Introduction to Optimization

Randomized Search Heuristics + Introduction to Continuous Optimization I

November 25, 2016

École Centrale Paris, Châtenay-Malabry, France

Dimo Brockhoff

INRIA Saclay – Ile-de-France

Course Overview

Date		Topic
Fri, 7.10.2016		Introduction
Fri, 28.10.2016	D	Introduction to Discrete Optimization + Greedy algorithms I
Fri, 4.11.2016	D	Greedy algorithms II + Branch and bound
Fri, 18.11.2016	D	Dynamic programming
Mon, 21.11.2016 in S103-S105	D	Approximation algorithms ~~and heuristics~~
Fri, 25.11.2016 in S103-S105	C	Randomized Search Heuristics + Introduction to Continuous Optimization I
Mon, 28.11.2016	C	Introduction to Continuous Optimization II
Mon, 5.12.2016	C	Gradient-based Algorithms
Fri, 9.12.2016	C	Stochastic Optimization and Derivative Free Optimization I
Mon, 12.12.2016	C	Stochastic Optimization and Derivative Free Optimization II
Fri, 16.12.2016	C	Benchmarking Optimizers with the COCO platform
Wed, 4.1.2017		Exam

all classes last 3h15 and take place in S115-S117 (see exceptions)

Overview of Today's Lecture

1. finish discrete optimization part with topic "Randomized Search Heuristics"
   - Randomized Local Search (RLS)
   - Variable Neighborhood Search (VNS)
   - Tabu Search (TS)
   - Evolutionary Algorithms (EAs)

2. present possible Master's and PhD thesis topics

3. start of "Introduction to Continuous Optimization I"

❶ (Randomized) Search Heuristics

Motivation General Search Heuristics

- often, problem complicated and not much time available to develop a problem-specific algorithm
- search heuristics are a good choice:
  - relatively easy to implement
  - easy to adapt/change/improve
    - e.g. when the problem formulation changes in an early product design phase
    - or when slightly different problems need to be solved over time
- search heuristics are also often "any-time", i.e. give a feasible solution early on which is then improved throughout the algorithm run → might be important in practice

Neighborhoods

For most (stochastic) search heuristics in discrete domain, we need to define a *neighborhood structure*

- which search points are close to each other?

Example: k-bit flip / Hamming distance k neighborhood

- search space: bitstrings of length n ($\Omega=\{0,1\}^n$)
- two search points are neighbors if their Hamming distance is k
- in other words: x and y are neighbors if we can flip exactly k bits in x to obtain y
- 0001001101 is neighbor of
  0001000101 for k=1
  0101000101 for k=2
  1101000101 for k=3

Neighborhoods II

Example: possible neighborhoods for the knapsack problem

- search space again bitstrings of length n ($\Omega=\{0,1\}^n$)
- Hamming distance 1 neighborhood:
  - add an item or remove it from the packing
- replacing 2 items neighborhood:
  - replace one chosen item with an unchosen one
  - makes only sense in combination with other neighborhoods because the number of items stays constant
- Hamming distance 2 neighborhood on the contrary:
  - allows to change 2 arbitrary items, e.g.
    - add 2 new items
    - remove 2 chosen items
    - or replace one chosen item with an unchosen one

Randomized Local Search (RLS)

Idea behind (Randomized) Local Search:

- explore the local neighborhood of the current solution (randomly)

Pure Random Search:

- go to randomly chosen neighbor (not dependent on obj. function)

First Improvement Local Search, Randomized Local Search (RLS):

- go to first (randomly) chosen neighbor which is better

Best Improvement Strategy:

- always go to the best neighbor
- not random anymore
- computationally expensive if neighborhood large

Variable Neighborhood Search

Main Idea: [Mladenovic and P. Hansen, 1997]

- change the neighborhood from time to time
  - local optima are not the same for different neighborhood operators
  - but often close to each other
  - global optimum is local optimum for all neighborhoods
- rather a framework than a concrete algorithm
  - e.g. deterministic and stochastic neighborhood changes
- typically combined with (i) first improvement, (ii) a random order in which the neighbors are visited and (iii) restarts

N. Mladenovic and P. Hansen (1997). "Variable neighborhood search". Computers and Operations Research 24 (11): 1097–1100.

Tabu Search

Disadvantages of local searches (with or without varying neighborhoods)

- they get stuck in local optima
- have problems to traverse large plateaus of equal objective function value (“random walk”)

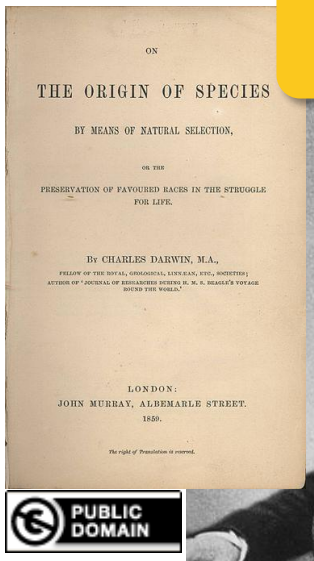
Tabu search addresses these by

- allowing worsening moves if all neighbors are explored
- introducing a tabu list of temporarily not allowed moves
- those restricted moves are
  - problem-specific and
  - can be specific solutions or not permitted “search directions” such as “don’t include this edge anymore” or “do not flip this specific bit”
- the tabu list is typically restricted in size and after a while, restricted moves are permitted again

Stochastic Optimization Algorithms

One class of (bio-inspired) stochastic optimization algorithms: Evolutionary Algorithms (EAs)

1859


ON

THE ORIGIN OF SPECIES

BY MEANS OF NATURAL SELECTION,

OR THE

PRESERVATION OF FAVOURED RACES IN THE STRUGGLE FOR LIFE.

