Optimization

Chapter 1
Introduction

General Information

Myself

- PhD: ETH Zurich (2002)
- Afterwards: Microsoft Research, ETH Zurich, MIT, U. Lugano (CH)
- In Freiburg since April 2012 (Chair of Algorithms and Complexity)

Lecture

- Optimization: Find (almost) best solution for some problem
- Appears in a lot of places throughout computer science
  - Operations research
  - Computational science
  - Machine learning
  - Computer graphics and computer vision
  - Robotics
  - Theory
  - etc.
- Lecture prepared jointly with Thomas Brox (computer vision)

Organizational Issues

Lecture & Exercise Tutorials

- Thursday 14:15-16:00, room 101-00-026
- Exercise tutorials roughly every third week
- Will plan exact handling of exercises/tutorials throughout the semester

Course web page

- http://ac.informatik.uni-freiburg.de/teaching/ss_13/optimization.php
- Will contain all important additional information

Lecture material

- Slides and recordings will be available online

  Note: Recordings are not guaranteed!
  - We will try to always provide them…

- Some slides / lecture material will be in English

Organizational Issues

Assignments

- Mix of practical and theoretical exercises
- We use Matlab/Octave for practical exercises
  - Short introduction in second half today…
- Help get a better understanding of the most important concepts
- Invest time into the assignments!
  - They're one of the best ways to prepare for the exam!
- Refer to course webpage for dates

Exam

- Written exam
- Make sure you are registered for the exam in time

General

- Please, be active!
  - Ask questions
  - Work through the slides after the lecture and make sure you got all the key insights, ideally together with other students

Optimization in Operations Research

How should we optimally

- plan some complex productions process?
  - fastest production
  - minimum cost
- arrange components on a computer chip?
  - fastest production
  - minimize chip‘s total area / energy consumption / clock rate / ...
- Organize supply chains
  - Managing upstream and down stream value added flow of materials, final goods, and related information
- ...

Optimization in Computational Science

Example: Simulation of complex mechanical systems

- Biomechanics of the human spine [Dickopf, Steiner, Krause '08]
  - Application: treatment of spinal stenosis resulting from tissue degeneration
  - Common therapy: place a spacer between the spinous processes
- Each simulation step requires solving some optimization problem

