$$

Update:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + t^{(k)} \Delta \boldsymbol{x}^{(k)}.$$

$k := k + 1$
until $\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\| < \varepsilon$
Output: $\boldsymbol{x}^{(k)}$

Line Search Methods

- Multivariate optimization in its described form requires a proper line search method.
- Exact line search along the straight line $\{\boldsymbol{x} + t\Delta\boldsymbol{x} \mid t \geq 0\}$ has to solve

$$t^* = \underset{t \geq 0}{\operatorname{argmin}} f(\boldsymbol{x} + t\Delta\boldsymbol{x})$$

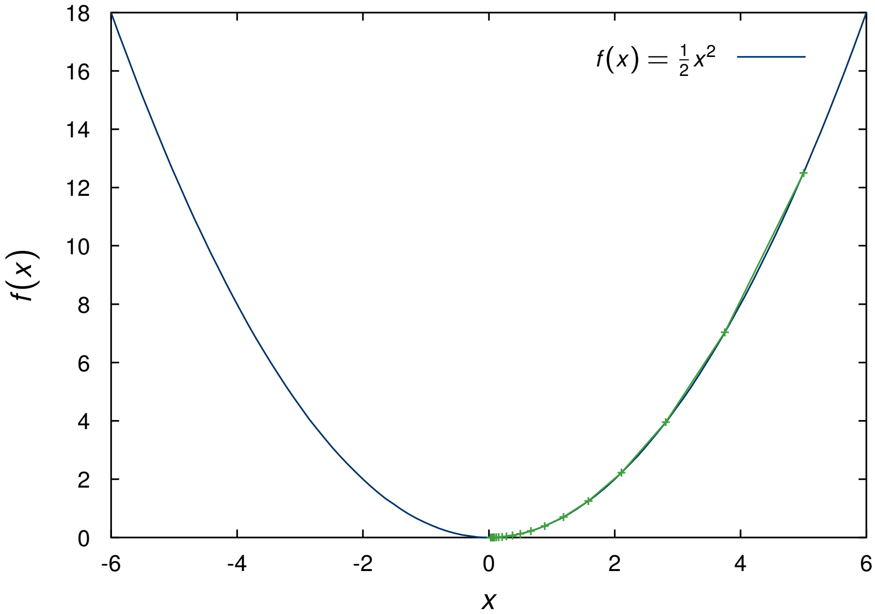
  and is rarely used.
- An overview of methods can be found in numerical recipes.

Pattern Recognition Lab

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG FACULTY OF ENGINEERING

Line Search Methods (cont.)

Setting $t = 0.25$:

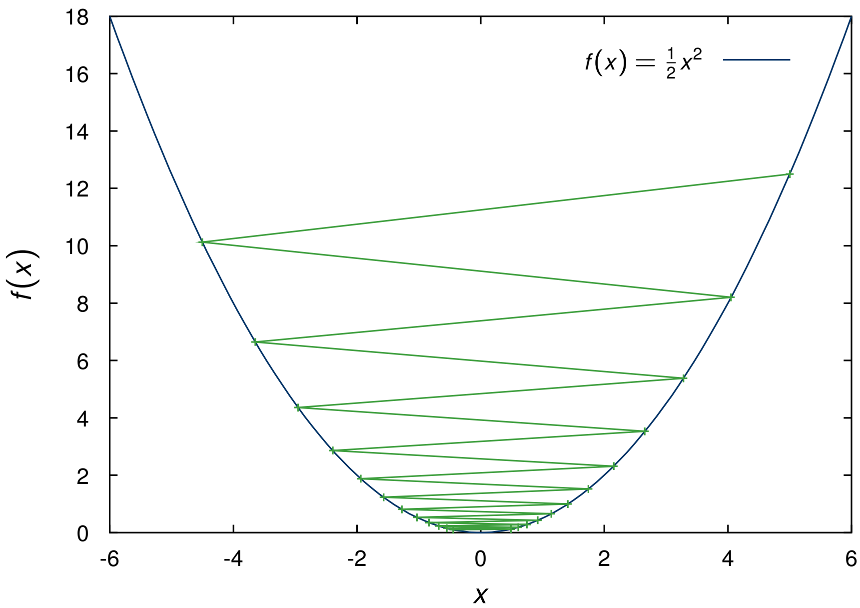


Pattern Recognition Lab

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG FACULTY OF ENGINEERING

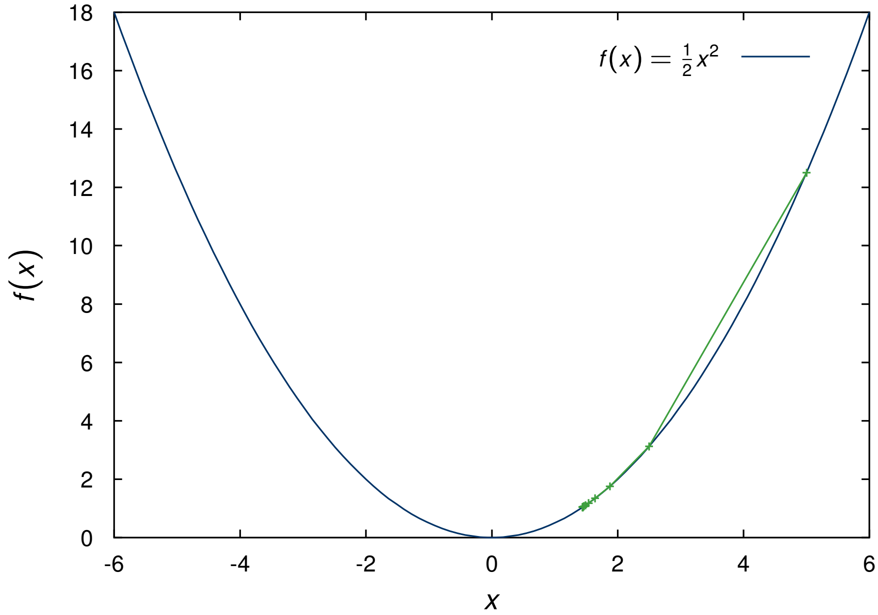
Line Search Methods (cont.)

Setting $t = 1.9$:



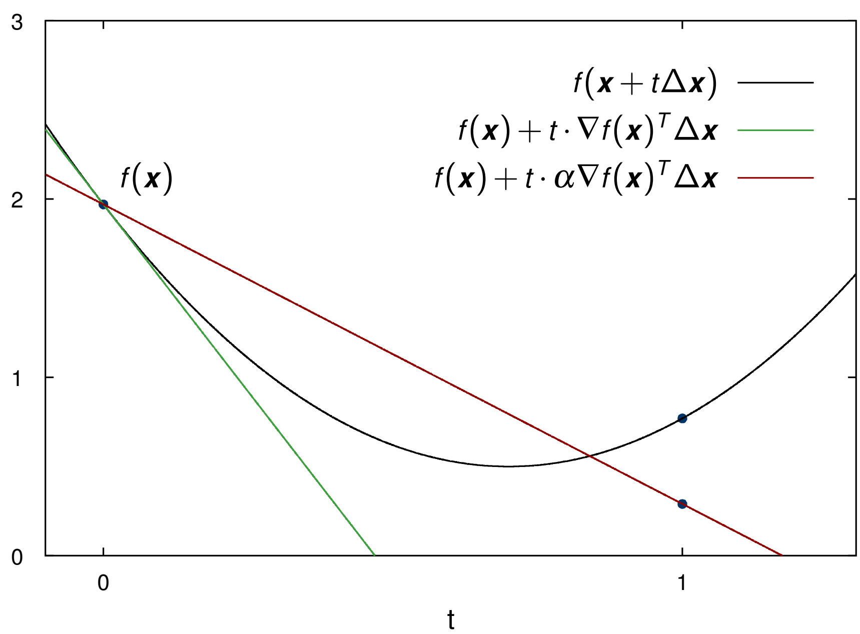
Line Search Methods (cont.)

Setting $t^{(k+1)} = \frac{1}{2} t^{(k)}$ and starting with $t^{(0)} = 0.5$:



Pattern Recognition Lab
FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
FACULTY OF ENGINEERING

Backtracking Line Search



Backtracking Line Search (cont.)

The Armijo-Goldstein line search algorithm:

Input: function f, search direction $\Delta \boldsymbol{x}$
Initialize: $t := 1$
Select: $\alpha \in [0, 0.5]$ and $\beta \in [0, 1]$.
while $f(\boldsymbol{x} + t\Delta\boldsymbol{x}) > f(\boldsymbol{x}) + \alpha t \cdot \nabla f(\boldsymbol{x})^T \Delta \boldsymbol{x}$ **do**
  $t := \beta t$
end while
Output: t

Gradient Descent Methods

A natural choice of the search direction is the negative gradient:

$$\Delta \boldsymbol{x}^{(k)} = -\nabla f(\boldsymbol{x}^{(k)})$$

Rule of thumb:

The negative gradient is the steepest descent direction.