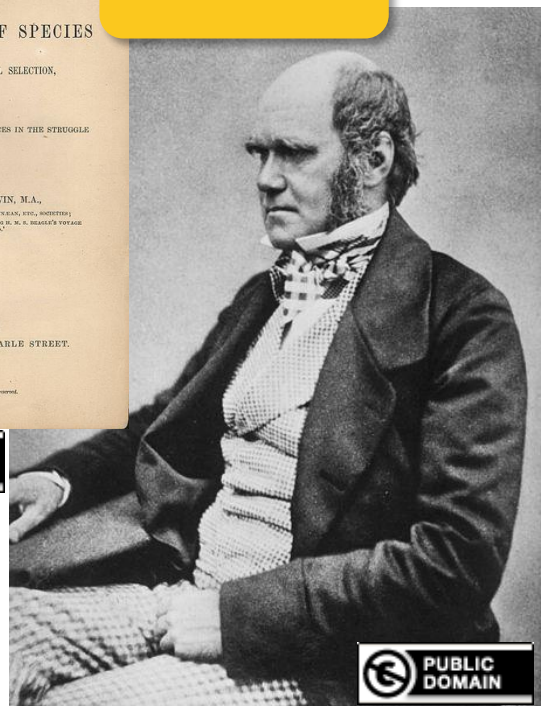
By CHARLES DARWIN, M.A.,

FELLOW OF THE ROYAL, GEOLOGICAL, LINNEAN, ETC., SOCIETIES;

AUTHOR OF 'JOURNAL OF RESEARCHES DURING H. M. S. BEAGLE'S VOYAGE ROUND THE WORLD.'

LONDON:
JOHN MURRAY, ALBEMARLE STREET.
1859.

The right of Translation is reserved.

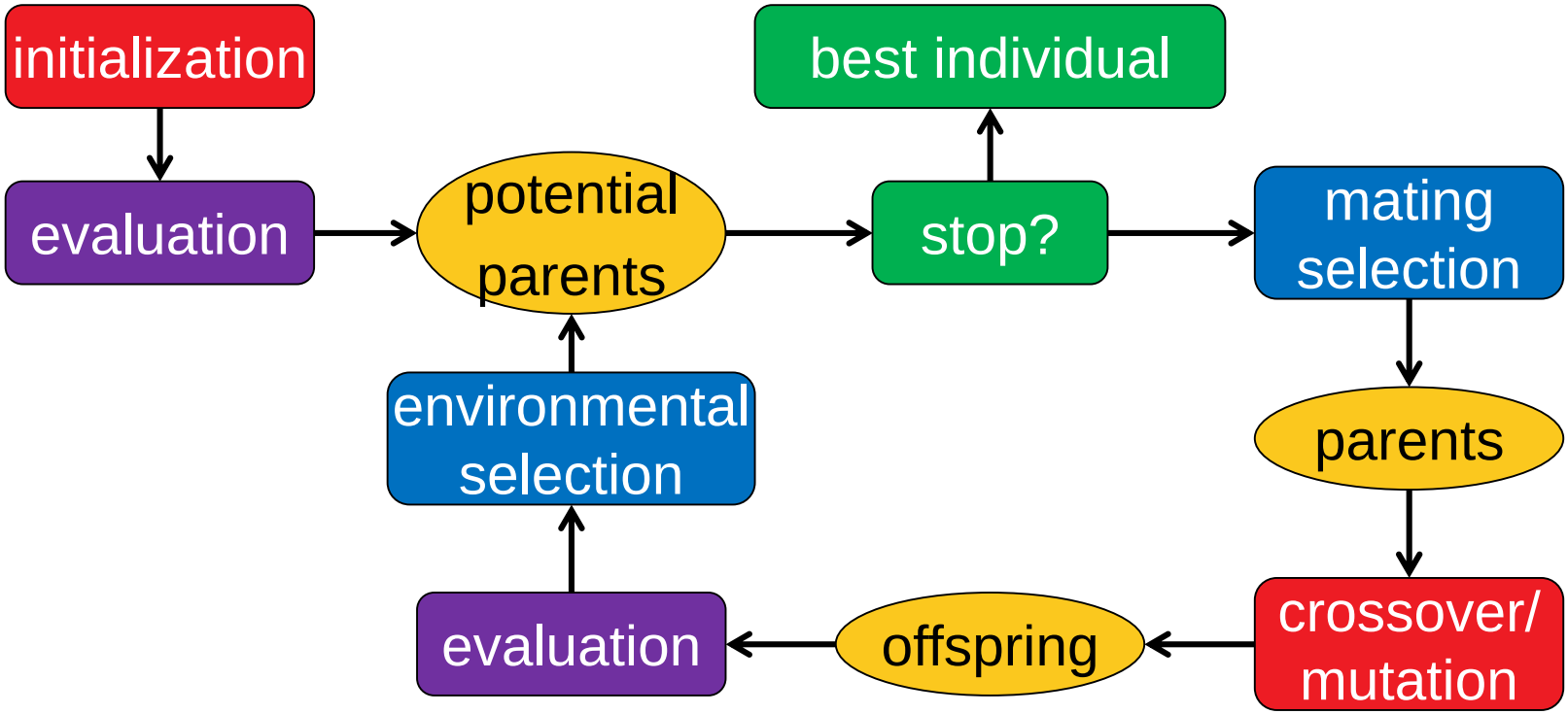


- Class of optimization algorithms originally inspired by the idea of biological evolution
- selection, mutation, recombination