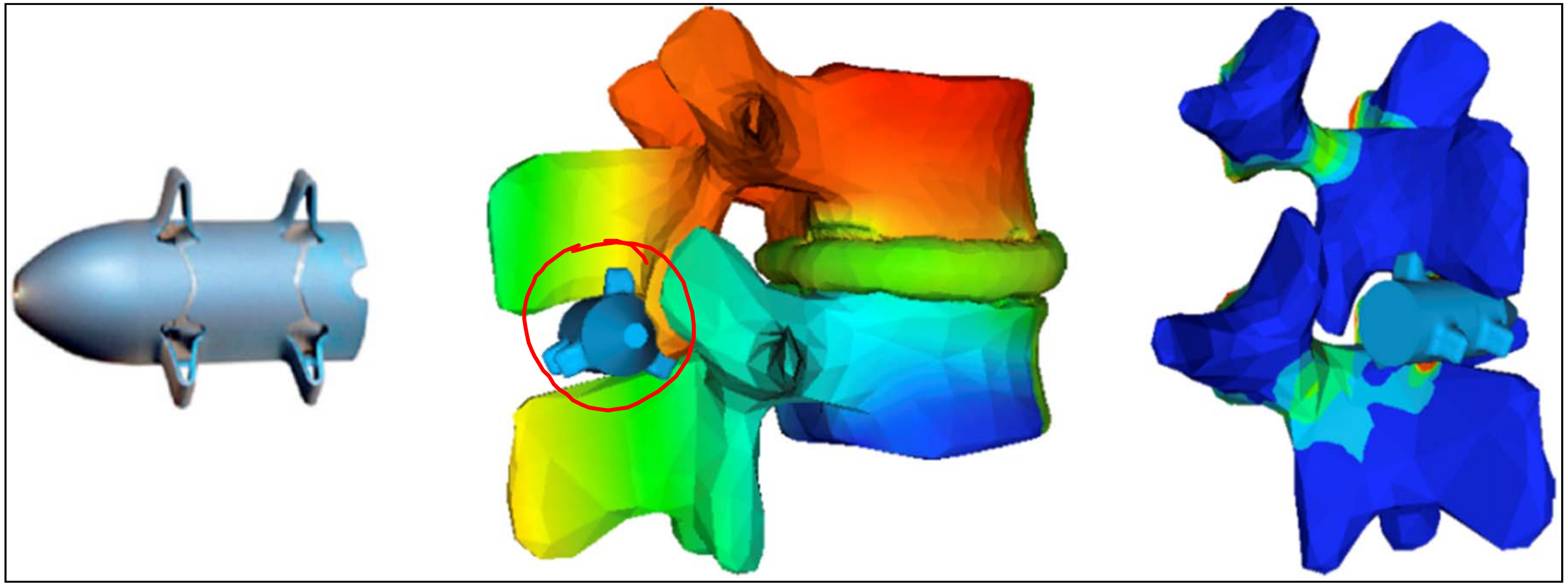


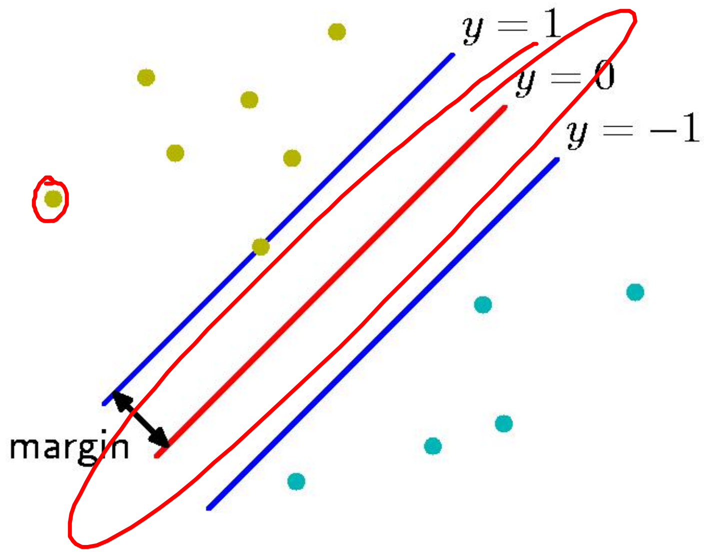
Figure: Spacer APERIUS PercLID System (left); deformed vertebral segment with stresses invertical direction (center); zoom to the processes with von Mises stresses (right) (left figure from [Kyphon Inc., Sunnyvale, CA, USA])

What is the Purpose of Optimization?

- Computer science is all about **algorithms**
- The choice of an algorithm depends much on the **problem**
  → It is important to see clear what the problem is
- Extended concept in computer science:
  - Have a formal description of your problem
    (**model**, **cost function**)
  - Apply algorithms that can solve those sorts of problems
    (**optimization methods**)
- Advantages:
  - The problem is per definition well defined and can be analyzed
    (e.g. is there a (single best) solution to this problem?)
  - Algorithms can be developed for a certain class of problems (and reused)
  - Problems can be formulated such that efficient algorithms are available to
    solve them

Example From Machine Learning

- Given: samples $\mathbf{x}_n$ of two different classes $t_n \in \{-1, 1\}$
- Goal: find a linear decision boundary
  $y = \mathbf{w}^\top \mathbf{x} + b$
  that allows to classify a new sample $\mathbf{x}$



1. Formulate the problem by a cost function:

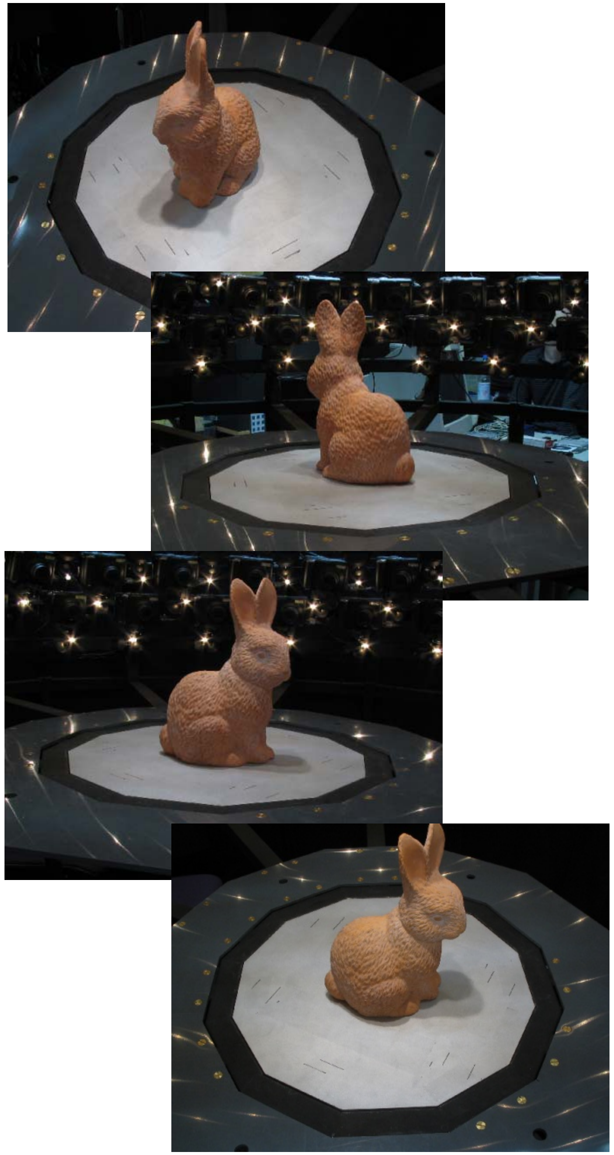
   $$\text{argmin}_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2$$ (small weights)

