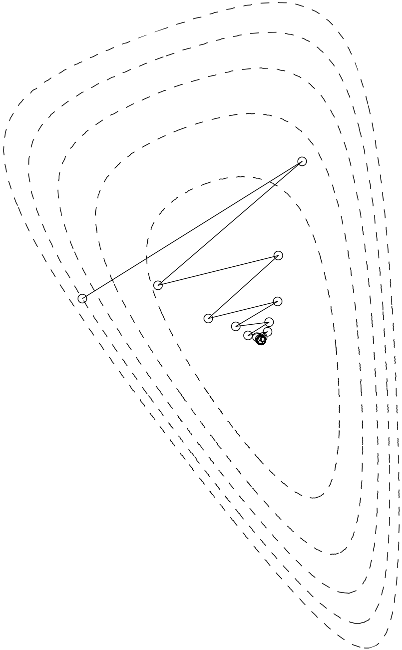


Optimization Algorithms

Introduction

Marc Toussaint
Technical University of Berlin
Winter 2024/25

Learning & Intelligent Systems Lab

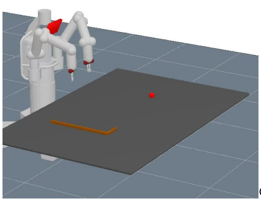
argmin.lis.tu-berlin.de

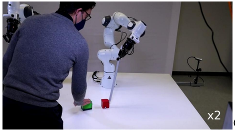
Research

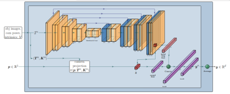
- Intersection of AI & Robotics
- Combining learning and reasoning
- physical reasoning, task-and-motion planning (logic-geometric programming)
- reinforcement learning, perception-based policies, reactive control/learning
- driven by robotics problems

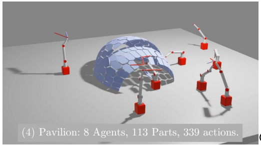
Applications

time ~2/70









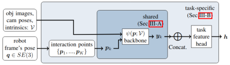
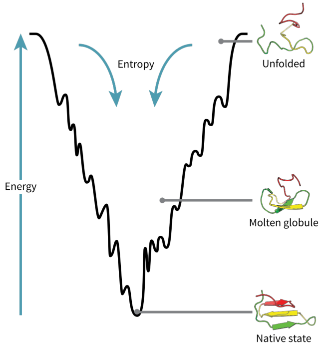


Fig. 2: The interaction feature prediction scheme of DVC

Why is Optimization interesting?

$$\delta \int_{t_0}^{t_1} L(q(t), \dot{q}(t), t) \; dt = 0$$

Principle of Least Action



protein folding

$$\min_\beta \|\beta\|^2 \quad \text{s.t.} \quad y_i(\phi(x_i)^\top \beta) \geq 1,\ i = 1, \dots, n$$

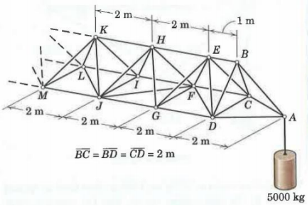
support vector machine

$$\min_{f \in \mathcal{H}} \sum_{i=1}^{n} \ell(f(x_i), y_i)$$

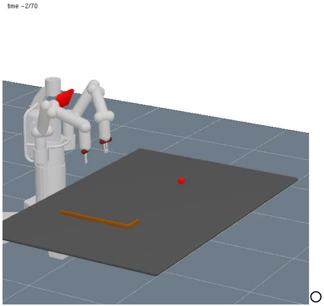
loss minimization (e.g., NNs)

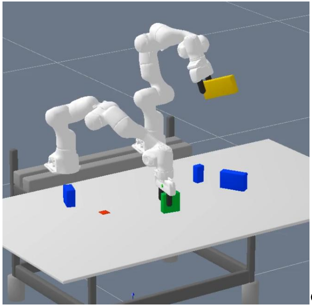
$$\min_{u,x} \int_0^T f(x(t), u(t))\, dt \quad \text{s.t.} \quad \dot{x} = f(x,u),\ g(x(t)) \leq 0,\ h(x(T)) = 0$$

