Optimization of Mechanical Systems

Prof. Dr.-Ing. Prof. E.h. Peter Eberhard (peter.eberhard@itm.uni-stuttgart.de)
teaching assistant: Mario Hermle, M.Sc. (mario.hermle@itm.uni-stuttgart.de)

Institute of Engineering and Computational Mechanics
Pfaffenwaldring 9, 4th floor
70569 Stuttgart

Lecture information

- **Language:** The lecture is taught in English.
- **Audience:** Students of the interdisciplinary graduate study program COMMAS and students from the study programs Mechanical Engineering, Mechatronics, Engineering Cybernetics, Technology Management, Automotive and Engine Technology, Mathematics, Simulation Technology, etc.
- **Credits:** 3 ECTS (2 SWS)
- **WT 23/24:** October 18, 2023 - February 7, 2024;
  Christmas break: December 23, 2023 - January 6, 2024
- **Lectures:** Wednesday, 9:45 - 11:15 a.m., in V7.12 (Pfaffenwaldring 7), weekly;
  starting October 18, 2023
- **Internet:** The course web page can be found at
  `www.itm.uni-stuttgart.de/en/courses/lectures/optimization-of-mechanical-systems`
- **Course material:** Handouts can be downloaded at the course web page.
- **Exercises:** Exercises are an incorporated part of the lecture. Additional material may be provided on the web page (admission to the ILIAS course happens automatically when registering for the course in the C@mpus system).
- **Office hours:** We offer consultations to answer questions also outside the lecture room. These consultations happen during the semester when the need arises. To request a consultation, please contact the teaching assistant Mario Hermle, M.Sc..
- **Exam:** The exam will be scheduled by the examination office, quite likely for the end of February 2024 or beginning of March 2024. Details on location and timing will be announced once available. The exam is mandatory for COMMAS students. To take part in the exam, it is necessary to register with the examination office (Prüfungsamt).

Optimization of Mechanical Systems

Lecture Content

Bibliography

S.K. Agrawal and B.C. Fabien: *Optimization of Dynamic Systems.* Dordrecht: Kluwer Academic Publishers, 1999.

M.P. Bendsoe and O. Sigmund: *Topology Optimization: Theory, Methods and Applications.* Berlin: Springer, 2004.

D. Bestle: *Analyse und Optimierung von Mehrkörpersystemen - Grundlagen und rechnergestützte Methoden.* Berlin: Springer, 1994.

D. Bestle and W. Schiehlen (Eds.): *Optimization of Mechanical Systems*. Proceedings of the IUTAM Symposium Stuttgart, Dordrecht: Kluwer Academic Publishers, 1996.

S. Boyd and L. Vandenberghe: *Convex Optimization*. Cambridge: Cambridge University Press, 2004.

P. Eberhard: *Zur Mehrkriterienoptimierung von Mehrkörpersystemen*. Vol. 227, Reihe 11, Düsseldorf: VDI–Verlag, 1996.

R. Fletcher: *Practical Methods of Optimization*. Chichester: Wiley, 1987.

R. Haftka and Z. Gurdal: *Elements of Structural Optimization*. Dordrecht: Kluwer Academic Publishers, 1992.

E.J. Haug and J.S. Arora: *Applied Optimal Design: Mechanical and Structural Systems.* New York: Wiley, 1979.

J. Nocedal and S.J. Wright: *Numerical Optimization.* New York: Springer, 2006.

A. Osyczka: *Multicriterion Optimization in Engineering*. Chichester: Ellis Horwood, 1984.

K. Schittkowski: *Nonlinear Programming Codes: Information, Tests, Performance. Lect. Notes in Econ. and Math. Sys., Vol. 183.* Berlin: Springer, 1980.

W. Stadler (Ed.): *Multicriteria Optimization in Engineering and in the Sciences*. New York: Plenum Press, 1988.

G.N. Vanderplaats: *Numerical Optimization Techniques for Engineering Design.* Colorado Springs: Vanderplaats Research & Development, 2005.

P. Venkataraman: *Applied Optimization with MATLAB Programming*. New York: Wiley, 2002.

T.L. Vincent and W.J. Grantham: *Nonlinear and Optimal Control Systems.* New York: Wiley, 1997.

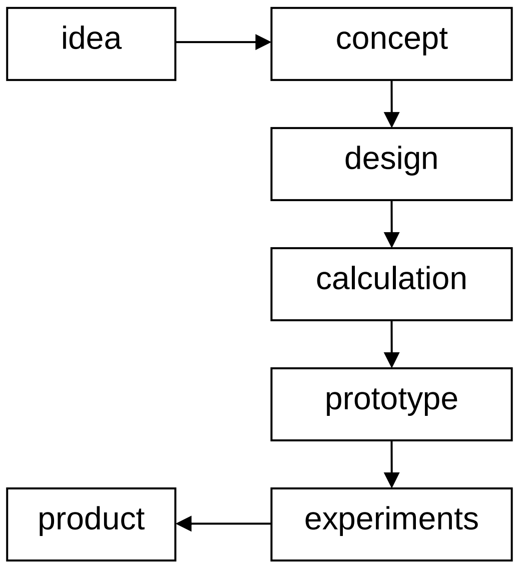
1.1 Motivation

design optimization of mechanical systems

classical/engineering approach

intuition, experience of the design engineer, experiments, and fiddling with hardware prototypes at the end of the design process

sequential process

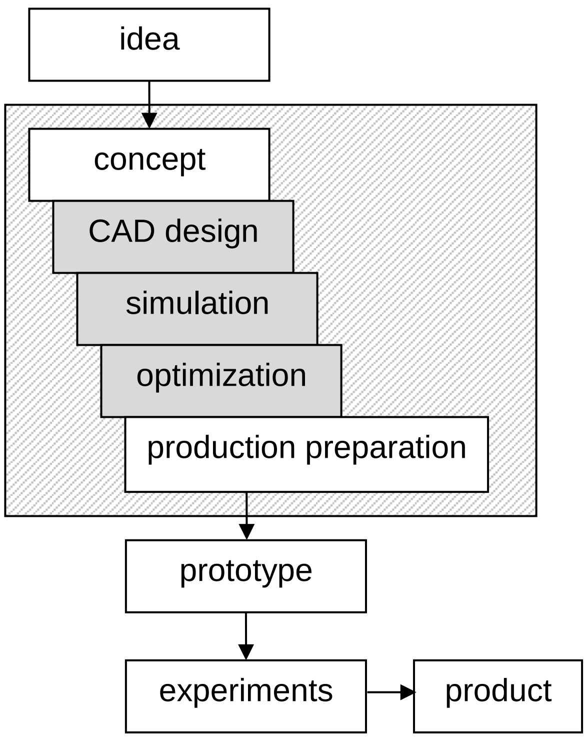


- only small changes of the design possible to influence the system behavior
- more fundamental changes might require the reconsideration of all previous design steps
- “optimal” solution without quantitative objectives
- hardware experiments are costly and time intensive (only few are possible)

mathematical/numerical approach

intuition, experience of the design engineer, and virtual prototypes based on computer simulations throughout the whole design process

concurrent engineering



- provides many design degrees of freedom at beginning of the design process
- parameter studies can be executed swiftly and easily
- systematic way to find optimal solution with respect to defined criteria
- cost efficient
- shortening of the development time
- initial design for hardware experiments is already close to optimum

Optimization beyond mechanical design

The focus of this lecture is on optimization of mechanical systems using the **mathematical/numerical approach**, for which the formulation and the solution of **mathematical optimization** problems are key issues. The concepts and methods are presented in a general manner so that they can be applied in any application domain. Apart from design, applications connected to mechanics include the **control** of actuated mechanical systems such as robots, and the **inference** of unknown parameters, e.g., in mechanical models. Popular **machine learning** methods also build upon optimization. Throughout all these applications, optimization can be seen as a formal, mathematical framework for informed **decision making**. As such, optimization is also extensively used in many other disciplines, such as, e.g., process engineering, logistics, asset management, and economics.

There are different ways to express the same design or decision-making problem as a mathematical optimization problem. However, the concrete **formulation** of the problem and its mathematical properties can have a decisive impact on the expected **solution effort** and the choice of appropriate optimization **algorithms**.

The **systematic formulation of an optimization problem** requires answers to three basic questions:

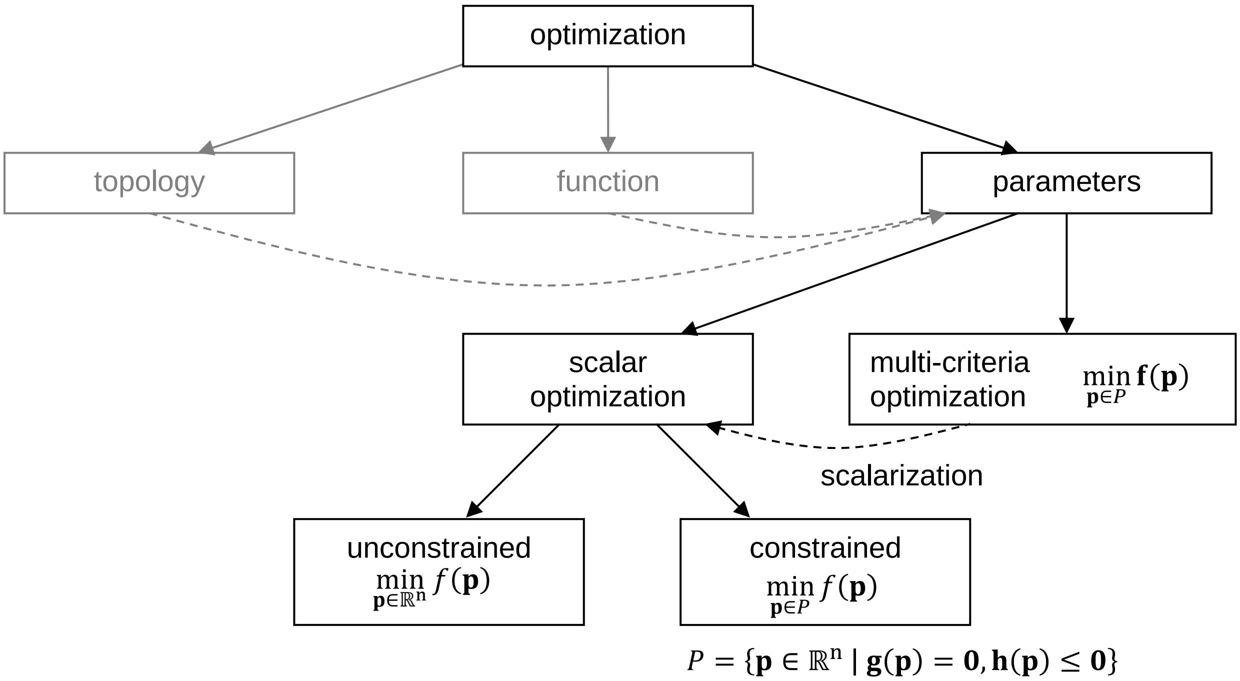
1. What should be achieved by the optimization?