Metaphors

Classical Optimization	Evolutionary Computation
variables or parameters	variables or chromosomes
candidate solution vector of decision variables / design variables / object variables	individual, offspring, parent
set of candidate solutions	population
objective function loss function cost function error function	fitness function
iteration	generation

Generic Framework of an EA



stochastic operators

"Darwinism"

stopping criteria

Important:
representation (search space)

The Historic Roots of EAs

Genetic Algorithms (GA)

J. Holland 1975 and D. Goldberg (USA)

$$\Omega = \{0,1\}^n$$

Evolution Strategies (ES)

I. Rechenberg and H.P. Schwefel, 1965 (Berlin)

$$\Omega = \mathbb{R}^n$$

Evolutionary Programming (EP)

L.J. Fogel 1966 (USA)

$$\Omega = \mathbb{R}^n$$

Genetic Programming (GP)

J. Koza 1990 (USA)

$$\Omega = \text{space of all programs}$$

nowadays one umbrella term: evolutionary algorithms

Genotype – Phenotype mapping

The genotype – phenotype mapping

- related to the question: how to come up with a fitness of each individual from the representation?
- related to DNA vs. actual animal (which then has a fitness)

Fitness of an individual not always = f(x)

- include constraints
- include diversity
- others
- but needed: always a total order on the solutions

Handling Constraints

Several possible ways to handle constraints, e.g.:

- resampling until a new feasible point is found (“often bad idea”)
- penalty function approach: add constraint violation term (potentially scaled, see also the Lagrangian in the continuous part of the lecture)
- repair approach: after generation of a new point, repair it (e.g. with a heuristic) to become feasible again if infeasible
  - continue to use repaired solution in the population or
  - use repaired solution only for the evaluation?
- multiobjective approach: keep objective function and constraint functions separate and try to optimize all of them in parallel
- some more...

Examples for some EA parts
(for discrete search spaces)

Selection

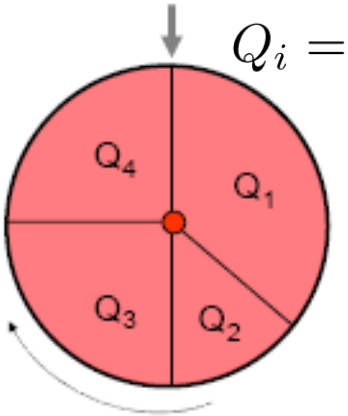
Selection is the major determinant for specifying the trade-off between exploitation and exploration

Selection is either

stochastic or deterministic

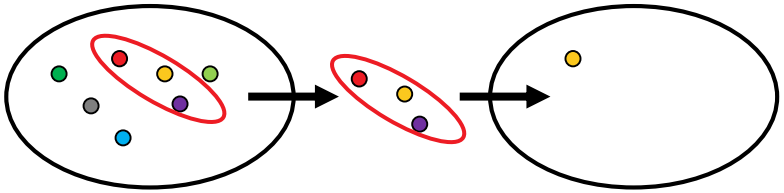
e.g. fitness proportional

$$Q_i = \frac{f(x_i)}{\sum_{j=1}^{\mu} f(x_j)}$$



Disadvantage: depends on scaling of f

e.g. via a tournament



e.g. (μ+λ), (μ,λ)

best μ from offspring *and* parents

best μ from offspring only

Mating selection (selection for variation): usually stochastic

Environmental selection (selection for survival): often deterministic

Variation Operators

Variation aims at generating new individuals on the basis of those individuals selected for mating

Variation = Mutation and Recombination/Crossover

mutation: *mut:* $\Omega \rightarrow \Omega$

recombination: *recomb:* $\Omega^r \rightarrow \Omega^s$ where $r \geq 2$ and $s \geq 1$

- choice always depends on the problem and the chosen representation
- however, there are some operators that are applicable to a wide range of problems and tailored to standard representations such as vectors, permutations, trees, etc.

Variation Operators: Guidelines

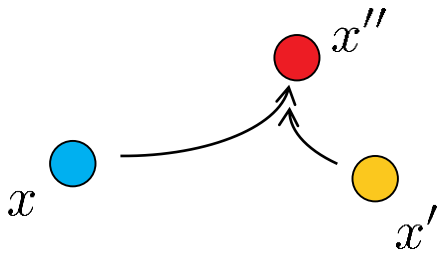
Two desirable properties for mutation operators:

- "exhaustiveness": every solution can be generated from every other with a probability greater than 0
- "locality":

$$d(x, x') < d(x, x'') => Prob(\text{mut}(x) = x') > Prob(\text{mut}(x) = x'')$$

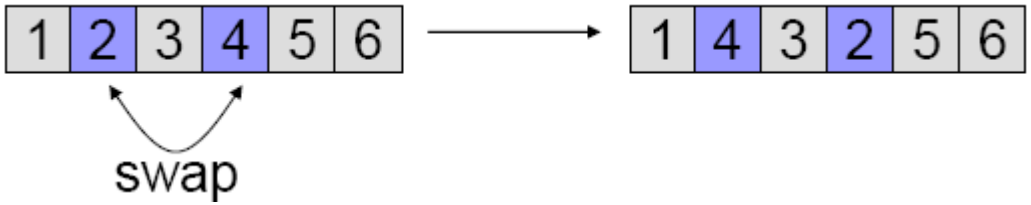
Desirable property of recombination operators ("in-between-ness"):

$$x'' = \text{recomb}(x, x') \Rightarrow d(x'', x) \leq d(x, x') \wedge d(x'', x') \leq d(x, x')$$

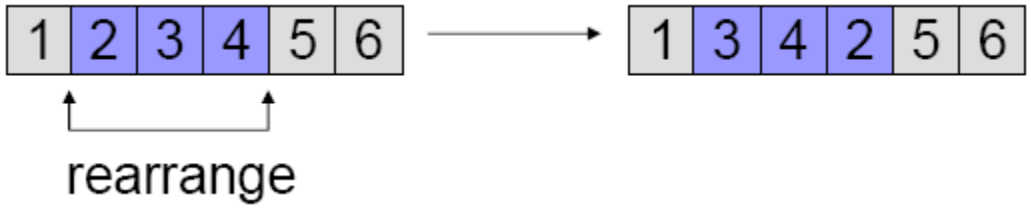


Examples of Mutation Operators on Permutations

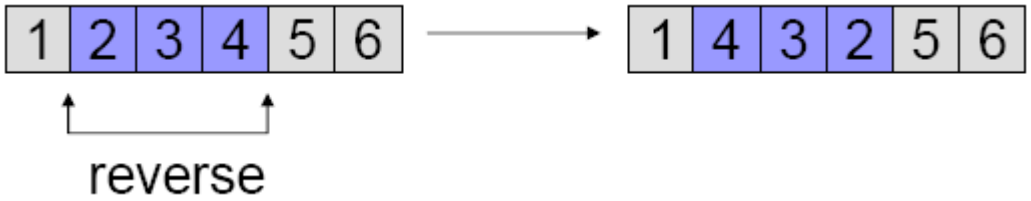
Swap:



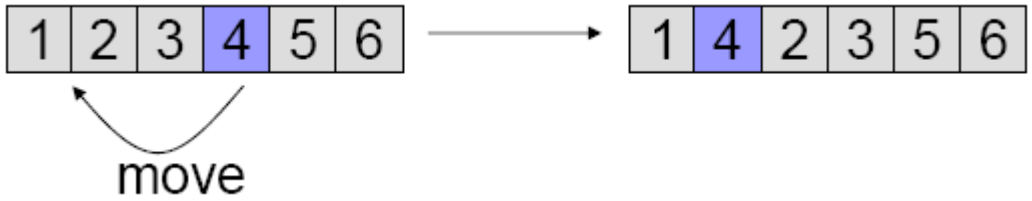
Scramble:



Invert:

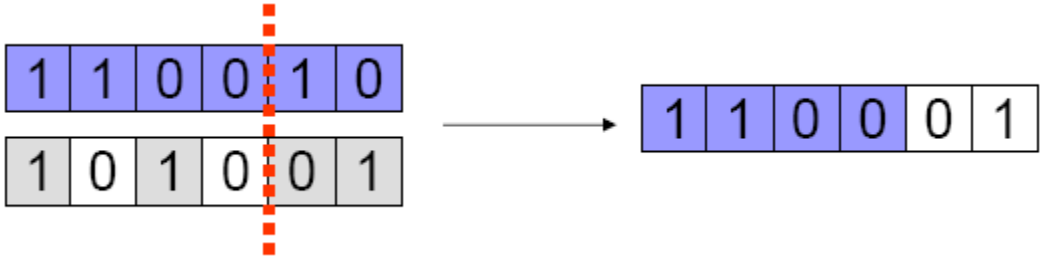


Insert:

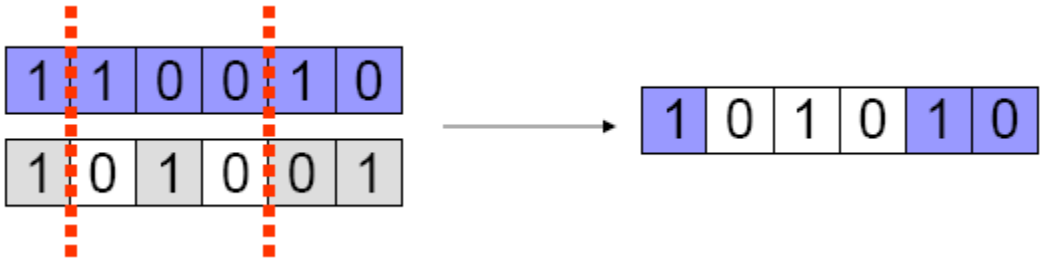


Examples of Recombination Operators: $\{0,1\}^n$

1-point crossover



n-point crossover



uniform crossover

choose each bit independently from one parent or another

Exercise: Mutation on Bitstrings

Question:

What kind of mutation operators can you imagine on the search space of all bitstrings of length n?

- keep in mind exhaustiveness and locality!

Exercise: Mutation on Bitstrings

Question:

What kind of mutation operators can you imagine on the search space of all bitstrings of length n?