Gradient Descent Methods (cont.)

Input: function f, initial estimate $\boldsymbol{x}^{(0)}$
intialize: $k := 0$
repeat
Set descent direction: $\Delta \boldsymbol{x}^{(k)} = -\nabla f(\boldsymbol{x}^{(k)})$
Line search (1-D optimization):

$$t^{(k)} = \operatorname*{argmin}_{t \geq 0} f(\boldsymbol{x}^{(k)} + t \cdot \Delta \boldsymbol{x}^{(k)})$$

Update:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + t^{(k)} \Delta \boldsymbol{x}^{(k)}.$$

$k := k + 1$
until $\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\|_2 < \varepsilon$
Output: $\boldsymbol{x}^{(k)}$

Pattern Recognition Lab

FAU FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Next Time in

Pattern Recognition

Steepest Descent Methods

(Normalized) steepest descent, what does it mean?

We search for the unit vector that shows the largest decrease in the linear approximation of f:

$$\Delta \boldsymbol{x} = \underset{\boldsymbol{u}}{\operatorname{argmin}} \{\nabla f(\boldsymbol{x})^T \boldsymbol{u};\ \|\boldsymbol{u}\|_p = 1\}$$

Conclusions:

- The steepest descent direction depends on the chosen norm.
- The negative gradient is not necessarily the best choice for the search direction.

Steepest Descent Methods (cont.)

We consider now the first order Taylor approximation of $f(\boldsymbol{x} + \boldsymbol{u})$ around the selected position $\boldsymbol{x}$:

$$f(\boldsymbol{x} + \boldsymbol{u}) \approx f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^T \boldsymbol{u}.$$

- Here $\nabla f(\boldsymbol{x})^T \boldsymbol{u}$ is the directional derivative at $\boldsymbol{x}$ in direction $\boldsymbol{u}$.
- The vector $\boldsymbol{u}$ denotes a descent direction if the inner product with the gradient vetor is negative, i. e.

$$\nabla f(\boldsymbol{x})^T \boldsymbol{u} < 0\ .$$

Steepest Descent Methods (cont.)

Input: function f, initial estimate $\boldsymbol{x}^{(0)}$, norm $\|.\|$
intialize: $k := 0$
repeat

Compute highest descent direction:

$$\Delta \boldsymbol{x}^{(k)} = \underset{\boldsymbol{u}}{\operatorname{argmin}} \{ \nabla f(\boldsymbol{x}^{(k)})^T \boldsymbol{u};\ \|\boldsymbol{u}\| = 1 \}$$

Line search (1-D optimization):

$$t^{(k)} = \underset{t \geq 0}{\operatorname{argmin}}\, f(\boldsymbol{x}^{(k)} + t \cdot \Delta \boldsymbol{x}^{(k)})$$

Update:

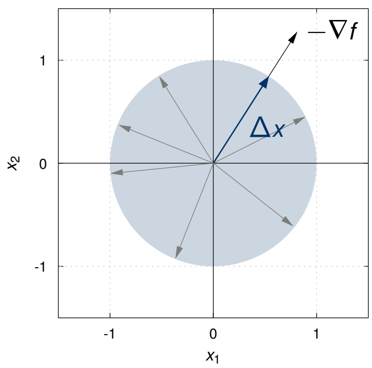
$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + t^{(k)} \Delta \boldsymbol{x}^{(k)}.$$

$k := k + 1$
until $\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\| < \varepsilon$
Output: $\boldsymbol{x}^{(k)}$

Pattern Recognition Lab
FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
FACULTY OF ENGINEERING

L_2-Norm

The unit ball for the L_2-norm:

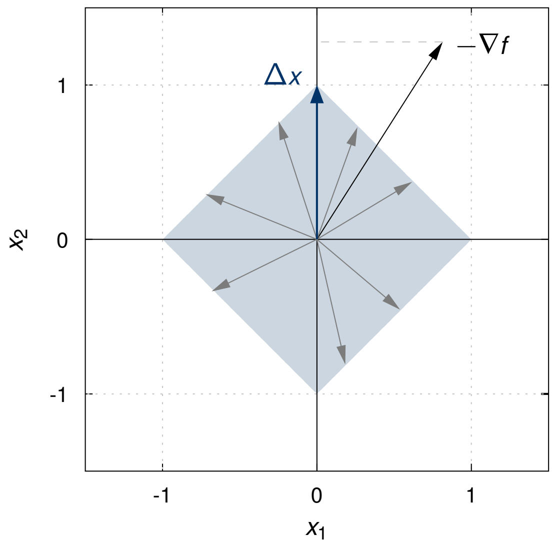


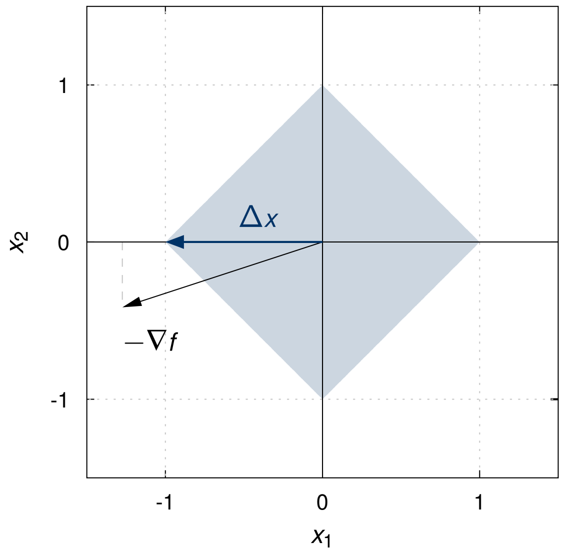
For the L_2-norm the steepest descent direction is the negative gradient:

$$\Delta \boldsymbol{x} = \underset{\boldsymbol{u}}{\operatorname{argmin}} \{ \nabla f(\boldsymbol{x})^T \boldsymbol{u};\ \|\boldsymbol{u}\|_2 = 1 \} = -\nabla f(\boldsymbol{x})$$

L_1-Norm

The unit ball for the L_1-norm:





L_1-Norm (cont.)

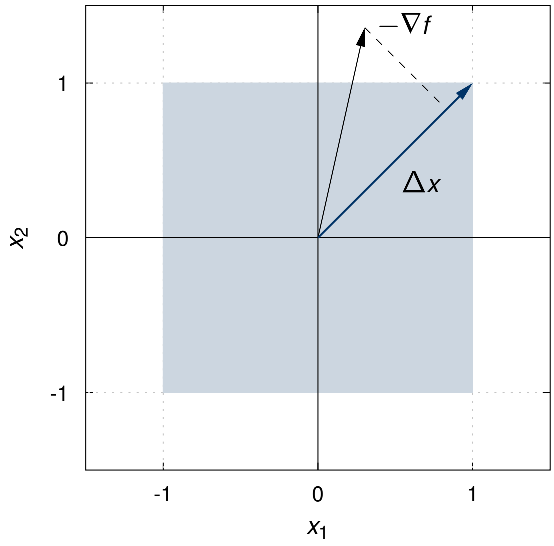
- The steepest descent for the L_1-norm selects in each iteration the component of $\nabla f(\boldsymbol{x})$ with maximum absolute value and then decreases or increases dependent on the sign of the selected component.
- Let i be the index of the gradient component with maximum absolute value, and let $\boldsymbol{e}_i \in \mathbb{R}^d$ denote the corresponding base vector.
  The steepest descent direction is given by:

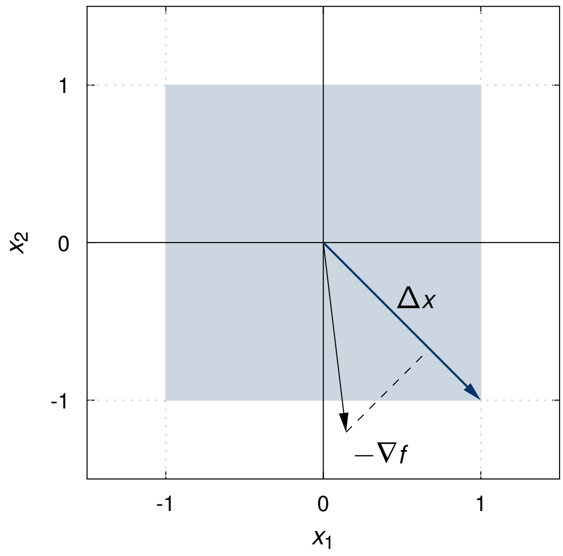
$$\begin{aligned}\Delta \boldsymbol{x} &= \operatorname*{argmin}_{\boldsymbol{u}} \{\nabla f(\boldsymbol{x})^T \boldsymbol{u};\ \|\boldsymbol{u}\|_1 = 1\} \\ &= -\operatorname{sgn}\left(\frac{\partial}{\partial x_i} f(\boldsymbol{x})\right) \boldsymbol{e}_i\end{aligned}$$

- Note: Steepest descent using the L_1-norm results in the *coordinate descent algorithm*.