2. Which changeable variables can influence the optimization goals?

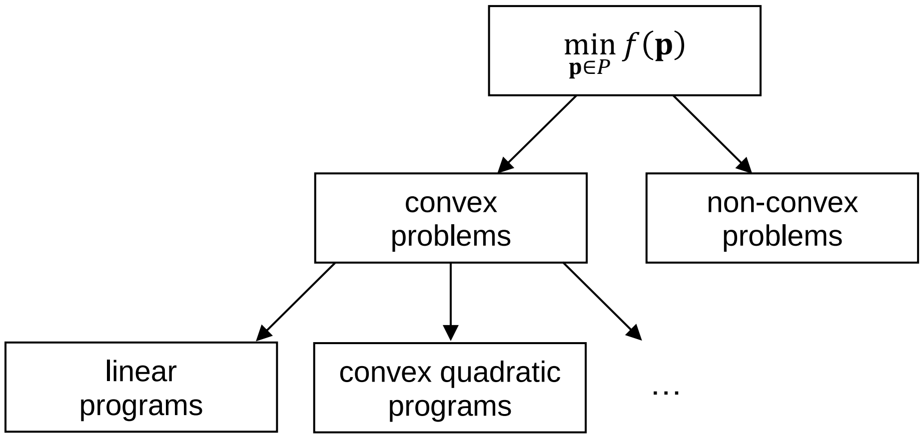
3. Which restrictions apply?

1.3 Classification of Optimization Problems

Optimization problems can be distinguished and classified in various ways, where the following classification makes sense from an application-oriented perspective, e.g., thinking of mechanical design optimization.



Hence, optimization tasks in applications usually rely on the solution of (a series of) scalar optimization problems. Perceiving unconstrained optimization as a special case of constrained optimization with $P = \mathbb{R}^n$, this highlights the importance of constrained scalar optimization. However, the class of constrained scalar optimization problems contains problems of very different properties, heavily influencing which solution strategies are prudent and whether one can even expect to reliably obtain an optimal parameter vector with given solution time and effort. This motivates the following classification.

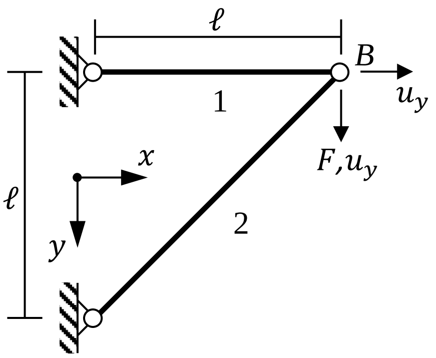


Naturally, further systematic classifications are possible. For instance, this lecture focuses on continuous optimization where the feasible set P is a connected subset of $\mathbb{R}^n$. In contrast, there are also discrete optimization problems where the optimization variable can only take integer values, e.g., the problem of determining the optimal gear to drive a car under given circumstances with maximum efficiency. Mixed-integer problems contain continuous as well as discrete optimization variables.

Optimization in Engineering Applications

Static Analysis – Truss Framework

A simple truss structure, shown to the right, shall be optimized. The truss consists of two round bars with Young's modulus $E = 2{,}1 \cdot 10^{11}\,\mathrm{N/m^2}$ and density $\rho = 2750\,\mathrm{kg/m^3}$. As design variables the radii of the bars r_1 and r_2 are chosen



$$\boldsymbol{p} = \begin{bmatrix} r_1 \\ r_2 \end{bmatrix}, \text{ whereby } 2\,\mathrm{mm} \le r_i \le 5\mathrm{mm},\ \ i = 1{,}2.$$

When applying a force $F = 100\,\mathrm{N}$ at point B a displacement $\boldsymbol{u}$ is caused, which can be computed using the finite element method

$$\boldsymbol{K}\boldsymbol{u} = \boldsymbol{q},$$

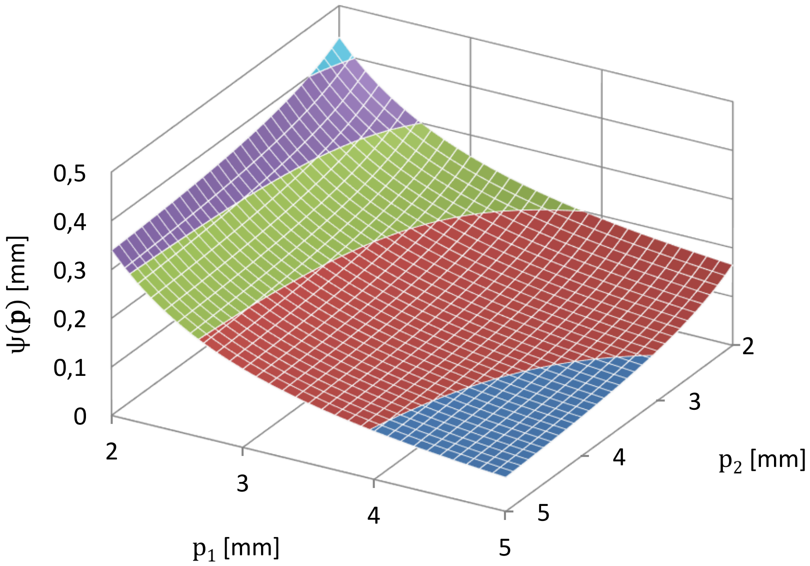
with the stiffness matrix

$$\boldsymbol{K} = \frac{E}{\ell 2\sqrt{2}} \begin{bmatrix} A_2 & A_1 2\sqrt{2} + A_2 \\ A_2 & A_2 \end{bmatrix},$$

the vector of nodal coordinates $\boldsymbol{u} = [u_x \quad u_y]^\top$ and the vector of applied forces $\boldsymbol{q} = [0 \quad F]^\top$. In an optimization the displacement u_y shall be minimized. Thus, the scalar objective function reads

$$\psi(\boldsymbol{p}) = u_y = \frac{\sqrt{2}}{2}\frac{F\ell}{E}\left(\frac{4r_1^2 + \sqrt{2}r_2^2}{\pi r_1^2 r_2^2}\right).$$

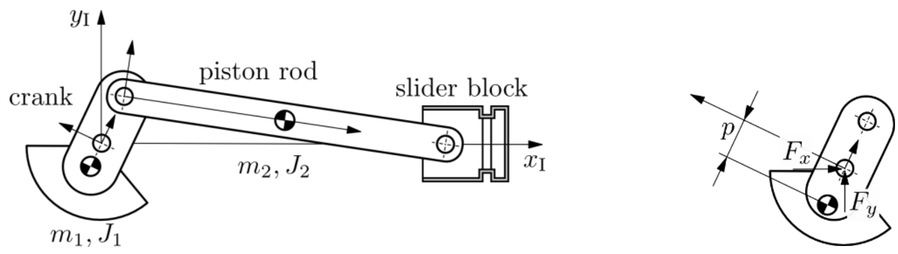
Evaluating $\psi(\boldsymbol{p})$ in the feasible design space returns the following results.



It can be seen that by increasing the radii, the displacement is reduced. Thus, if there are no additional constraint equations, such as mass restriction, the solution of the minimization problem is $p_1^* = p_2^* = 5\,\mathrm{mm}$ and $\psi(\boldsymbol{p}^*) = 0.07\,\mathrm{mm}$.

Dynamic Analysis – Slider-Crank Mechanism

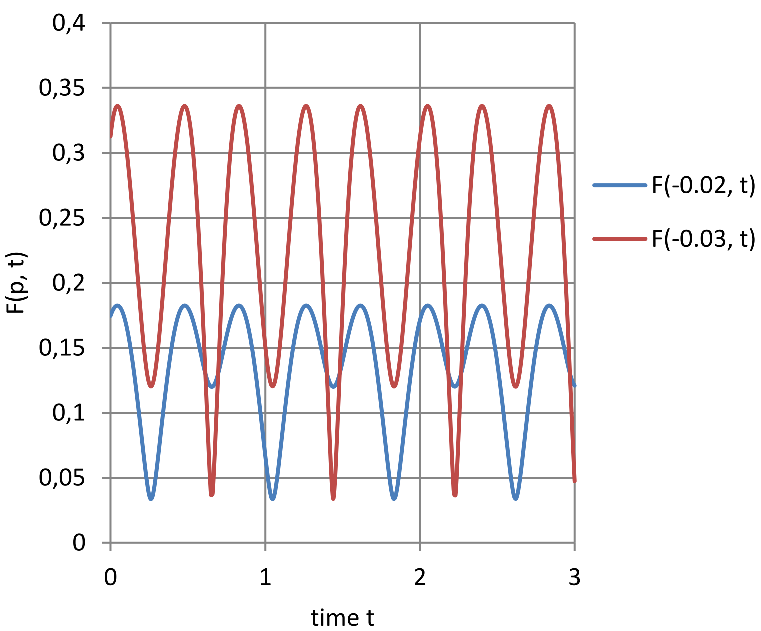
Not only static but also dynamic problems are analyzed and optimized in engineering. For instance, using the method of multibody systems the slider-crank mechanism shown below is modeled. The multibody systems consists of the crank ($m_1 = 0.24\text{kg},\ J_1 = 0.26\ \text{kg m}^2$), the piston rod ($m_2 = 0.16\text{kg},\ J_2 = 0.0016\ \text{kg m}^2$) as well as the slider block ($m_3 = 0.46\text{kg}$). The crank angle is assumed to rotate at constant angular velocity $\dot{\varphi} = 8\ \text{Hz}$ and, thus, the motion of the mechanism is clearly defined.