- keep in mind exhaustiveness and locality!

Possible Answers:

- randomly flip a single bit (local but not exhaustive)
- randomly choose a number k of bits from 1 to n, then flip k randomly chosen bits
  - operator exhaustive but not always local:
    - not local if choice of k is uniform
    - hence, choose smaller k's with larger probability
- standard bit flip mutation: flip each bit independently with probability 1/n
  - exhaustive and local

A Canonical Genetic Algorithm

- search space of all binary strings of length n, maximization
- uniform initialization
- generational cycle of the population:
  - evaluation of solutions
  - mating selection (e.g. roulette wheel)
  - crossover (e.g. 1-point)
  - environmental selection (e.g. plus-selection)

Conclusions

- EAs are generic algorithms (randomized search heuristics, meta-heuristics, ...) for black box optimization

  no or almost no assumptions on the objective function

- They are typically less efficient than problem-specific (exact) algorithms (in terms of #funevals)

  not the case in the continuous case (we will see later)

- Allow for an easy and rapid implementation and therefore to find good solutions fast

  easy to incorporate problem-specific knowledge to improve the algorithm

Conclusions

I hope it became clear...

...that approximation algorithms are often what we can hope for in practice (might be difficult to achieve guarantees though)

...that heuristics is what we typically can afford in practice (no guarantees and no proofs)

❷ Potential Master's/PhD thesis projects

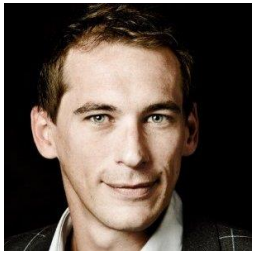
The RandOpt Team @ Inria/Ecole Polytechnique

Randopt = randomized/stochastic optimization

Anne Auger, CR1, HDR
team leader
single-obj. opt., theory, algo. design, applications

Asma Atamna
PhD student

Dimo Brockhoff, CR1
multiobjective opt., algo. design, theory



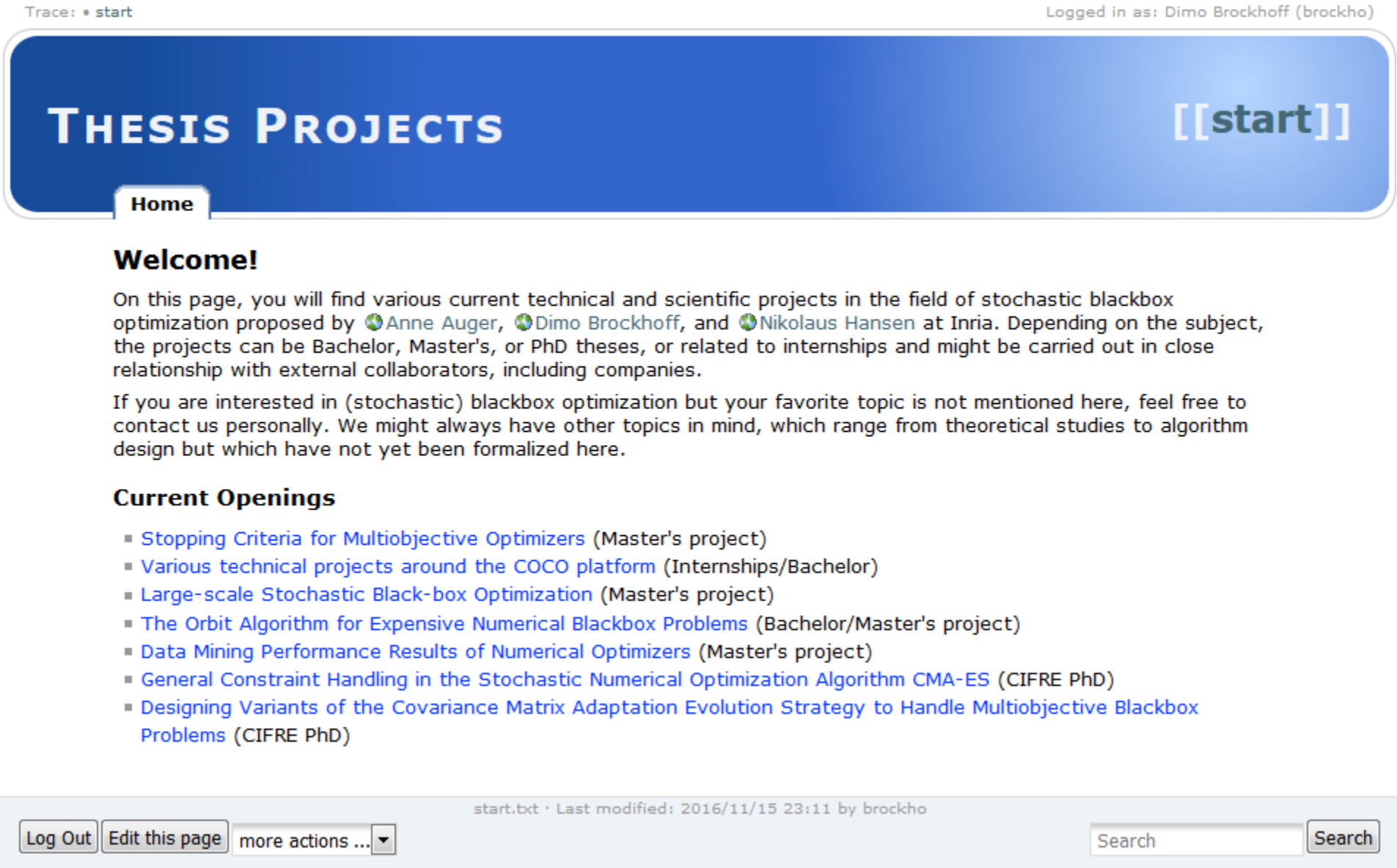
Adrien Renaud
engineer

Nikolaus Hansen, DR2, HDR
single-obj. opt., algo design, applications, theory

Dejan Tušar
engineer

Potential Research Topics for Master's/PhD Theses

http://randopt.gforge.inria.fr/thesisprojects/



Potential Research Topics for Master's/PhD Theses

More projects without the involvement of companies:

- stopping criteria in multiobjective optimization
- large-scale variants of CMA-ES
- algorithms for expensive optimization based on CMA-ES

all above: relatively flexible between theoretical (e.g. proofs of convergence) and practical projects

Coco-related:

- implementing and benchmarking algorithms for expensive opt.
- data mining performance results

not all subjects online yet:
please contact us if you are interested!

❸ Introduction to Continuous Optimization

Overview Continuous Optimization Part

Introduction to Continuous Optimization

- examples (from ML / black-box problems)
- typical difficulties in optimization (e.g. constraints)

Mathematical Tools to Characterize Optima

- reminders about differentiability, gradient, Hessian matrix
- unconstraint optimization
  - first and second order conditions
  - convexity
- constrained optimization

Gradient-based Algorithms

- quasi-Newton method (BFGS)

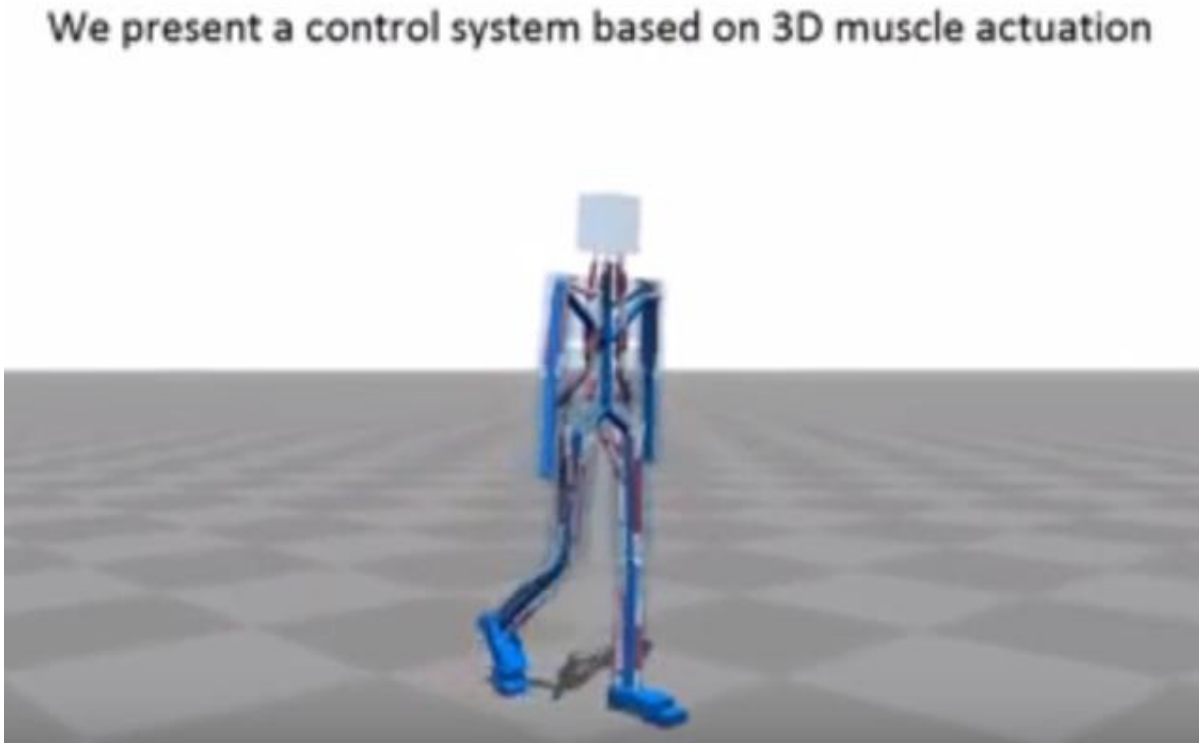
Learning in Optimization / Stochastic Optimization

- stochastic adaptive algorithms (CMA-ES)

Benchmarking Numerical Blackbox Optimizers

First Example of a Continuous Optimization Problem

Computer simulation teaches itself to walk upright (virtual robots (of different shapes) learning to walk, through stochastic optimization (CMA-ES)), by Utrecht University:



https://www.youtube.com/watch?v=yci5FuI1ovk

T. Geitjtenbeek, M. Van de Panne, F. Van der Stappen: "Flexible Muscle-Based Locomotion for Bipedal Creatures", SIGGRAPH Asia, 2013.