L_∞-Norm

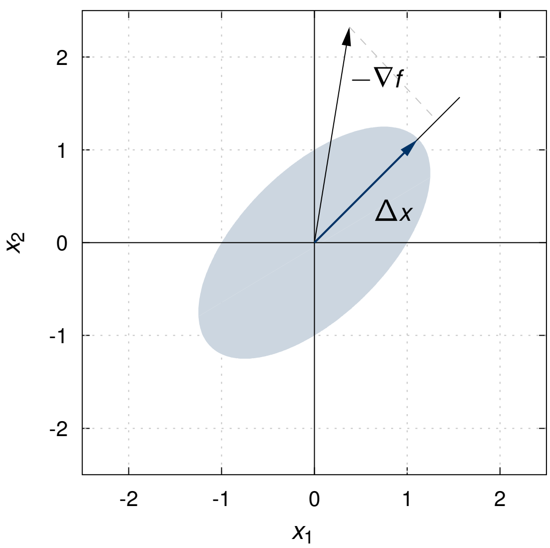
The unit ball for the L_∞-norm:





L_P-Norm

The unit ball for the L_P-norm:



L_P-Norm (cont.)

The steepest descent for the L_P-norm is given by:

$$\begin{aligned} \Delta \boldsymbol{x} &= \underset{\boldsymbol{u}}{\operatorname{argmin}} \{ \nabla f(\boldsymbol{x})^T \boldsymbol{u}; \ \|\boldsymbol{u}\|_{\boldsymbol{P}} = 1 \} \\ &= \underset{\boldsymbol{u}}{\operatorname{argmin}} \{ \nabla f(\boldsymbol{x})^T \boldsymbol{u}; \ (\boldsymbol{u}^T \boldsymbol{P} \boldsymbol{u})^{\frac{1}{2}} = 1 \} \\ &= \underset{\boldsymbol{u}}{\operatorname{argmin}} \{ \nabla f(\boldsymbol{x})^T \boldsymbol{u}; \ \|\boldsymbol{P}^{\frac{1}{2}} \boldsymbol{u}\|_2 = 1 \} \end{aligned}$$

As we did in the LDA-transform, we introduce a transform to get spherical data:

$$\boldsymbol{u}' = \boldsymbol{P}^{\frac{1}{2}} \boldsymbol{u}$$

and thus

$$f(\boldsymbol{u}) = f(\boldsymbol{P}^{-\frac{1}{2}} \boldsymbol{u}') = f'(\boldsymbol{u}')$$

L_P-Norm (cont.)

Instead of $f(\boldsymbol{x})$ we now minimize $f'(\boldsymbol{x}')$ using the L_2-norm and back-transform the result:

$$\begin{aligned} \Delta \boldsymbol{x}' &= \underset{\boldsymbol{u}}{\operatorname{argmin}} \{ \nabla f'(\boldsymbol{x}')^T \boldsymbol{u}'; \ \|\boldsymbol{u}'\|_2 = 1 \} \\ &= -\nabla f'(\boldsymbol{x}') \\ &= -\boldsymbol{P}^{-\frac{1}{2}} \nabla f(\boldsymbol{P}^{-\frac{1}{2}} \boldsymbol{x}') \\ &= -\boldsymbol{P}^{-\frac{1}{2}} \nabla f(\boldsymbol{x}) \end{aligned}$$

L_P-Norm (cont.)

Now we get for $\Delta \boldsymbol{x}$:

$$\begin{aligned}\Delta \boldsymbol{x} &= \boldsymbol{P}^{-\frac{1}{2}} \Delta \boldsymbol{x}' \\ &= \boldsymbol{P}^{-\frac{1}{2}} \left(-\boldsymbol{P}^{-\frac{1}{2}} \nabla f(\boldsymbol{x}) \right) \\ &= -\boldsymbol{P}^{-1} \nabla f(\boldsymbol{x}).\end{aligned}$$