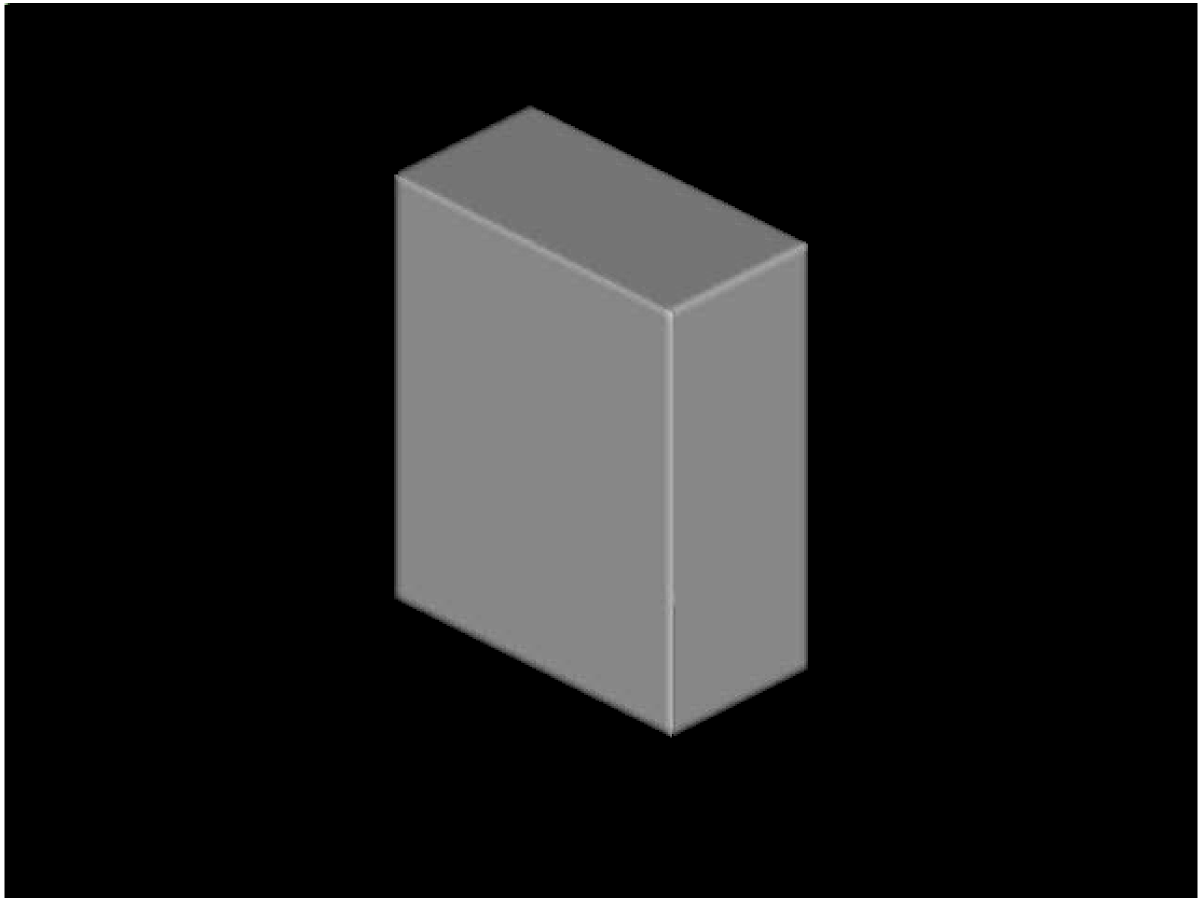
   with constraints

   $$t_n(\mathbf{w}^\top \mathbf{x}_n + b) \geq 1$$ (all given samples must be on the correct side)

2. Constrained optimization problem: **quadratic program**
   → use standard algorithms to solve this problem

Example From Computer Vision



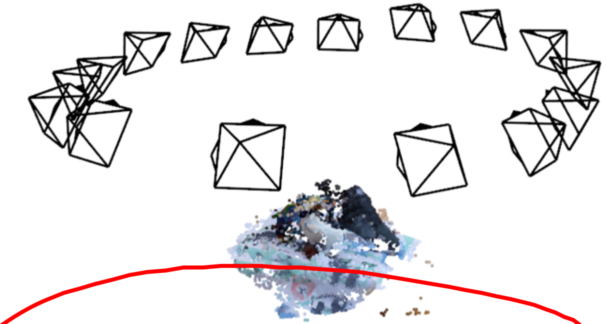


Find the surface that best explains all input images for given camera positions

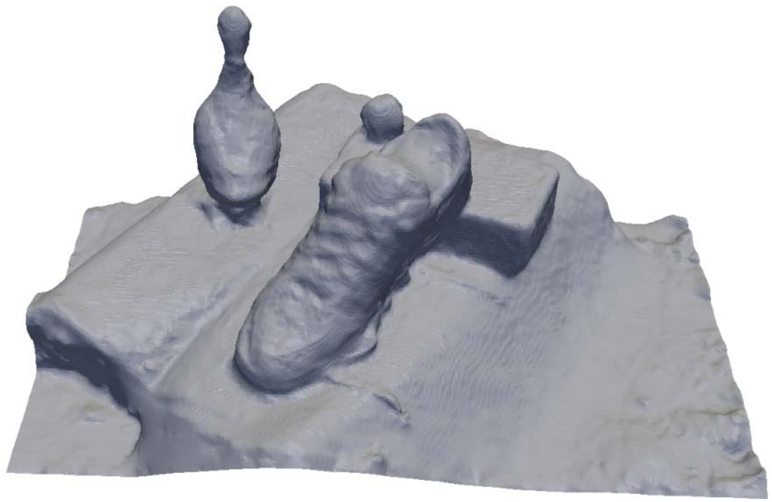
Combination of Bundle Adjustment and Dense Reconstruction