Continuous Optimization

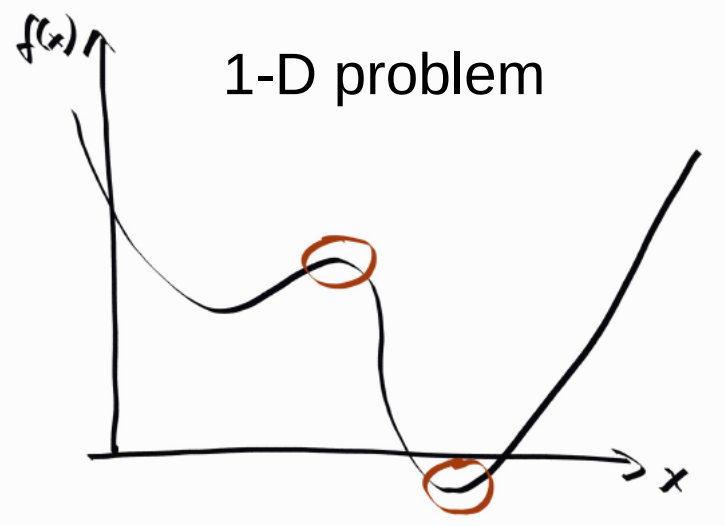
- Optimize f: $\begin{cases} \Omega \subset \mathbb{R}^n \to \mathbb{R} \\ x = (x_1, \dots, x_n) \to f(x_1, \dots, x_n) \end{cases}$

  $\in \mathbb{R}$

  unconstrained optimization

- Search space is continuous, i.e. composed of real vectors $x \in \mathbb{R}^n$

- $n = \begin{cases} \text{dimension of the problem} \\ \text{dimension of the search space } \mathbb{R}^n \text{ (as vector space)} \end{cases}$



1-D problem

2-D level sets

Unconstrained vs. Constrained Optimization

Unconstrained optimization

$$\inf\{f(x) \mid x \in \mathbb{R}^n\}$$

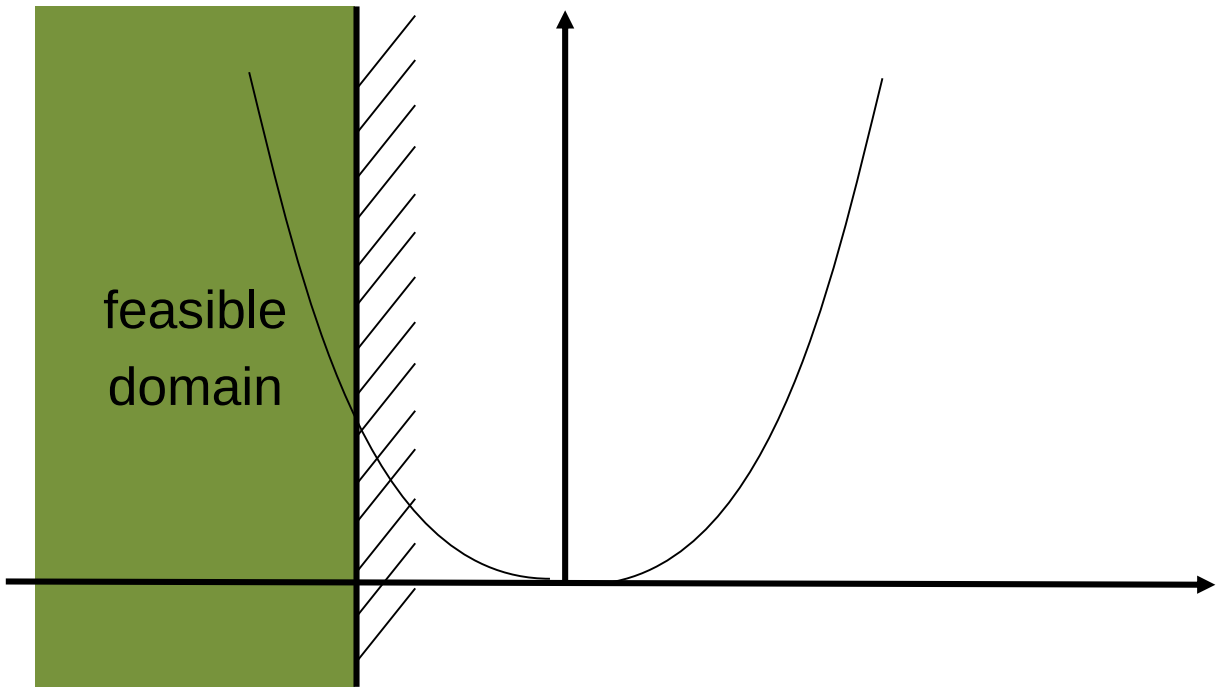
Constrained optimization

- Equality constraints: $\inf\{f(x) \mid x \in \mathbb{R}^n, g_k(x) = 0, 1 \leq k \leq p\}$
- Inequality constraints: $\inf\{f(x) \mid x \in \mathbb{R}^n, g_k(x) \leq 0, 1 \leq k \leq p\}$

where always g_k: $\mathbb{R}^n \to \mathbb{R}$

Example of a Constraint

$$\min_{x \in \mathbb{R}} f(x) = x^2 \text{ such that } x \leq -1$$



Ana,lytical Functions

Example: 1-D

$$f_1(x) = a(x - x_0)^2 + b$$

where $x, x_0, b \in \mathbb{R}, a \in \mathbb{R}$

Generalization:

convex quadratic function

$$f_2(x) = (x - x_0)^T A\,(x - x_0) + b$$

where $x, x_0 \in \mathbb{R}^n, b \in \mathbb{R}, A \in \mathbb{R}^{\{n \times n\}}$

and A symmetric positive definite (SPD)

Exercise:
What is the minimum of $f_2(x)$?