Performing a simulation for the time domain $t \in [0 \quad 3]\ \text{s}$, the resulting reaction force between the crank and the inertial frame, which is defined as

$$F(p,t) = \sqrt{F_x^2(p,t) + F_y^2(p,t)},$$

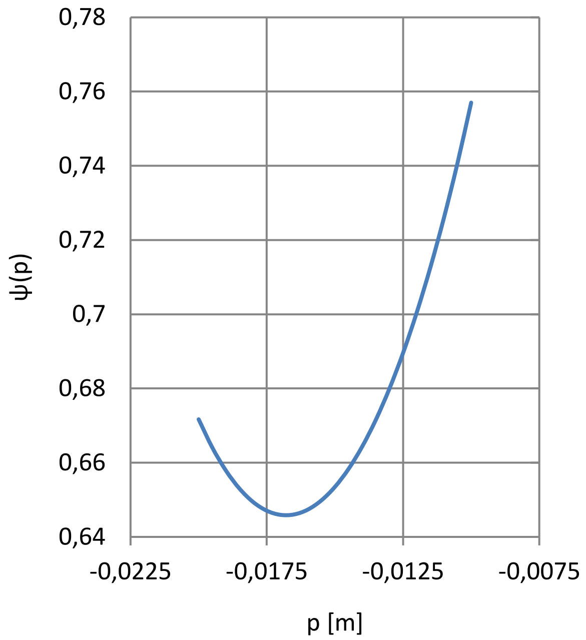
can be computed. For two different values $p = -0.02\ \text{m}$ and $p = -0.03\ \text{m}$ the resulting reaction forces $F(p,t)$ are displayed below.



Performing an optimization, $F(p,t)$ shall be minimized. However, in contrast to static problems, first the transient system response has to be converted into a scalar value. Therefore, the time-dependent resulting reaction force $F(p,t)$ is integrated over the simulation time t. Thus, it holds for the objective function

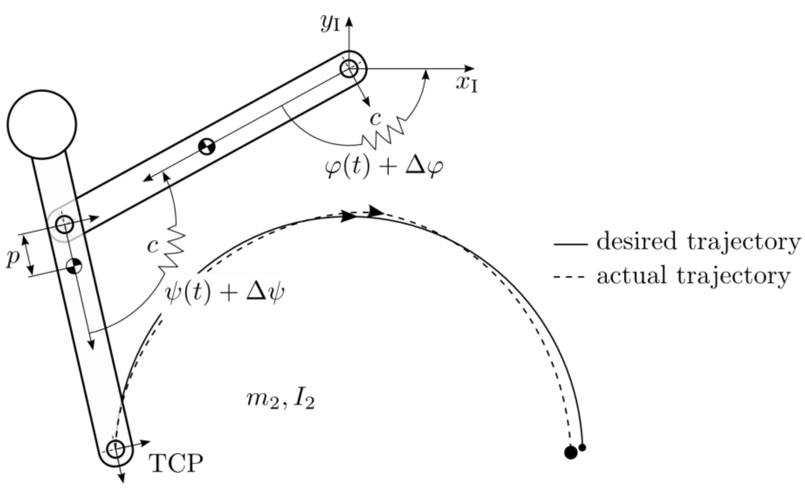
$$\psi(p) = \int_{t^0}^{t^1} F(p,t)dt = \int_{0}^{3s} \sqrt{F_x^2(p,t) + F_y^2(p,t)}dt.$$

Then, evaluating the objective function $\psi(p)$ for $p \in [-0.02 \quad -0.01]^\top$ m the local minimum can be determined as $p^\star \approx -0.017$ and $\psi(p^\star) \approx 0.646$.



Dynamic Analysis – Planar 2-Arm Welding Robot

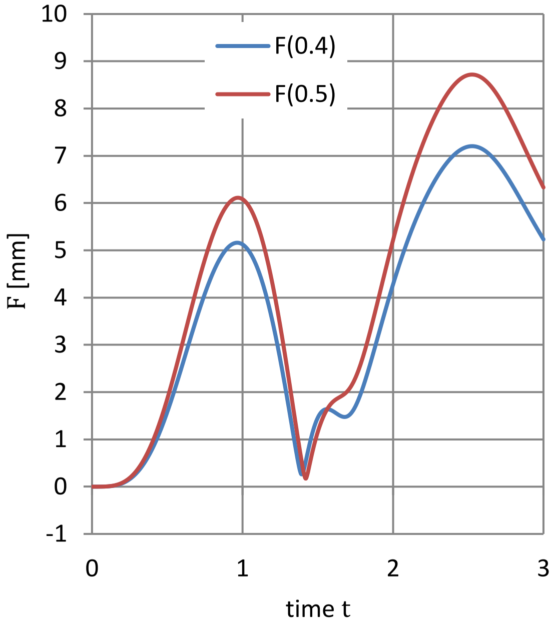
A further example for the optimization of dynamic systems is the planar 2-arm welding robot shown below. For the welding process the tool center point (TCP) has to follow a semi-circular trajectory (—) within 3 second. The joint angles φ and ψ are modeled as rheonomic constraints, i.e., $\varphi = \varphi(t)$ and $\psi = \psi(t)$. However, due to joint elasticity, which is modeled by rotational springs with stiffness c, there are additional rotations of the two arms $\Delta\varphi$ and $\Delta\psi$. These additional rotations represent the generalized degrees of freedom of the system $\boldsymbol{y} = [\Delta\varphi \quad \Delta\psi]^\top$. As a consequence, the actual trajectory of the TCP (- - -) differs from the desired trajectory.

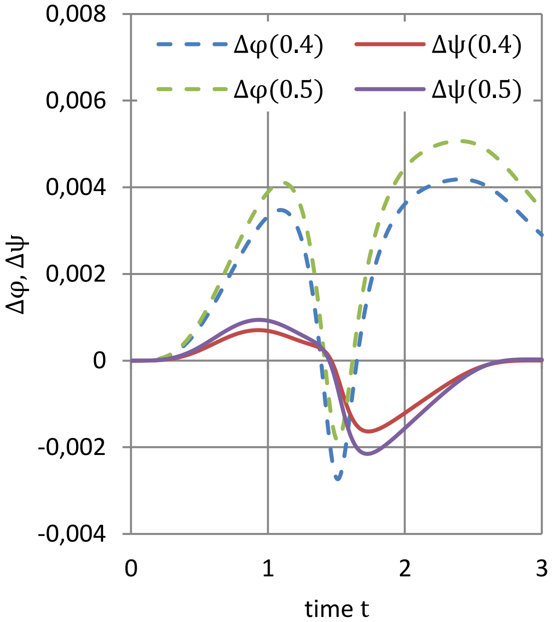


By varying the design variables p the center of gravity of the second arm is changed and, thereby, the tracking error of the TCP shall be reduced. The tracking error F is determined by the Euclidean distance between the actual position $\boldsymbol{r}_\mathrm{a} = [x_\mathrm{a} \quad y_\mathrm{a}]^\top$ and the desired position $\boldsymbol{r}_\mathrm{d} = [x_\mathrm{d} \quad y_\mathrm{d}]^\top$ and is computed as

$$F(p,\boldsymbol{y},t) = \sqrt{\big(x_\mathrm{a}(p,\boldsymbol{y},t) - x_\mathrm{d}(t)\big)^2 + \big(y_a(p,\boldsymbol{y},t) - y_\mathrm{d}(t)\big)^2}.$$

It can be seen, that not only the tracking error F but also that the generalized degrees of freedom $\Delta\varphi$ and $\Delta\psi$ depend on the design variable p.

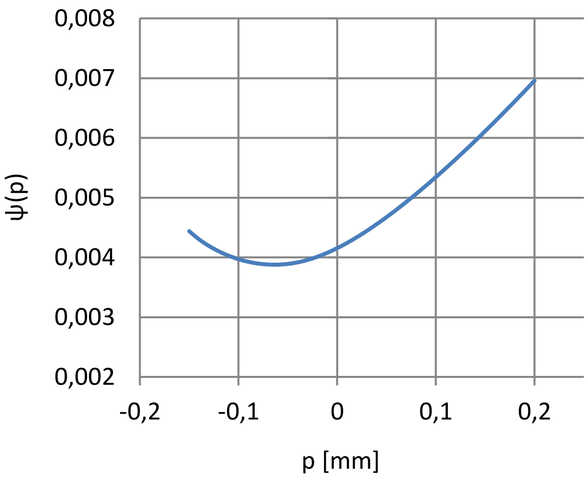




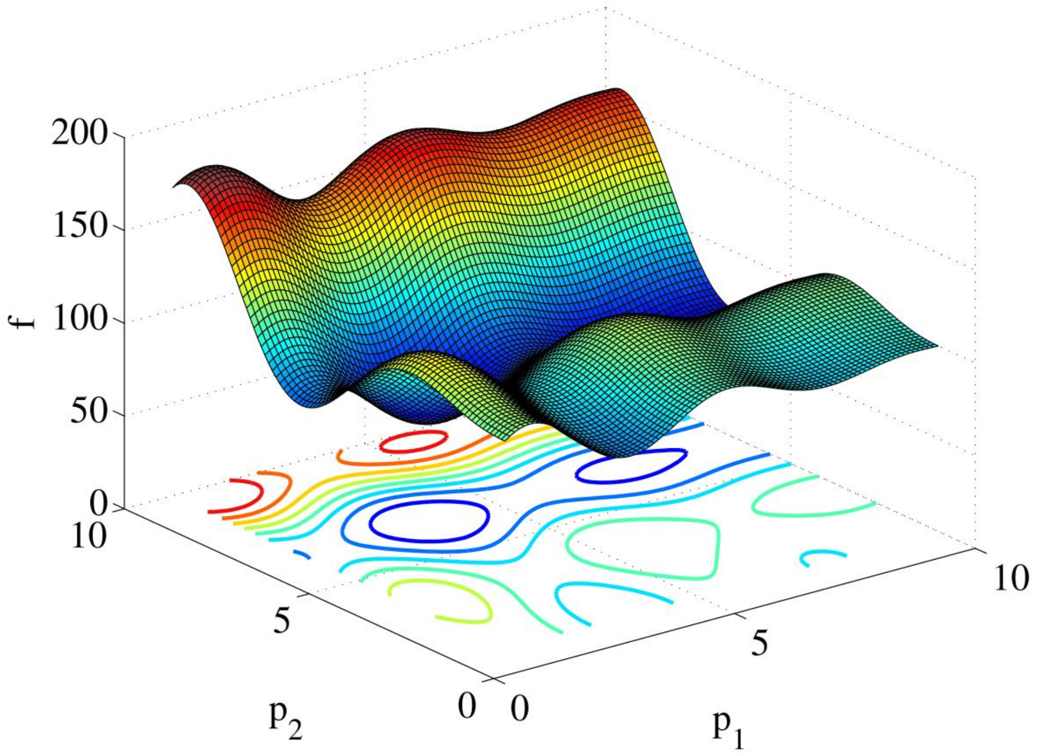
To obtain a scalar objective function, the tracking error F is integrated over the simulation time