Author: Benjamin Ummenhofer

Image from a video



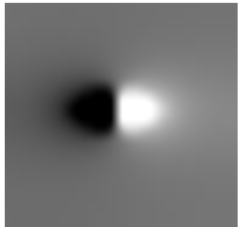
Result from sparse bundle adjustment



Dense surface reconstruction

Point constraints from bundle adjustment

- Initial bundle adjustment provides camera parameters and a point cloud
- Intuition: the surface should pass through the given 3D points and be as small as possible



- Region term → non-trivial global optimum
- Each 3D point yields a constraint for the interior and exterior of the object that can be mollified and aggregated over all 3D points → $R_1(\mathbf{X})$
- Together with minimization of the surface:

$$E(u) = \int R_1 u + \nu |\nabla u| \, d\mathbf{X} \quad u(\mathbf{X}) \in [0, 1]$$

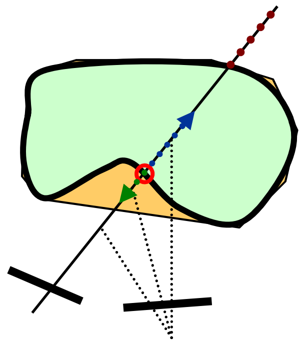
Convex problem → global optimum (see later)

Author: Benjamin Ummenhofer

Including Photoconsistency

- Once we have an initial surface (level set of u), we can compute the visibility and the photoconsistency $\rho(\mathbf{X})$

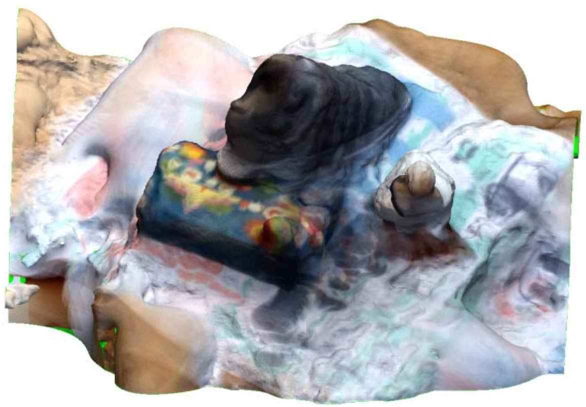
- We can also compute another region term $R_2(\mathbf{X})$ based on the photoconsistency (Kolev et al. 2009)



- Energy:

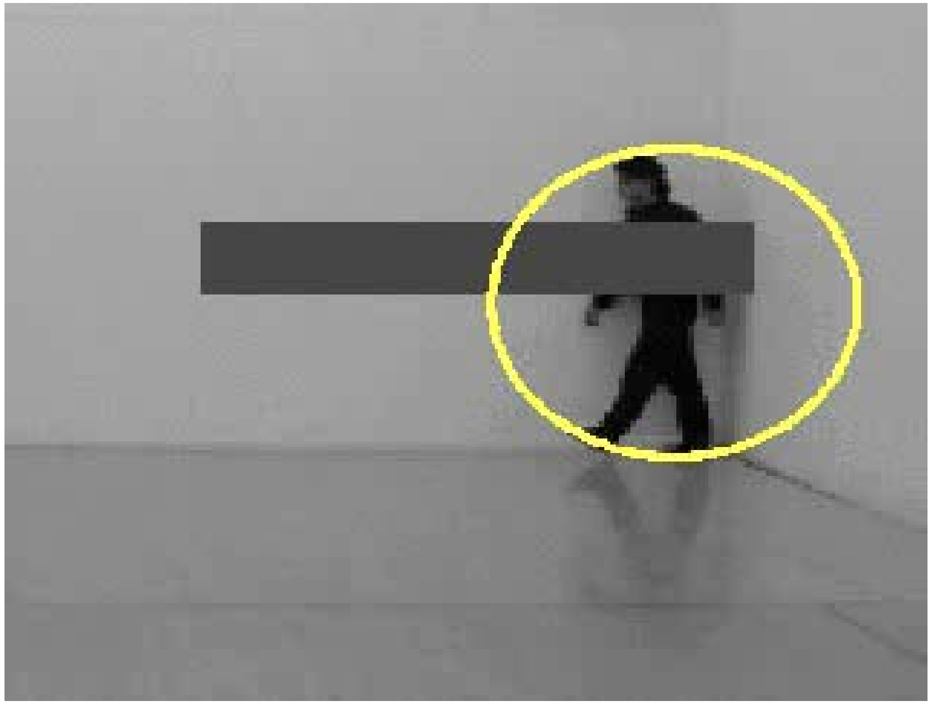
$$E(u) = \int (R_1 + R_2) u + \nu \rho |\nabla u| \, d\mathbf{X} \qquad u(\mathbf{X}) \in [0,1]$$