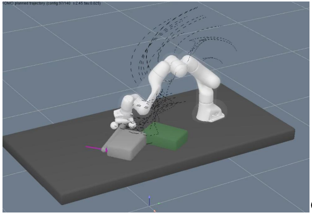
optimal control

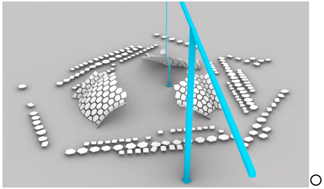


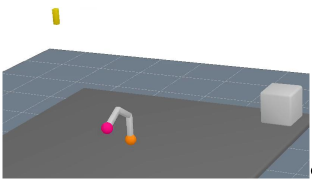
construction statics

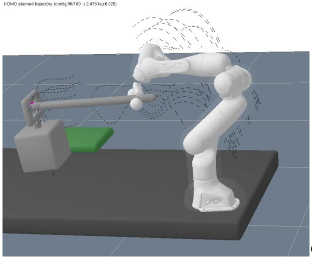












“Logic Geometric Programming”

Why is Optimization interesting?

- *Optimality principles are a means of scientific and engineering description*
- It is often easier to describe a thing (natural or artifical) via an optimality priciple than directly
- Almost any scientific field uses optimality principles to describe nature & artifacts
  - Physics, Chemistry, Biology, Mechanics, ...
  - Operations research, scheduling, ...
  - Computer Vision, Speach Recognition, Machine Learning, Robotics, AI, ...
- Endless applications

Teaching optimization

- Optimization includes largely different approaches/formalisms:
  - Standard continuous, convex or non-linear optimization
  - Discrete Optimization
  - Global Optimization
  - Stochastic Optimization, Evolutionary Algorithms, Swarm optimization, etc

Teaching optimization

- Optimization includes largely different approaches/formalisms:
  - Standard continuous, convex or non-linear optimization
  - Discrete Optimization
  - Global Optimization
  - Stochastic Optimization, Evolutionary Algorithms, Swarm optimization, etc

- This lecture focusses on continuous, convex or non-linear optimization

Optimization Problems

- Generic optimization problem, also called **Mathematical Program**:

  Let $x \in \mathbb{R}^n,\ f:\ \mathbb{R}^n \to \mathbb{R},\ g:\ \mathbb{R}^n \to \mathbb{R}^m,\ h: \mathbb{R}^n \to \mathbb{R}^l$. Find

$$\min_x f(x) \ \text{ s.t. } \ g(x) \le 0,\ h(x) = 0$$

Optimization Problems

- Generic optimization problem, also called **Mathematical Program**:

  Let $x \in \mathbb{R}^n,\ f:\ \mathbb{R}^n \to \mathbb{R},\ g:\ \mathbb{R}^n \to \mathbb{R}^m,\ h: \mathbb{R}^n \to \mathbb{R}^l$. Find

$$\min_x f(x) \ \text{ s.t. }\ g(x) \le 0,\ h(x) = 0$$

- **Blackbox**: only $f(x)$ can be evaluated
- **Gradient**: $\nabla f(x)$ can be evaluated
- Gauss-Newton type: $f(x) = \phi(x)^\top \phi(x)$ and $\nabla\phi(x)$ can be evaluated
- **Stochastic Gradient**: only "samples of $\nabla f(x)$" can be evaluated efficiently
- **2nd order**: $\nabla^2 f(x)$ can be evaluated
- Approximate methods:
  - Use samples of $f(x)$ to approximate $\nabla f(x)$ locally
  - Use samples of $\nabla f(x)$ to approximate $\nabla^2 f(x)$ locally

Planned Outline

- Part 1: Downhill algorithms for unconstrained optimization:
  - gradient descent, backtracking line search, Wolfe conditions, convergence properties
  - covariant gradients, Newton, quasi-Newton methods, (L)BFGS
- Part 2: Algorithms for constrained optimization:
  - KKT conditions, Lagrangian
  - Log-barrier, Augmented Lagrangian, primal-dual Newton
  - SQP
- Part 3: Extended topics (subject to change):
  - Stochastic gradient methods
  - Global optimization
  - stochastic search, evolutionary algorithms
  - maybe this year: ADMM & NLP Reformulations

References

- Maths for Intelligent Systems script on ISIS page
- Boyd and Vandenberghe: *Convex Optimization.*
  `http://www.stanford.edu/~boyd/cvxbook/`
- Nocedal & Wright: *Numerical Optimization*
  `www.bioinfo.org.cn/~wangchao/maa/Numerical_Optimization.pdf`