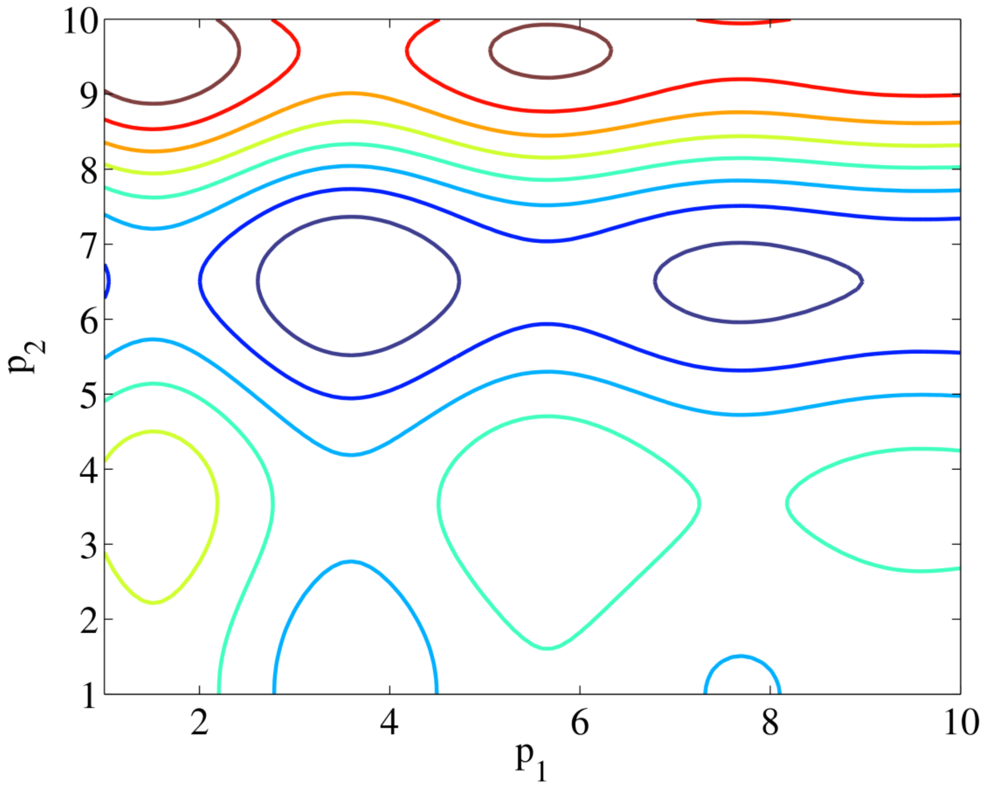
$$\psi(p) = \int_{t^0}^{t^1} F(p,\boldsymbol{y},t)dt = \int_0^{3s} \sqrt{\big(x_a(p,\boldsymbol{y},t) - x_d(t)\big)^2 + \big(y_a(p,\boldsymbol{y},t) - y_d(t)\big)^2} dt.$$

Evaluating the objective function for $p \in [-0.02 \quad -0.01]^\top$, a local minimum can be graphically determined at $p^\star \approx -0.04$ and $\psi(p^*) \approx 0.0039$.

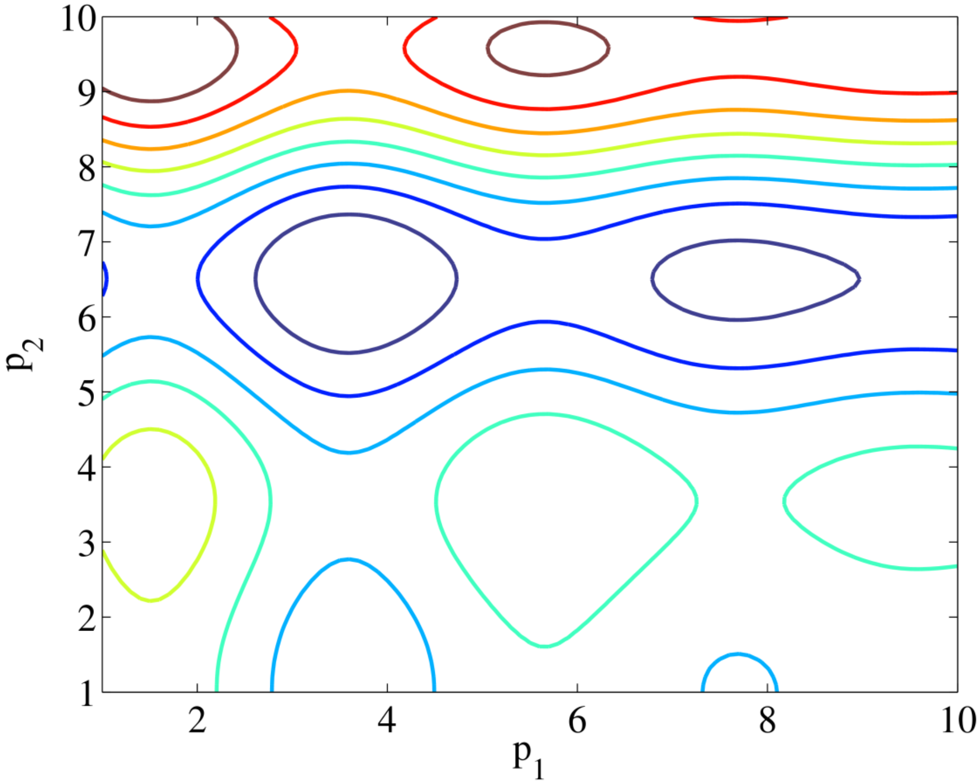


Geometric visualization in 2D

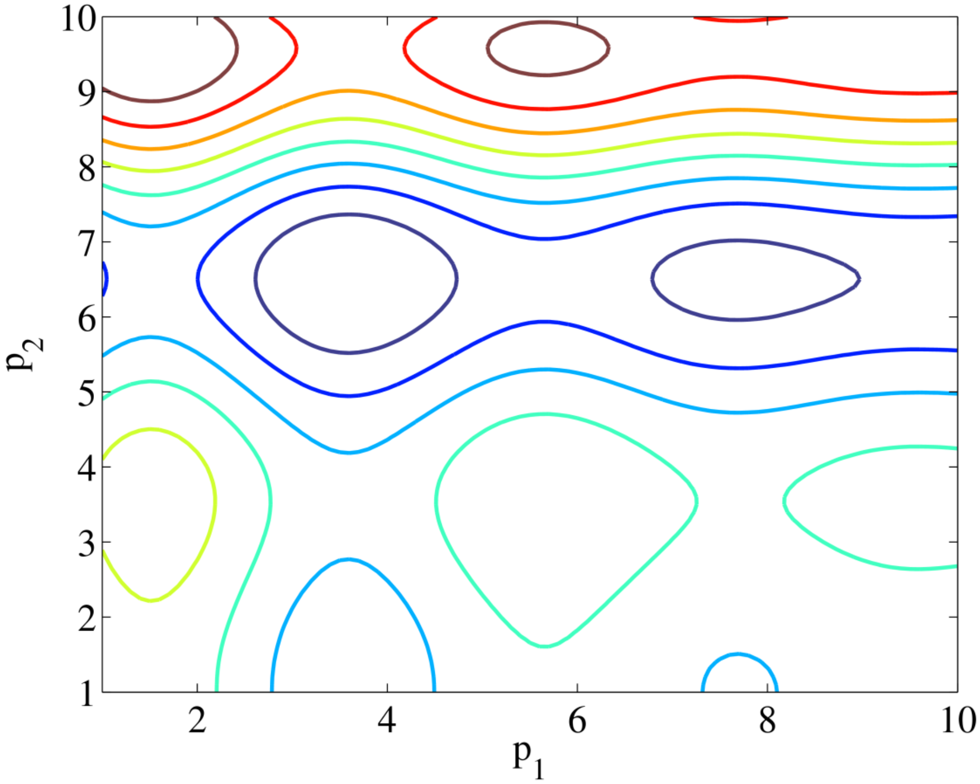




inequality constraints



equality constraints



Matrix Algebra and Matrix Analysis

vector $\boldsymbol{x} \in \mathbb{R}^n$: $\boldsymbol{x} = [x_1 \quad \cdots \quad x_n]^\top$, $x_i \in \mathbb{R}$,

matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$: $\boldsymbol{A} = \begin{bmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & & \vdots \\ A_{m1} & \cdots & A_{mn} \end{bmatrix}$, $A_{ij} \in \mathbb{R}$.