- For fixed $R_2(\mathbf{X}), \rho(\mathbf{X})$ this is still a convex problem
→ iteratively update the surface $u(\mathbf{X})$ and $R_2(\mathbf{X}), \rho(\mathbf{X})$



Region-Based Active Contours

Find the contour that best separates foreground and background



Author: Daniel Cremers

Image Segmentation: Region-Based Active Contours

- Let us restrict the Mumford-Shah functional to a two-region cartoon model.

- The energy states the optimal separation of pixel intensities:

$$E(C) = \int_{\Omega_1} (I - \mu_1)^2 \mathbf{dx} + \int_{\Omega_2} (I - \mu_2)^2 \mathbf{dx} + \nu |C|$$

- This is similar to k-means clustering (two-means), but with an additional constraint on the length of the separating contour.

- We can express this using a level set representation of the contour: (Chan-Vese 2001)

$$E(\phi) = \int \Big(H(\phi)(I - \mu_1)^2 + (1 - H(\phi))(I - \mu_2)^2 + \nu |\nabla H(\phi)| \Big) \, \mathbf{dx}$$

- Given ϕ, μ_1 and μ_2 can be found analytically as the mean intensities inside the two regions

$$\mu_1 = \frac{\int H(\phi) I \, \mathbf{dx}}{\int H(\phi) \, \mathbf{dx}} \qquad \mu_2 = \frac{\int (1 - H(\phi)) I \, \mathbf{dx}}{\int (1 - H(\phi)) \, \mathbf{dx}}$$

Motion Estimation

- Given a sequence of images $I(x,y,t)$, what is the motion of each pixel between subsequent frames?

- Formally, we seek a vector field $(u,v)(x,y,t)$ that transforms the second image into the first one.

Hamburg taxi sequence

- This vector field is called **optical flow**. It individually moves each point (x,y) at time t such that it fits the point at time $t+1$.

- We can express this task by the following minimization problem:

$$E(u,v) = \int_\Omega (I(x+u, y+v, t+1) - I(x,y,t))^2 dxdy \to \min$$

Zielfunktion und zulässige Menge

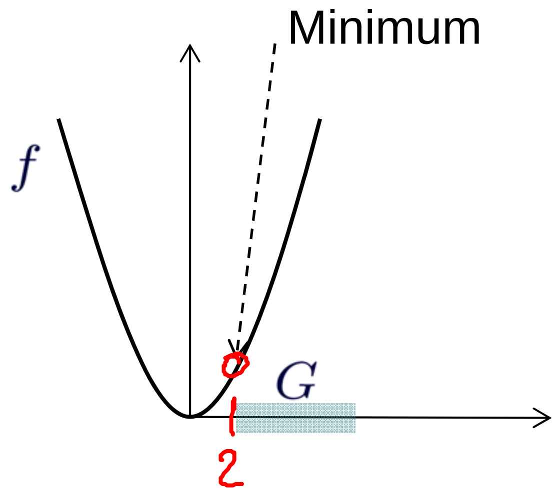
- Ein Optimierungsproblem besteht aus einer **zulässigen Menge** G und einer **Zielfunktion** $f : G \to \mathbb{R}$

- Beispiel:

  $f(x) = x^2$

  Nebenbed.

  $G = \{x \in \mathbb{R} | 2 \leq x \leq 5\}$

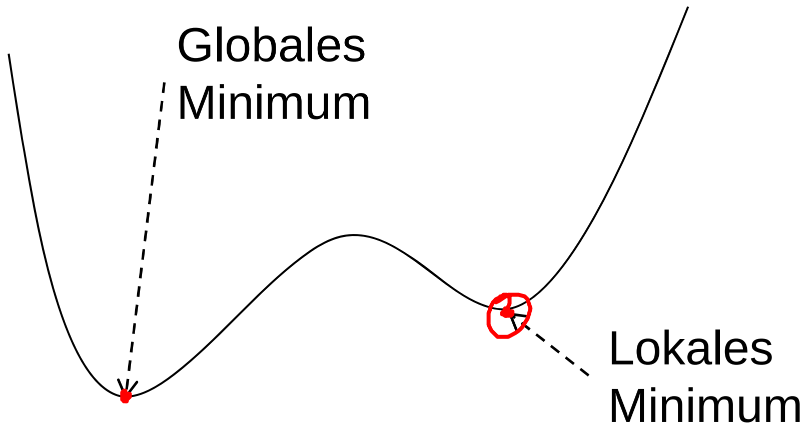


- **Minimum**: $\min_x f(x) = 4$

- **Minimierer**: $\text{argmin}_x f(x) = 2$

- Die nächsten Vorlesungen: $G = \mathbb{R}^n$
  - Kontinuierliche Variablen (beliebiger Dimensionalität)
  - Keine Nebenbedingungen