Conclusion: The steepest descent for the L_P-norm is given by

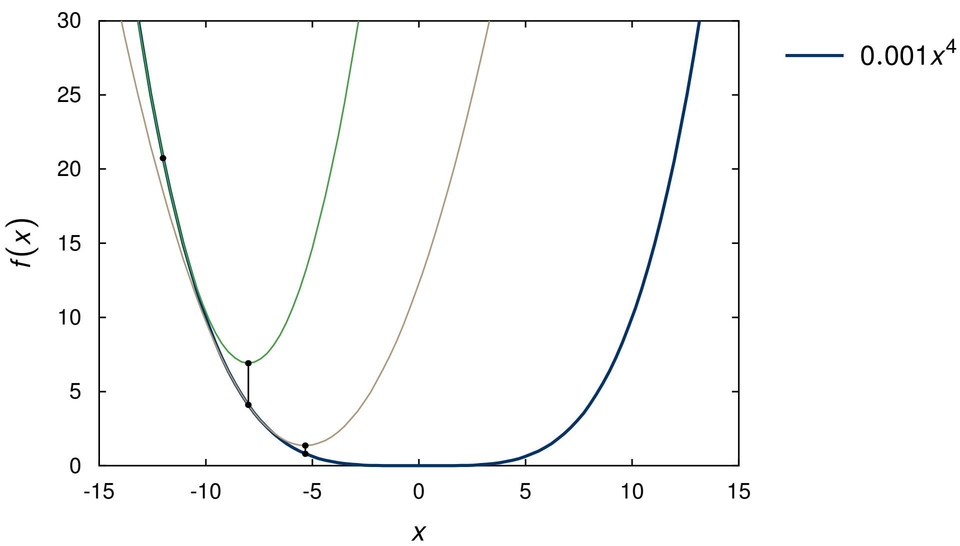
$$\Delta \boldsymbol{x} = -\boldsymbol{P}^{-1} \nabla f(\boldsymbol{x}) \; .$$

Pattern Recognition Lab
FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG FACULTY OF ENGINEERING

Newton's Method

The idea:

- Select a point.
- Compute the minimum of the second order Taylor approximation.



Newton's Method (cont.)

Second order Taylor approximation:

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

Now we select $\Delta\boldsymbol{x}$ such that

$$\nabla\{f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^T \Delta\boldsymbol{x} + \frac{1}{2}\Delta\boldsymbol{x}^T(\nabla^2 f(\boldsymbol{x}))\Delta\boldsymbol{x}\} = 0$$

Obviously the gradient is

$$\nabla f(\boldsymbol{x}) + \nabla^2 f(\boldsymbol{x})\Delta\boldsymbol{x} = 0$$

and thus

$$\Delta\boldsymbol{x} = -(\nabla^2 f(\boldsymbol{x}))^{-1}\nabla f(\boldsymbol{x})$$

Newton's Method (cont.)

Conclusion:

Newton's method is an $\boldsymbol{x}$-dependent steepest descent method regarding the $L_{\boldsymbol{P}}$-norm, where $\boldsymbol{P} = \nabla^2 f(\boldsymbol{x})$ is the Hessian.

Damped Newton's Method

Input: function f, initial estimate $\boldsymbol{x}^{(0)}$
intialize: $k := 0$
repeat
Compute Newton step:

$$\Delta \boldsymbol{x}^{(k)} = -\nabla^2 f(\boldsymbol{x}^{(k)})^{-1} \nabla f(\boldsymbol{x}^{(k)})$$

Line search (1-D optimization):

$$t^{(k)} = \operatorname*{argmin}_{t \geq 0} f(\boldsymbol{x}^{(k)} + t \cdot \Delta \boldsymbol{x}^{(k)})$$

Update:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + t^{(k)} \Delta \boldsymbol{x}^{(k)}.$$

$k := k + 1$
until $\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\| < \varepsilon$
Output: $\boldsymbol{x}^{(k)}$

Lessons Learned

- Gradient descent is widely applied.
- Gradient descent and coordinate descent are special cases of steepest descent methods.
- Steepest descent method depends on the chosen norm.

Pattern Recognition Lab

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Next Time in Pattern Recognition

Further Readings

This chapter is basically copied from:

- S. Boyd, L. Vandenberghe:
  Convex Optimization,
  Cambridge University Press, 2004.
  ☞ `http://www.stanford.edu/~boyd/cvxbook/`
- Jorge Nocedal, Stephen Wright:
  Numerical Optimization,
  Springer, New York, 1999.

Comprehensive Questions

- What is the general formulation for an unconstrained optimization problem?
- Why do we need a line search in gradient descent approaches?
- What is the Armijo-Goldstein line search algorithm?
- What are the steepest descent directions if we apply the L_∞, L_1, L_2, and $L_{\boldsymbol{P}}$ norm?