Basic Operations

operation	notation	components	mapping
addition	$\boldsymbol{C} = \boldsymbol{A} + \boldsymbol{B}$	$C_{ij} = A_{ij} + B_{ij}$	$\mathbb{R}^{m\times n} \times \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$
multiplication with scalar	$\boldsymbol{C} = \alpha\, \boldsymbol{A}$	$C_{ij} = \alpha\, A_{ij}$	$\mathbb{R} \times \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$
transpose	$\boldsymbol{C} = \boldsymbol{A}^\top$	$C_{ij} = A_{ji}$	$\mathbb{R}^{m\times n} \to \mathbb{R}^{n\times m}$
differentiation	$\boldsymbol{C} = \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{A}$ $\boldsymbol{C} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{y}}$	$C_{ij} = \frac{\mathrm{d}}{\mathrm{d}t}A_{ij}$ $C_{ij} = \frac{\partial x_i}{\partial y_j}$	$\mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ $\mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^{m\times n}$
matrix multiplication	$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ $\boldsymbol{C} = \boldsymbol{A}\,\boldsymbol{B}$	$y_i = \sum_k A_{ik}\, x_k$ $C_{ij} = \sum_k A_{ik}\, B_{kj}$	$\mathbb{R}^{m\times n} \times \mathbb{R}^n \to \mathbb{R}^m$ $\mathbb{R}^{m\times n} \times \mathbb{R}^{n\times p} \to \mathbb{R}^{m\times p}$
scalar product (dot/inner product)	$\alpha = \mathbf{x} \cdot \mathbf{y}$	$\alpha = \sum_k x_k\, y_k$	$\mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$
outer product	$\boldsymbol{A} = \boldsymbol{x} \otimes \boldsymbol{y}$	$A_{ij} = x_i\, y_j$	$\mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^{m\times n}$

Basic Rules

addition:

$$\boldsymbol{A} + (\boldsymbol{B} + \boldsymbol{C}) = (\boldsymbol{A} + \boldsymbol{B}) + \boldsymbol{C}$$
$$\boldsymbol{A} + \boldsymbol{B} = \boldsymbol{B} + \boldsymbol{A}$$

multiplication with scalar:

$$\alpha(\boldsymbol{A}\,\boldsymbol{B}) = (\alpha\,\boldsymbol{A})\,\boldsymbol{B} = \boldsymbol{A}\,(\alpha\,\boldsymbol{B})$$
$$\alpha(\boldsymbol{A} + \boldsymbol{B}) = \alpha\,\boldsymbol{A} + \alpha\,\boldsymbol{B}$$

transpose:

$$(\boldsymbol{A}^\top)^\top = \boldsymbol{A}$$
$$(\boldsymbol{A} + \boldsymbol{B})^\top = \boldsymbol{A}^\top + \boldsymbol{B}^\top$$
$$(\alpha\boldsymbol{A}^\top)^\top = \alpha\boldsymbol{A}$$
$$(\boldsymbol{A}\,\boldsymbol{B})^\top = \boldsymbol{B}^\top\,\boldsymbol{A}^\top$$

differentiation:

$$\frac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{A} + \boldsymbol{B}) = \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{A} + \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{B}$$
$$\frac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{A}\,\boldsymbol{B}) = \left(\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{A}\right)\boldsymbol{B} + \boldsymbol{A}\left(\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{B}\right)$$
$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{f}(\boldsymbol{x}) = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t}$$

matrix multiplication:

$$\boldsymbol{A}\,(\boldsymbol{B} + \boldsymbol{C}) = \boldsymbol{A}\,\boldsymbol{B} + \boldsymbol{A}\,\boldsymbol{C}$$
$$\boldsymbol{A}\,(\boldsymbol{B}\,\boldsymbol{C}) = (\boldsymbol{A}\,\boldsymbol{B})\,\boldsymbol{C}$$
$$\boldsymbol{A}\,\boldsymbol{B} \neq \boldsymbol{B}\,\boldsymbol{A} \quad \text{(in general)}$$

scalar product:

$$\boldsymbol{x}\cdot\boldsymbol{y} = \boldsymbol{y}\cdot\boldsymbol{x} = \boldsymbol{x}^\top\boldsymbol{y} = \boldsymbol{y}^\top\boldsymbol{x}$$
$$\boldsymbol{x}\cdot\boldsymbol{y} \geq 0 \quad \forall\,\boldsymbol{x}, \qquad \boldsymbol{x}\cdot\boldsymbol{x} = 0 \Leftrightarrow \boldsymbol{x} = 0$$
$$\boldsymbol{x}\cdot\boldsymbol{y} = 0 \;\Leftrightarrow\; \boldsymbol{x},\ \boldsymbol{y} \ \text{orthogonal}$$

Quadratic Matrices

identity matrix

$$\boldsymbol{I} = \begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{bmatrix}$$

diagonal matrix

$$\boldsymbol{D} = \mathrm{diag}(D_1, \dots D_n) \; = \begin{bmatrix} D_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & D_n \end{bmatrix}$$

inverse matrix $\boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{A}\,\boldsymbol{A}^{-1} = \boldsymbol{I}$

$(\boldsymbol{A}\,\boldsymbol{B})^{-1} = \boldsymbol{B}^{-1}\,\boldsymbol{A}^{-1}$

orthogonal matrix $\boldsymbol{A}^{-1} = \boldsymbol{A}^\top$

symmetric matrix $\boldsymbol{A} = \boldsymbol{A}^\top$

skew symmetric matrix $\boldsymbol{A} = -\boldsymbol{A}^\top$

decomposition

$$\boldsymbol{A} = \underbrace{\tfrac{1}{2}(\boldsymbol{A} + \boldsymbol{A}^\top)}_{\boldsymbol{B}=\boldsymbol{B}^\top} + \underbrace{\tfrac{1}{2}(\boldsymbol{A} - \boldsymbol{A}^\top)}_{\boldsymbol{C}=-\boldsymbol{C}^\top}$$

skew symmetric 3×3 matrix $\tilde{\boldsymbol{a}} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}$

$$\tilde{\boldsymbol{a}}\,\boldsymbol{b} = \boldsymbol{a} \times \boldsymbol{b}$$

$$\tilde{\boldsymbol{a}}\,\boldsymbol{b} = -\tilde{\boldsymbol{b}}\,\boldsymbol{a}$$

$$\tilde{\boldsymbol{a}}\,\tilde{\boldsymbol{b}} = \boldsymbol{b} \otimes \boldsymbol{a} - (\boldsymbol{a} \cdot \boldsymbol{b})\boldsymbol{I}$$

$$(\widetilde{\tilde{\boldsymbol{a}}\,\boldsymbol{b}}) = \boldsymbol{b} \otimes \boldsymbol{a} - \boldsymbol{a} \otimes \boldsymbol{b}$$

symmetric, positive definite matrix $\boldsymbol{A}$: $\boldsymbol{x}^\top\boldsymbol{A}\,\boldsymbol{x} > 0 \quad \forall\ \boldsymbol{x} \neq \boldsymbol{0}$

$\Leftrightarrow$ eigenvalues $\lambda_k > 0\ \ \forall\ k \in \{1, \dots, n\}$

symmetric, positive semidefinite matrix $\boldsymbol{A}$: $\boldsymbol{x}^\top\boldsymbol{A}\,\boldsymbol{x} \geq 0 \quad \forall\ \boldsymbol{x}$

$\Leftrightarrow$ eigenvalues $\lambda_k \geq 0\ \ \forall\ k \in \{1, \dots, n\}$

Notation

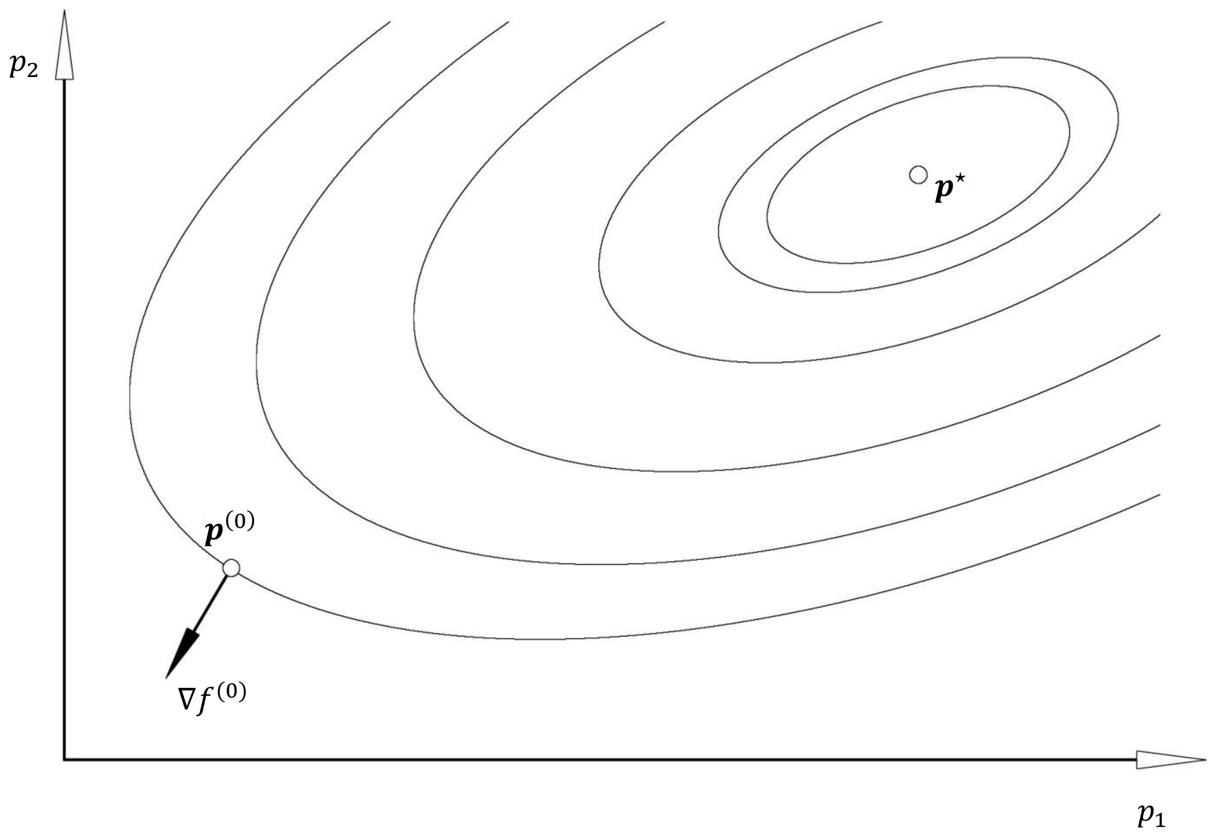
$a \in \mathbb{R}$, $\boldsymbol{b} \in \mathbb{R}^{\mathrm{m}}$, $\boldsymbol{C} \in \mathbb{R}^{\mathrm{m}\times\mathrm{n}}$