Lokale und globale Minima



- Für ein **lokales Minimum** $f(x^L)$ muss gelten

  $$f(x^L) \leq f(x) \quad \forall x \in U(x^L)$$

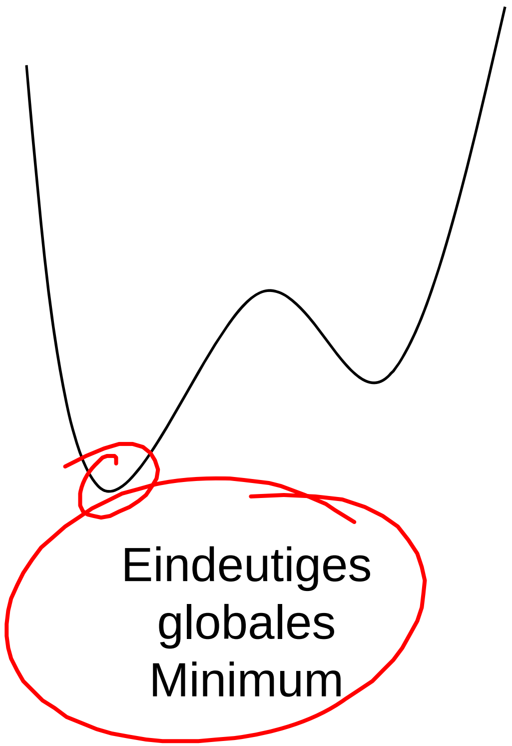
  Dabei ist $U(x^L)$ eine Normkugel mit einem hinreichend kleinen Radius $\epsilon > 0$:

  $$U(x^L) = \{x \in \mathbb{R}^n | \|x - x^L\| < \epsilon\}$$

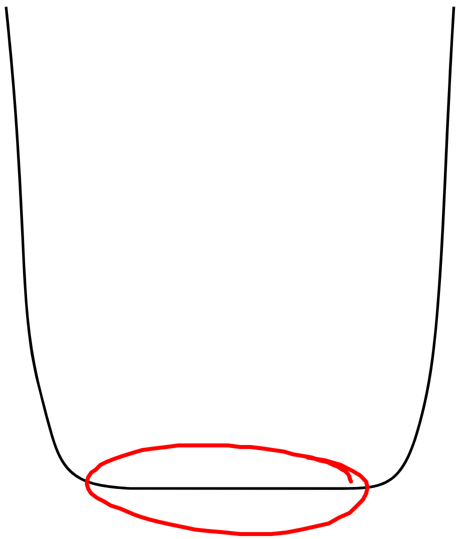
- Für ein **globales Minimum** $f(x^*)$ muss gelten

  $$f(x^*) \leq f(x) \quad \forall x \in \mathbb{R}^n$$

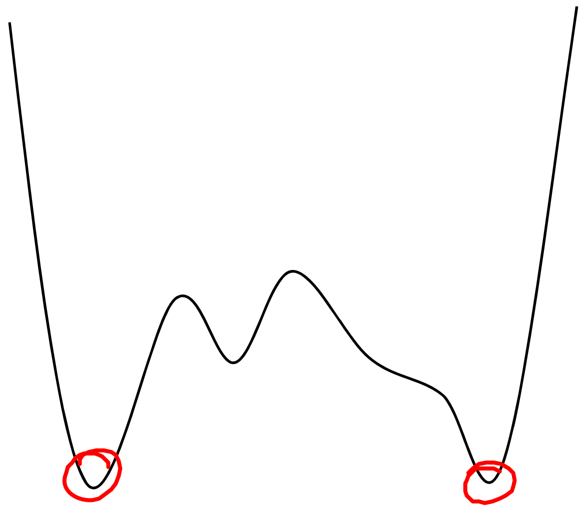
Im allgemeinen kann es mehrere globale Minima geben