(this course won't of course cover all this - just for reference)

Organization

- 6 LPs (180h, 12h/w, 15 weeks)
- Lectures, weekly, in person
- Exercise Sheets & Coding Assignments:
  - Weekly exercise sheets, mostly analytic problems, discussed in the tutorials
  - Hand-in coding assignments, every ~ 3 weeks: Submit your optimization algorithms/problems via git $\to$ are numerically evaluated/graded

- ISIS as central webpage
- Contact:
  - Tutors: Sayantan Auddy `<auddy@tu-berlin.de>`, Cornelius Braun, Hongyou Zhou
  - Office (grades/etc): Ilaria Cicchetti-Nilsson `<office@lis.ut-berlin.de>`

Assignments & Exam

- Voting System for the exercise sheets:
  - Before attending the tutorial, students mark in an ISIS questionnaire which exercises they have worked on
  - Students are randomly selected to present their solutions (no need for correct solutions – just something to present and discuss)
  - When not attending: upload pdf notes/solutions on ISIS

- **Exam prerequisites:**
  - at least 50% votes in the exercises, and
  - at least 50% points in the hand-in coding assignments

  (If you fulfilled these prerequisites last year, you don't have to redo them.)

- The written exam will be about analytical problems, determines final grade (no portfolio)

Registration

- Registration for the exam in Moses will open in January
- To gain the exam prerequisites you'll have to register for the coding exercises (will be organized in the second/third week), and submit your votes on the exercise sheets
- There is no further registration for this course necessary

Prerequisites

- Module description:
  - Good knowledge in linear algebra and calculus
    $\rightarrow$ Specifically, the **‘Maths for Intelligent Systems Script’** up to Chapter 3.
  - Basic programming in Python

Prerequisites

- Module description:
  - Good knowledge in linear algebra and calculus
    $\rightarrow$ Specifically, the **'Maths for Intelligent Systems Script'** up to Chapter 3.
  - Basic programming in Python

- Maths Self-Check:
  - Are you familiar with basics on functions, Jacobians, Hessian, matrix derivatives? (Sec. 2)?
  - Do you have intuition about basic linear algebra? (Fig. 2 and Sec. 3.6)
  - Can you solve basic matrix equations and derive gradients/Jacobians? (e.g. Sec 2.4)

Prerequisites

- Module description:
  - Good knowledge in linear algebra and calculus
    $\rightarrow$ Specifically, the **'Maths for Intelligent Systems Script'** up to Chapter 3.
  - Basic programming in Python

- Maths Self-Check:
  - Are you familiar with basics on functions, Jacobians, Hessian, matrix derivatives? (Sec. 2)?
  - Do you have intuition about basic linear algebra? (Fig. 2 and Sec. 3.6)
  - Can you solve basic matrix equations and derive gradients/Jacobians? (e.g. Sec 2.4)

- Coding:
  - Numeric coding in Python (numpy)
  - Familiarity with git will help

Module description (Moses 41016)

- Learning Outcomes
  - The students will be able to develop and apply optimization algorithms.
  - They can formulate real-world problems appropriately as mathematical programs.
  - They have a detailed understanding of the different categories of optimization problems, and methods to approach them.
  - They have a basic understanding of the theory behind and properties of optimization algorithms.
  - They have an overview of and experience with existing optimization software and are able to apply them to solve optimization problems.
- Content
  - The course is on continuous optimization problems, with focus on non-linear mathematical programming (constrained optimization).
  - Part 1 introduces efficient downhill algorithms in the unconstrained case: ...
  - Part 2 will introduce efficient algorithms for constrained optimization: ...
  - Part 3 will cover extended topics (global optimization, stochastic gradient, stochastich search) ...
- Prerequisites
  - Good knowledge in linear algebra and calculus
  - Basic programming knowledge in Python

Module description (Moses 41016)

- Grading
  - graded, written exam, English (120min)
- This module is used in the following module lists:
  - Computational Engineering Science (Informationstechnik im Maschinenwesen) (Master of Science)
  - Computer Engineering (Master of Science)
  - Computer Science (Informatik) (Master of Science)
  - Elektrotechnik (Master of Science)
  - ICT Innovation (Master of Science)
  - Medieninformatik (Master of Science)
  - Physikalische Ingenieurwissenschaft (Master of Science)