	matrix notation	index notation
1) derivative with respect to a scalar $x \in \mathbb{R}$	• $\frac{\partial a}{\partial x}$ • $\frac{\partial \boldsymbol{b}}{\partial x} = \begin{bmatrix} \frac{\partial b_1}{\partial x} \\ \vdots \\ \frac{\partial b_m}{\partial x} \end{bmatrix}$ • $\frac{\partial \boldsymbol{C}}{\partial x} = \begin{bmatrix} \frac{\partial C_{11}}{\partial x} & \cdots & \frac{\partial C_{1n}}{\partial x} \\ \vdots & \ddots & \vdots \\ \frac{\partial C_{m1}}{\partial x} & \cdots & \frac{\partial C_{mn}}{\partial x} \end{bmatrix}$	• $\frac{\partial a}{\partial x}$ • $\frac{\partial b_i}{\partial x}$ • $\frac{\partial C_{ij}}{\partial x}$
2) derivative with respect to a vector $\boldsymbol{x} \in \mathbb{R}^{\mathrm{n}}$	• $\frac{\partial a}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial a}{\partial x_1} \\ \vdots \\ \frac{\partial a}{\partial x_n} \end{bmatrix}$ • $\frac{\partial \boldsymbol{b}}{\partial x} = \begin{bmatrix} \frac{\partial b_1}{\partial x_1} & \cdots & \frac{\partial b_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial b_m}{\partial x_1} & \cdots & \frac{\partial b_m}{\partial x_n} \end{bmatrix}$	• $\frac{\partial a}{\partial x_i}$ • $\frac{\partial b_i}{\partial x_j}$
3) derivative of a scalar product $\boldsymbol{b}^\top \boldsymbol{x}$, $\boldsymbol{x} \in \mathbb{R}^{\mathrm{m}}$, $b_i = \mathrm{const}$	• $\frac{\partial}{\partial \boldsymbol{x}}\left(\boldsymbol{b}^\top \boldsymbol{x}\right) = \boldsymbol{b}$	• $\frac{\partial}{\partial x_j}(b_i\, x_i) = b_i \frac{\partial x_i}{\partial x_j} = b_i\, \delta_{ij} = b_j$
4) derivative of a quadratic form $\boldsymbol{x}^\top \boldsymbol{A} \boldsymbol{x}$, $\mathbf{A} = \mathbf{A}^\top \in \mathbb{R}^{n\times n}$, $A_{ij} = \mathrm{const.}$, $\boldsymbol{x} \in \mathbb{R}^{\mathrm{n}}$	• $\frac{\partial}{\partial \boldsymbol{x}}\left(\boldsymbol{x}^\top \boldsymbol{A} \boldsymbol{x}\right) = 2\, \boldsymbol{A}\boldsymbol{x}$	• $\frac{\partial}{\partial x_j}(x_m\, A_{mn}\, x_n) = 2\, A_{jn} x_n$

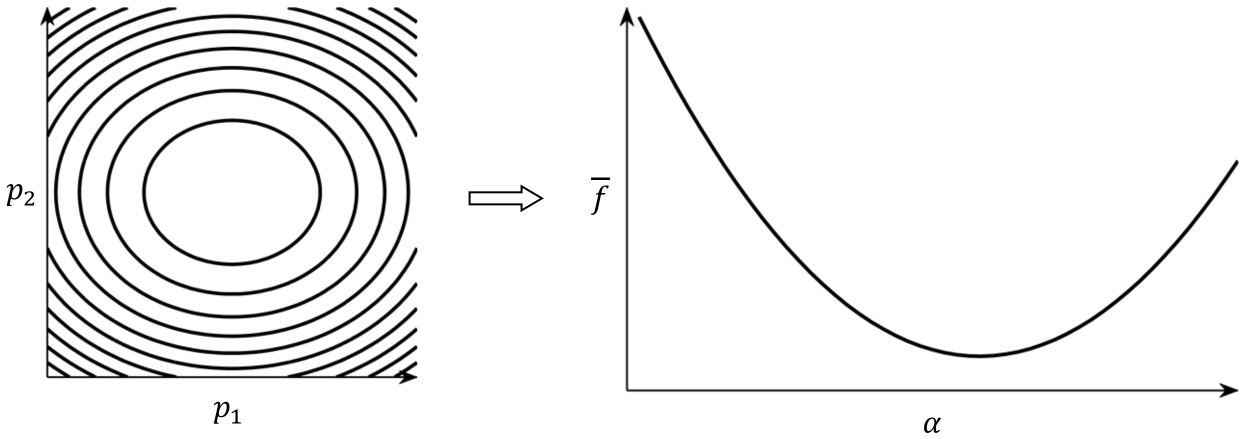
Deterministic Optimization Strategies

optimization strategy	search direction	model order	information order
search parallel to the axes	$\boldsymbol{s}^{(v)} = \boldsymbol{e}_{v \bmod n}$		
gradient based method	$\boldsymbol{s}^{(v)} = -\nabla f^{(v)}$		
conjugate gradient method	$\boldsymbol{s}^{(0)} = -\nabla f^{(0)}$ $\boldsymbol{s}^{(v+1)} = -\nabla f^{(v+1)} + \frac{\left\Vert \nabla f^{(v+1)} \right\Vert^2}{\Vert \nabla f^{(v)} \Vert^2} \boldsymbol{s}^{(v)}$		
Newton method	$\boldsymbol{s}^{(v)} = -\left(\nabla^2 f^{(v)}\right)^{-1} \cdot \nabla f^{(v)}$		

Example: quadratic criteria function



Line Search



possible requirements for line search

- exact minimization:
  $\overline{f}'(\alpha^{(v)}) = \boldsymbol{s}^{(v)} \cdot \nabla f^{(v+1)} \overset{!}{=} 0$
  many function evaluations → inefficient

- sufficient improvement:
  in order to avoid infinitesimally small improvements,
  some conditions have been proposed,
  e.g., Wolfe-Powell conditions
  $\overline{f}(\alpha) \overset{!}{\leq} \overline{f}(0) + \alpha\rho\overline{f}'(0),\quad \rho \in (0{,}1)$, e.g., $\rho = 0.01$
  $\overline{f}'(\alpha) \overset{!}{\geq} \sigma\overline{f}'(0),\quad \sigma \in (\rho, 1)$, e.g., $\sigma = 0.1$

Karush-Kuhn-Tucker Conditions

If $\boldsymbol{p}^\star$ is a regular point and a local minimizer of the optimization problem

$$\min_{\boldsymbol{p}\in P} f(\boldsymbol{p}) \quad \text{with} \quad P = \left\{\boldsymbol{p} \in \mathbb{R}^n \,\middle|\, \boldsymbol{g}(\boldsymbol{p}) = \boldsymbol{0},\ \boldsymbol{h}(\boldsymbol{p}) \leq \boldsymbol{0},\ \boldsymbol{g}\colon \mathbb{R}^n \to \mathbb{R}^\ell,\ \boldsymbol{h}\colon \mathbb{R}^n \to \mathbb{R}^m\right\},$$

then Lagrange multipliers $\boldsymbol{\lambda}^\star$ and $\boldsymbol{\mu}^\star$ exist, for which $\boldsymbol{p}^\star$, $\boldsymbol{\lambda}^\star$, $\boldsymbol{\mu}^\star$ fulfill the conditions

$$\left.\left(\nabla f - \sum_{i=1}^{\ell} \lambda_i \left(\frac{\partial g_i}{\partial \boldsymbol{p}}\right)^\top - \sum_{j=1}^{m} \mu_j \left(\frac{\partial h_j}{\partial \boldsymbol{p}}\right)^\top\right)\right|_{\boldsymbol{p}^\star,\boldsymbol{\lambda}^\star,\boldsymbol{\mu}^\star} = \boldsymbol{0},$$

$$\boldsymbol{g}(\boldsymbol{p}^\star) = \boldsymbol{0},$$

$$\boldsymbol{h}(\boldsymbol{p}^\star) \leq \boldsymbol{0},$$

$$\boldsymbol{\mu}^\star \leq \boldsymbol{0},$$

$$\mu_j^\star h_j(\boldsymbol{p}^\star) = 0, \quad \mathrm{j} \in \{1, \ldots, \mathrm{m}\}.$$

If we introduce the Lagrangian function

$$L(\boldsymbol{p}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \coloneqq f(\boldsymbol{p}) - \textstyle\sum_{i=1}^{\ell} \lambda_i g_i(\boldsymbol{p}) - \sum_{j=1}^{m} \mu_j h_j(\boldsymbol{p}),$$

we can write the Karush-Kuhn-Tucker conditions as

$$\left(\left.\frac{\partial L}{\partial \boldsymbol{p}}\right|_{\boldsymbol{p}^\star,\boldsymbol{\lambda}^\star,\boldsymbol{\mu}^\star}\right)^\top = \boldsymbol{0}\,, \quad \left(\left.\frac{\partial L}{\partial \boldsymbol{\lambda}}\right|_{\boldsymbol{p}^\star,\boldsymbol{\lambda}^\star,\boldsymbol{\mu}^\star}\right)^\top = \boldsymbol{0}\,, \quad \left(\left.\frac{\partial L}{\partial \boldsymbol{\mu}}\right|_{\boldsymbol{p}^\star,\boldsymbol{\lambda}^\star,\boldsymbol{\mu}^\star}\right)^\top \geq \boldsymbol{0}\,,$$

$$\boldsymbol{\mu}^\star \leq \boldsymbol{0}\,, \qquad \mu_j^\star h_j(\boldsymbol{p}^\star) = \boldsymbol{0}\,, \qquad j \in \{1, \ldots, m\}\,.$$

Lagrange-Newton-Method / Sequential Quadratic Programming (SQP)

= Recursive Quadratic Programming (RQP)
= Variable Metric Method

Simplifying assumption: Only equality constraints, i.e.,

$$\min_{\boldsymbol{p}\in P} f(\boldsymbol{p}) \quad \text{with} \quad P = \{\boldsymbol{p} \in \mathbb{R}^n \mid \boldsymbol{g}(\boldsymbol{p}) = \boldsymbol{0}\},$$

with the corresponding Karush-Kuhn-Tucker conditions reading

$$\boldsymbol{a}(\boldsymbol{p}^\star, \boldsymbol{\lambda}^\star) \coloneqq \begin{bmatrix} \left(\dfrac{\partial L}{\partial \boldsymbol{p}}\right)^\top \\ \left(\dfrac{\partial L}{\partial \boldsymbol{\lambda}}\right)^\top \end{bmatrix} = \begin{bmatrix} \nabla f(\boldsymbol{p}^\star) - \sum_i \nabla g_i(\boldsymbol{p}^\star)\lambda_i^\star \\ \boldsymbol{g}(\boldsymbol{p}^\star) \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix}.$$

$$\begin{bmatrix} \nabla^2 f - \sum_i \nabla^2 g_i \lambda_i & -\left(\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{p}}\right)^{\mathsf{T}} \\ \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{p}} & \boldsymbol{0} \end{bmatrix}^{(\nu)} \begin{bmatrix} \boldsymbol{p}^{(\nu+1)} - \boldsymbol{p}^{(\nu)} \\ \boldsymbol{\lambda}^{(\nu+1)} - \boldsymbol{\lambda}^{(\nu)} \end{bmatrix} = -\begin{bmatrix} \nabla f - \left(\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{p}}\right)^{\mathsf{T}} \boldsymbol{\lambda} \\ \boldsymbol{g} \end{bmatrix}^{(\nu)}$$

$$\begin{bmatrix} \boldsymbol{W}^{(\nu)} & -\left(\frac{\partial \boldsymbol{g}^{(\nu)}}{\partial \boldsymbol{p}}\right)^{\mathsf{T}} \\ \frac{\partial \boldsymbol{g}^{(\nu)}}{\partial \boldsymbol{p}} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \delta \boldsymbol{p}^{(\nu+1)} \\ \boldsymbol{\lambda}^{(\nu+1)} \end{bmatrix} = -\begin{bmatrix} \nabla f^{(\nu)} \\ \boldsymbol{g}^{(\nu)} \end{bmatrix}$$