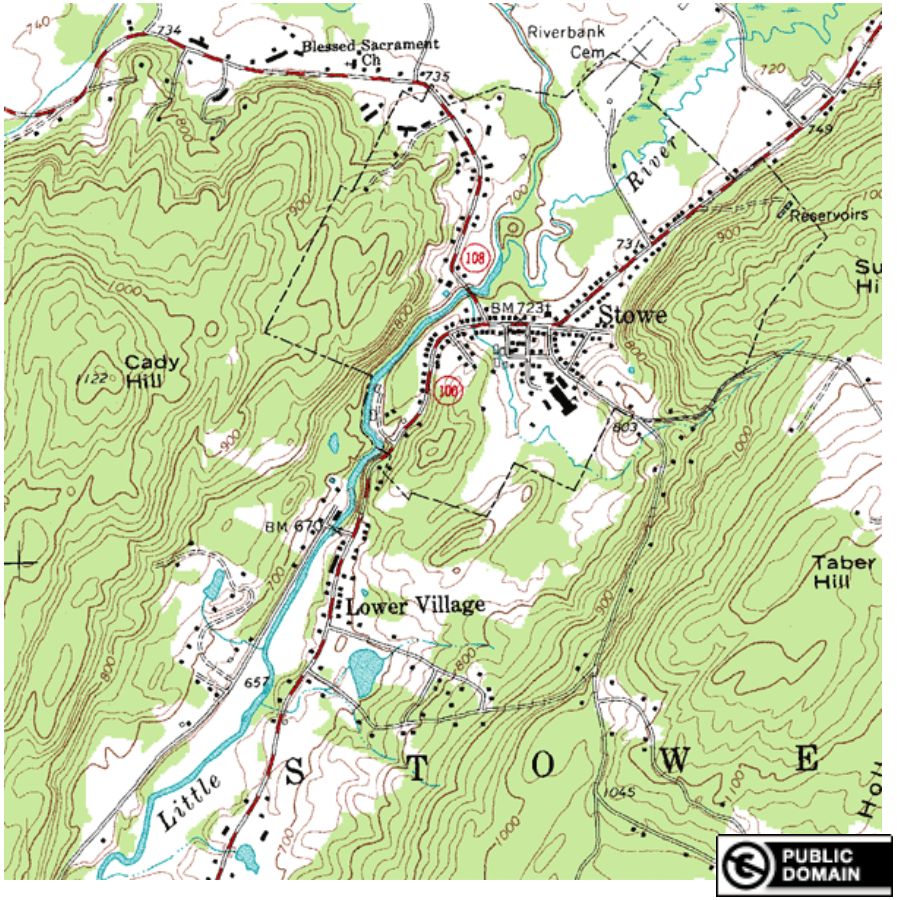
Levels Sets of Convex Quadratic Functions

Continuation of exercise:
What are the level sets of f_2?

Reminder: level sets of a function

$$L_c = \{x \in \mathbb{R}^n \mid f(x) = c\}$$

(similar to topography lines / level sets on a map)



Levels Sets of Convex Quadratic Functions

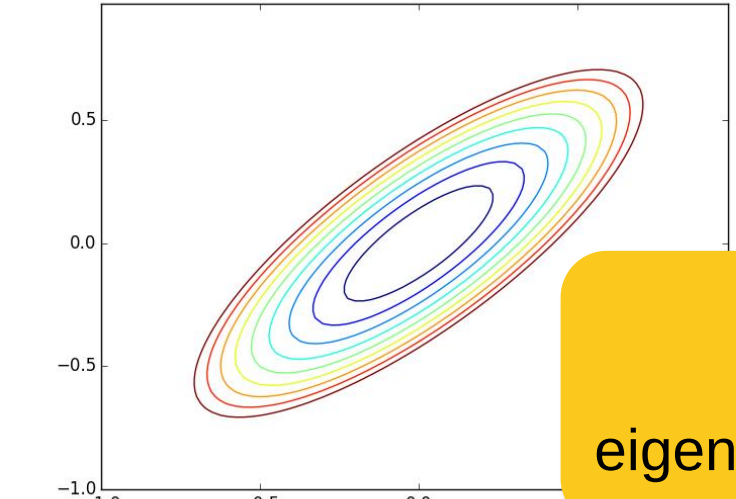
Continuation of exercise:
What are the level sets of f_2?

- Probably too complicated in general, thus an example here
- Consider $A = \begin{pmatrix} 9 & 0 \\ 0 & 1 \end{pmatrix}, b = 0, n = 2$
  a) Compute $f_2(x)$.
  b) Plot the level sets of $f_2(x)$.
  c) Optional: More generally, for $n = 2$, if A is SPD with eigenvalues $\lambda_1 = 9$ and $\lambda_2 = 1$, what are the level sets of $f_2(x)$?

Answer for c)

The general case of A being symmetric positive definite:

- level sets are ellipsoids as well, rotated and squeezed according to entries in A
- more precisely:
  - axes of ellipsoid are the eigenvectors of A
  - scaling is given by the eigenvalues



$$A = \begin{pmatrix} +5 & -4 \\ -4 & +5 \end{pmatrix}$$

eigenvalues: 9 and 1 (axis ratio: 3)