Eindeutiges globales Minimum



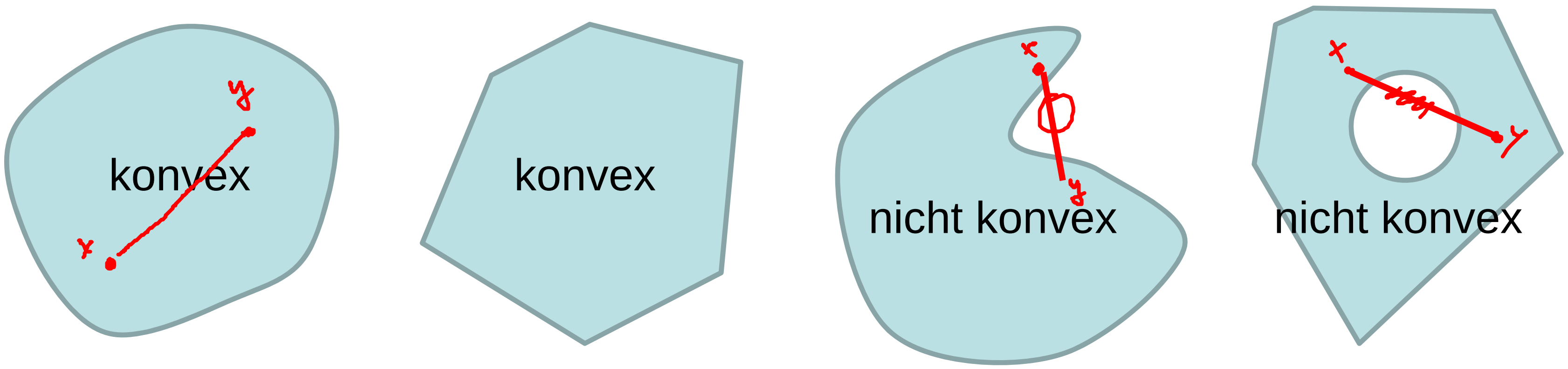
Zusammenhängende Region globaler Minima



Mehrere globale Minima

Kann man feststellen, ob eine Optimierungsaufgabe "gutmütig" ist, also ein eindeutiges globales Minimum und keine weiteren lokalen Minima hat?

Konvexität, konvexe Mengen



- Eine Menge $G \subset \mathbb{R}^n$ ist konvex, wenn für beliebige Punkte $x, y \in G$ auch die Verbindungslinie

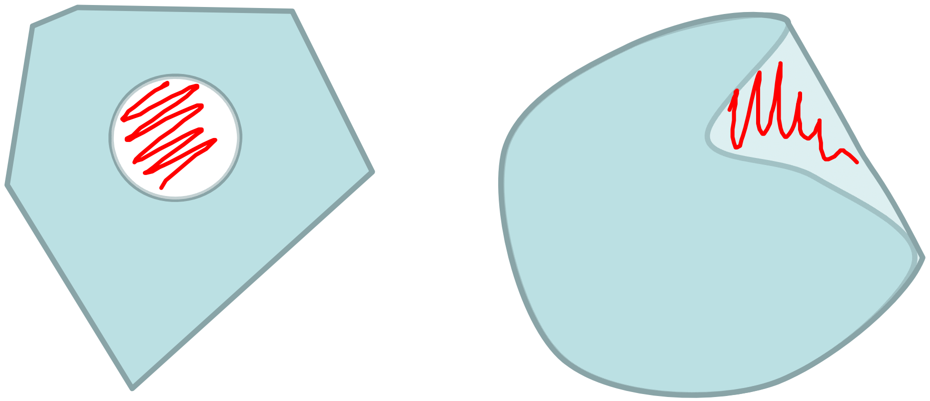
$$[x,y] := \{z := (1-\lambda)x + \lambda y | \lambda \in [0,1]\}$$

in G enthalten ist:

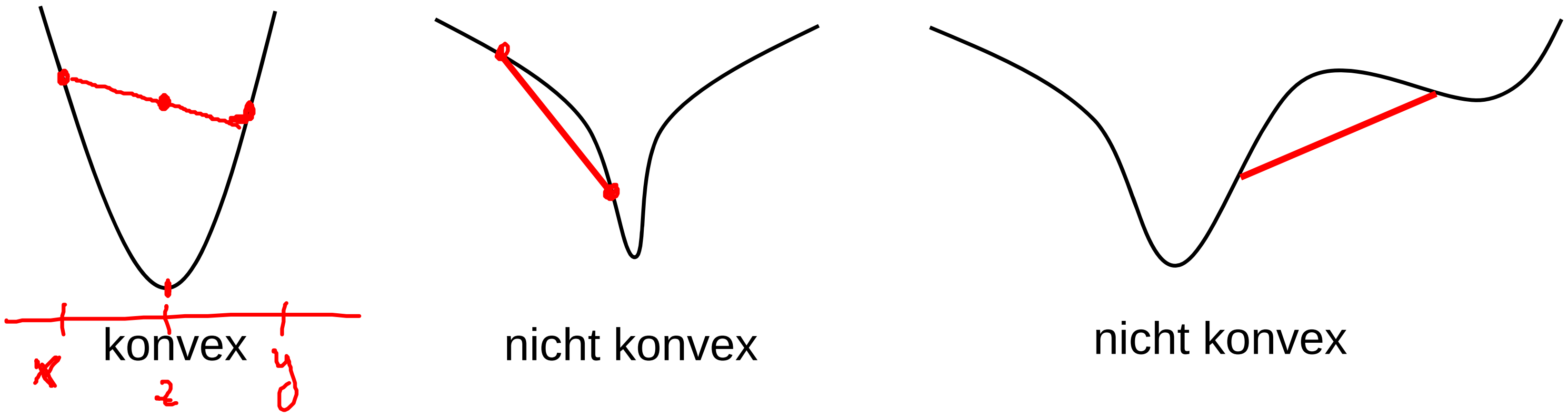
$$x, y \in G \;\Rightarrow\; [x,y] \subset G$$

Konvexe Hülle

- Die **konvexe Hülle** einer Menge G ist die kleinste konvexe Menge, die G vollständig enthält.