In the case that the performance function and constraint equations are general nonlinear functions, the parameter variation $\delta\boldsymbol{p}$ is not necessarily the best possible parameter variation for the original optimization problem. In order to achieve a higher flexibility, the method can be combined with a line search, i.e., $\delta\boldsymbol{p} = \alpha\boldsymbol{s}$,

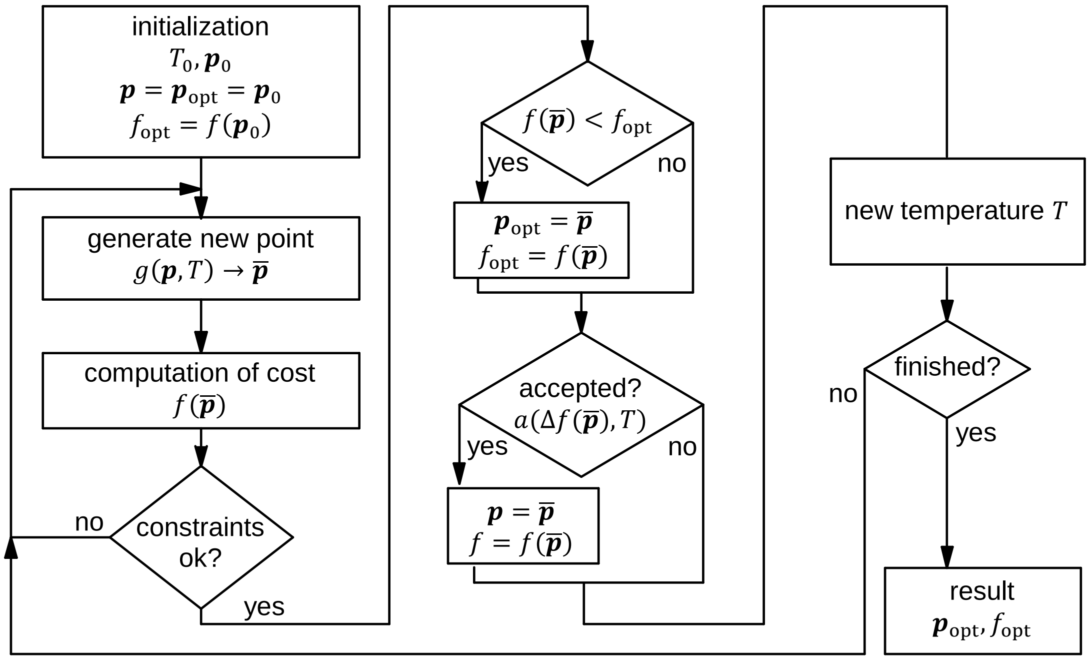
$$\begin{bmatrix} \boldsymbol{W}^{(\nu)} & -\left(\dfrac{\partial \boldsymbol{g}^{(\nu)}}{\partial \boldsymbol{p}}\right)^{\top} \\ \dfrac{\partial \boldsymbol{g}^{(\nu)}}{\partial \boldsymbol{p}} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{s} \\ \boldsymbol{\lambda} \end{bmatrix} = -\begin{bmatrix} \nabla f^{(\nu)} \\ \boldsymbol{g}^{(\nu)} \end{bmatrix},$$

which corresponds to

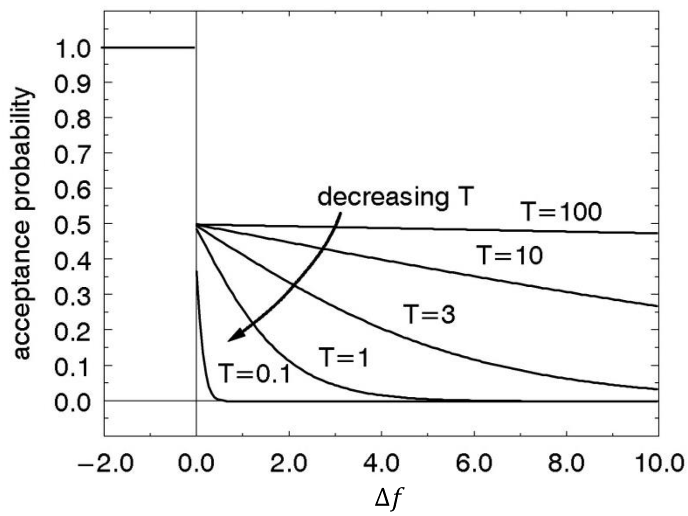
$$\min_{s \in S} \frac{1}{2} \boldsymbol{s}^{\top} \boldsymbol{W}^{(\nu)} \boldsymbol{s} + \left(\nabla f^{(\nu)}\right)^{\top} \boldsymbol{s} \qquad \text{with} \qquad S = \left\{ \boldsymbol{s} \in \mathbb{R}^n \,\middle|\, \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{p}} \boldsymbol{s} + \boldsymbol{g} = \boldsymbol{0} \right\}.$$

Simulated Annealing

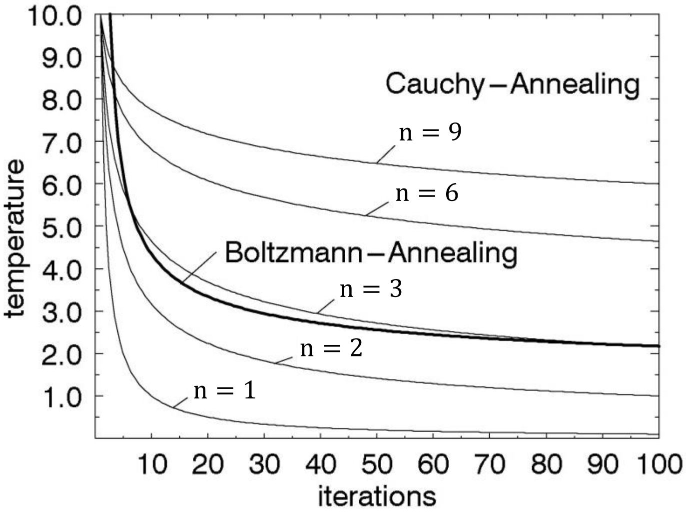
basic algorithm



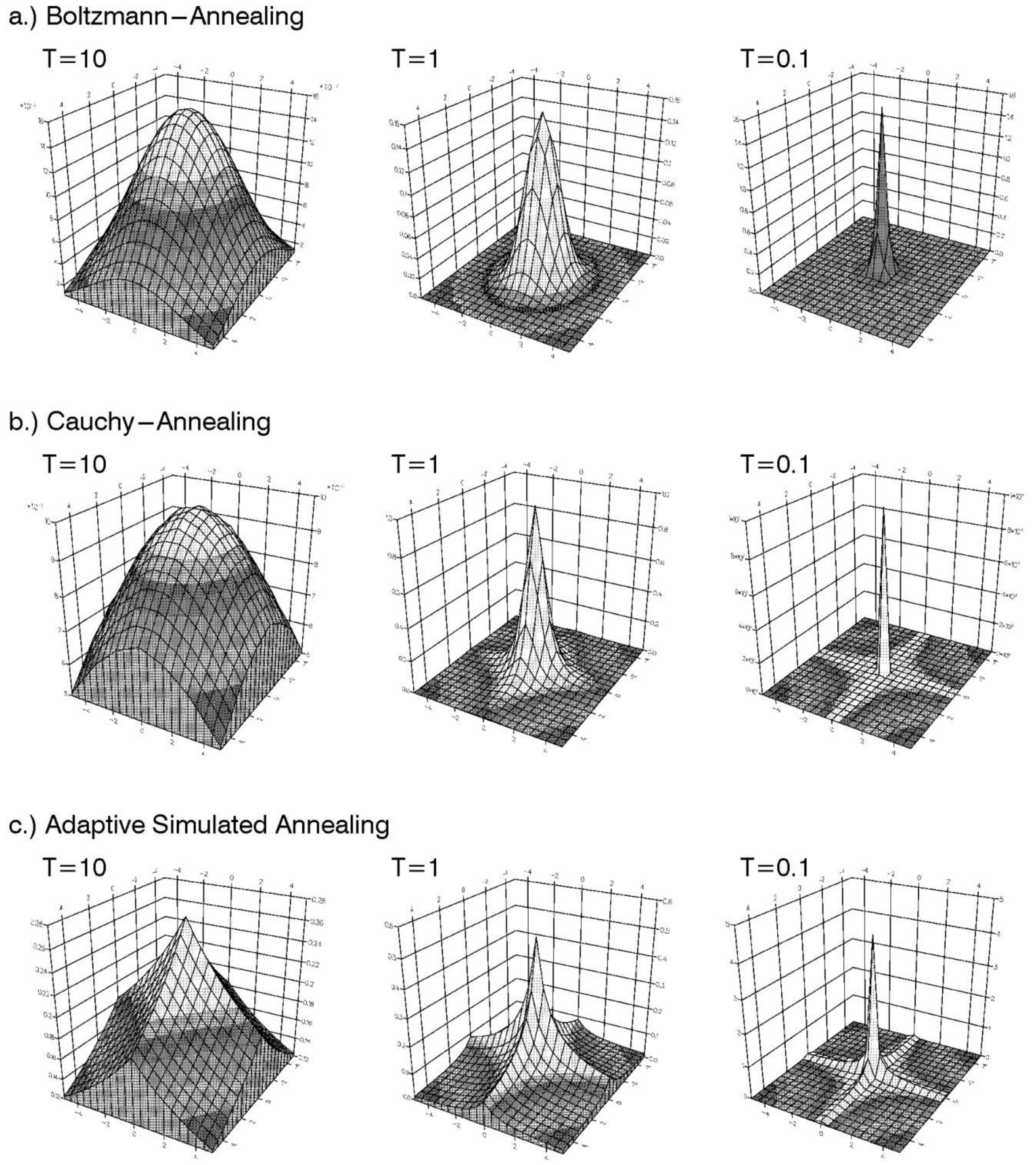
acceptance function



cooling velocity

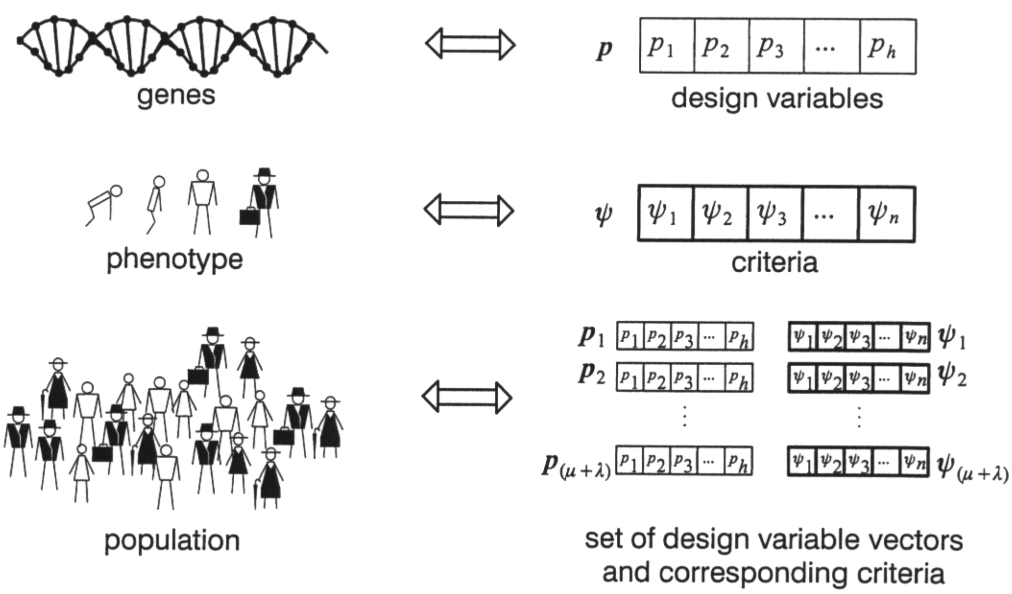


generation probability

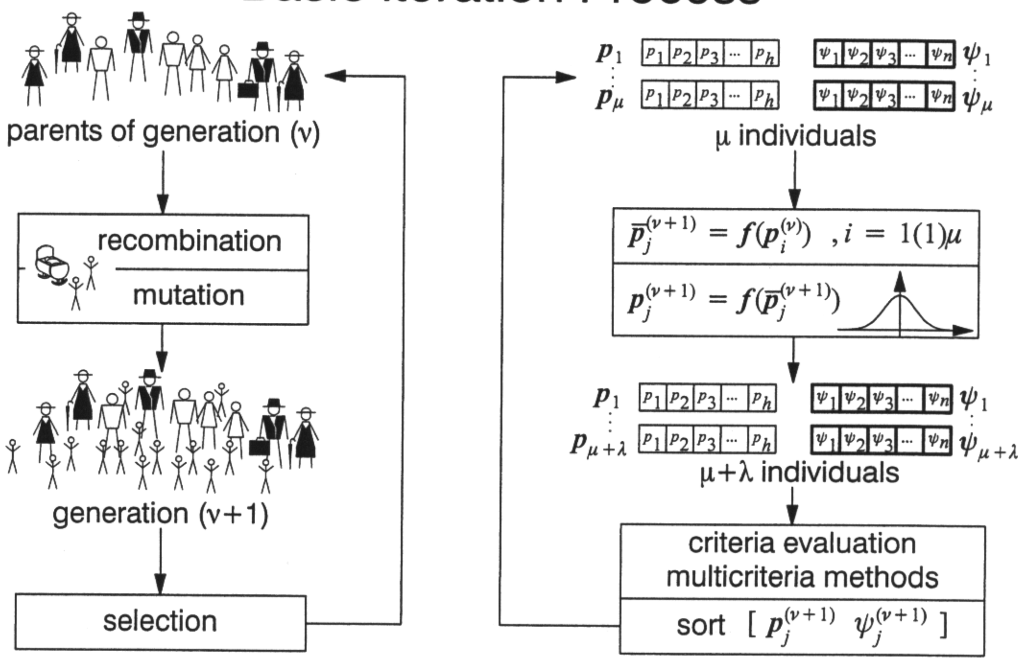


Optimization by a Stochastic Evolution Strategy

Analogy to Biological Evolution



Basic Iteration Process



from: P. Eberhard, F. Dignath, L. Kübler: Parallel Evolutionary Optimization of Multibody Systems with Application to Railway Dynamics, Vol. 9, No 2, 2003, pp. 143–164.

Particle Swarm Optimization

simulation of social behavior of bird flock (introduced by Kennedy & Eberhart in 1995)

recursive update equation

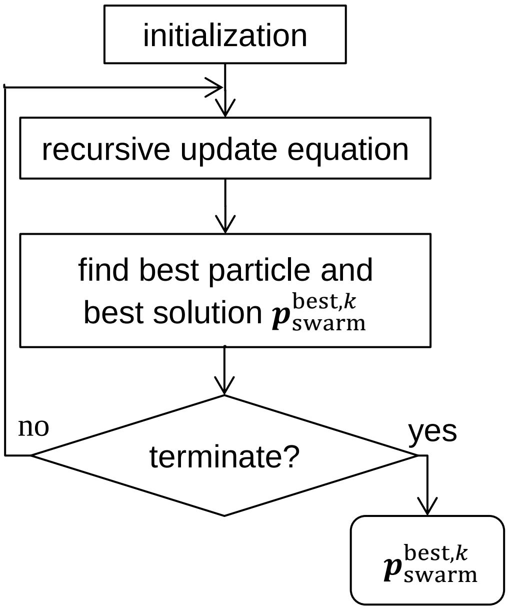
$$\boldsymbol{p}_i^{k+1} = \boldsymbol{p}_i^k + \Delta\boldsymbol{p}_i^{k+1}$$

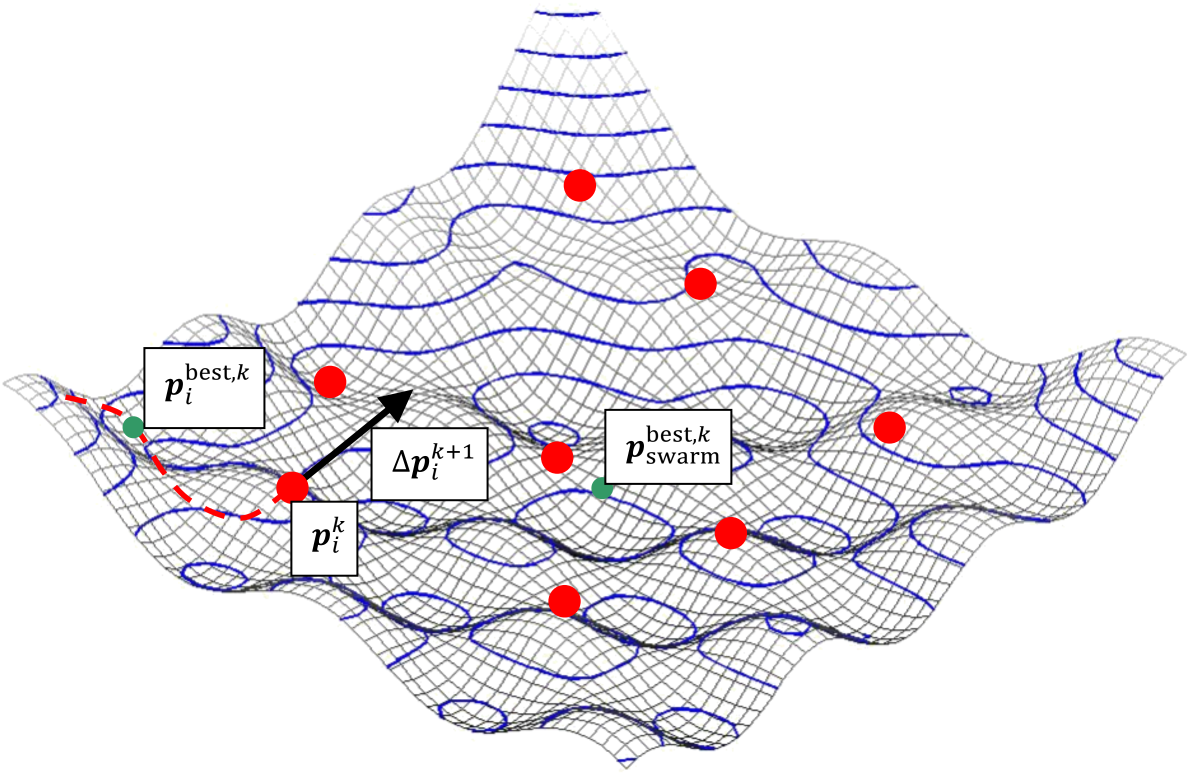
$$\Delta\boldsymbol{p}_i^{k+1} = w\Delta\boldsymbol{p}_i^k + c_1 r_{1,i}^k\left(\boldsymbol{p}_i^{\text{best},k} - \boldsymbol{p}_i^k\right) + c_2 r_{2,i}^k\left(\boldsymbol{p}_{\text{swarm}}^{\text{best},k} - \boldsymbol{p}_i^k\right)$$

tradition/ inertia — learning — social behavior

$\boldsymbol{p}_i^k$	position of particle i at time k
$\Delta\boldsymbol{p}_i^k$	velocity of particle i at time k
$r_1, r_2 \in U[0,1]$	evenly distributed numbers
$w,\ c_1,\ c_2$	control parameters

algorithm





Principles of Reduction in Multicriteria Optimization