Konvexe Funktionen



- Eine über einer konvexen Menge G erklärte Funktion $f: G \to \mathbb{R}$ heißt **konvex,** falls

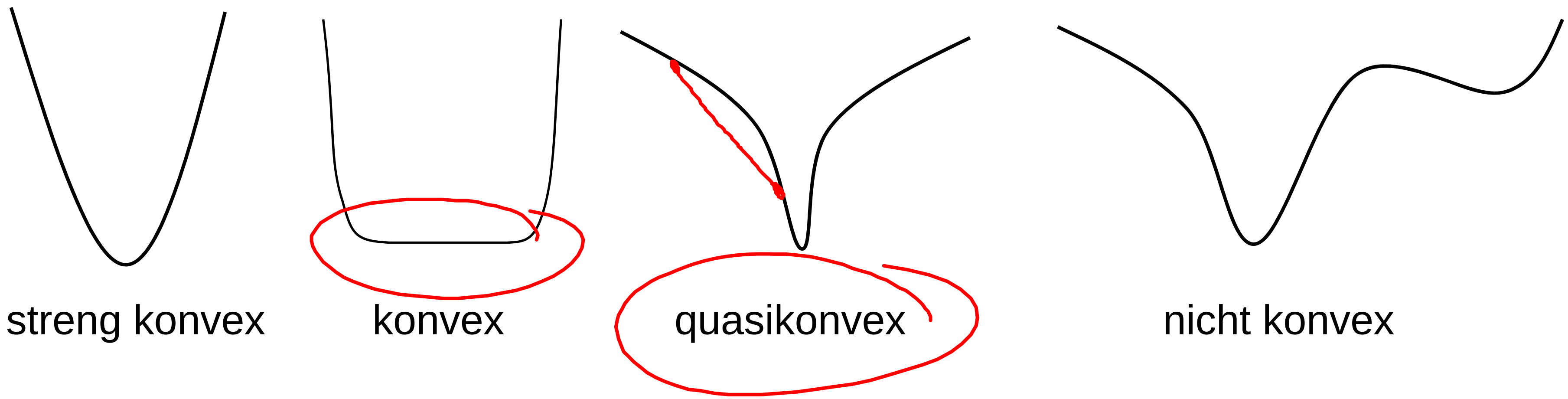
$$x, y \in G; x \neq y \;\Rightarrow\; f((1-\lambda)x+\lambda y) \leq (1-\lambda)f(x)+\lambda f(y) \quad \forall \lambda \in [0,1]$$

- Sie heißt **streng konvex**, falls

$$x, y \in G; x \neq y \;\Rightarrow\; f((1-\lambda)x+\lambda y) < (1-\lambda)f(x)+\lambda f(y) \quad \forall \lambda \in [0,1]$$

Konvexität und globale Minima

- Eine streng konvexe Funktion besitzt ein eindeutiges globales Minimum und keine lokalen Minima.
- Eine konvexe Funktion kann mehrere globale Minima besitzen, jedoch keine zusätzlichen lokalen Minima.
- Eine nicht-konvexe Funktion, die keine lokalen Minima besitzt, wird als **quasikonvex** bezeichnet.



Differenzierbare Funktionen

- Nachfolgend gehen wir davon aus, dass die Funktion f mindestens einmal differenzierbar ist: $f \in \mathcal{C}^1$

- Gradient von f :

$$\nabla f = \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n} \right)^\top$$

- Notwendige Bedingung für ein Minimum von f :

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

- Ist f konvex, so ist dies auch eine hinreichende Bedingung.

- Allgemein stellt die Bedingung $\nabla f(x) = 0$ ein **nichtlineares Gleichungssystem** dar, das numerisch gelöst werden muss.

Gradientenverfahren

- Iteratives Verfahren

  $$x^{k+1} = x^k + \tau^k d^k$$

  mit einem Startpunkt x^0, einer Änderungsrichtung d^k und einer Schrittweite τ^k

- Beim Gradientenverfahren entspricht die Änderungsrichtung dem negativen Gradienten der Zielfunktion f an der aktuellen Stelle x^k

  $$d^k := -\nabla f(x^k)$$

- Die Schrittweise τ^k wird optimal bestimmt, so dass

  $$f(x^k + \tau^k d^k) \leq f(x^k + \alpha d^k) \quad \forall \alpha \geq 0$$

  Mehr zur Schrittweitenbestimmung später…

Overview (Tentatively)

- Chapter 1: Introduction
- Chapter 2: Unconstrained continuous optimization (convexity, gradient descent (first order methods))
- Chapter 3: Unconstrained continuous optimization II (Newton, Quasi-Newton, graduated non-convexity)
- Chapter 4: Constrained optimization (projection, Lagrange multipliers, duality)
- Chapter 5: Linear programming I
- Chapter 6: Linear programming II
- Chapter 7: Discrete optimization and relaxation
- Chapter 8: Branch and bound and miscellaneous
- Chapter 9: Simulated annealing and sampling methods