کنترل پیش بین
Model Predictive Control

ارائه کننده: امیرحسین نیکوفرد

مهندسی برق و کامپیوتر دانشگاه خواجه نصیر

دانشگاه صنعتی خواجه نصیرالدین طوسی

Numerical Optimization Methods

- Introduction
- Unconstrained Optimization
  - Gradient Methods
  - Newton’s Method
- Constrained Optimization
  - Gradient Methods
  - Interior Point Methods
  - Active Set Methods
- Software
  - Modeling languages for optimization problems

Introduction

In all but the simplest cases, an analytical solution to,

$$z^* \in \arg\min f(z) \quad s.t. \quad z \in S$$

cannot be obtained.

- Numerical computation of a solution that is "good enough" by

Iterative optimization methods:

Given an initial guess z^0, produce a sequence of iterates

$$z^{k+1} = \Psi(z^k, f, S), \quad k = 0, 1, \ldots, k_{max}$$

such that

$$|f(z^{k_{max}}) - f(z^*)| \leq \epsilon \quad \text{and} \quad \text{dist}(z^{k_{max}}, S) \leq \delta,$$

where ϵ and δ are user defined tolerances.

Introduction

Important aspects of optimization algorithms:

- Convergence: is k_{max} finite for every $\delta, \varepsilon > 0$?
- Convergence speed: dependence of errors $f(z^{k_{max}}) - f(z^*)$ and dist($z^{k_{max}}$, S) on iteration counter
- Feasibility: for some methods $\delta = 0$, but in general $\delta \neq 0$
- Numerical robustness in presence of finite precision arithmetics
- Warm-starting: can the method take advantage of z^0 being close to z^*?
- Preconditioning: equivalence transformation of (P) into a similar problem(P') that can be solved in fewer iterations?

Unconstrained Optimization

Unconstrained Optimization Using Gradient Information [**Cauchy 1847**]

- **Goal:** Solve the unconstrained (i.e. $S = R^n$) problem

$$\min_x f_o(x)$$

where $f : R^n \longrightarrow R$ is **convex** and continuously differentiable.

- Idea: Gradient ∇f gives direction of steepest local ascent

$\Rightarrow$ Make steps of size h^k into anti-gradient direction $-\nabla f$:

$$z^{k+1} = z^k - h^k \nabla f(z^k)$$

Unconstrained Optimization

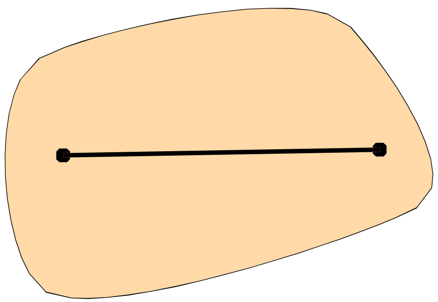
Convex set

Definition:

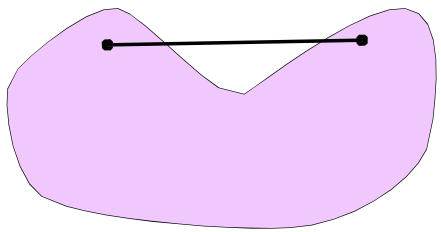
A set $S \subseteq \mathbb{R}^n$ is **convex** if for all $x_1, x_2 \in S$

$$\lambda x_1 + (1-\lambda)x_2 \in S, \ \forall \lambda \in [0, 1]$$

convex set



nonconvex set



Unconstrained Optimization

Convex Function

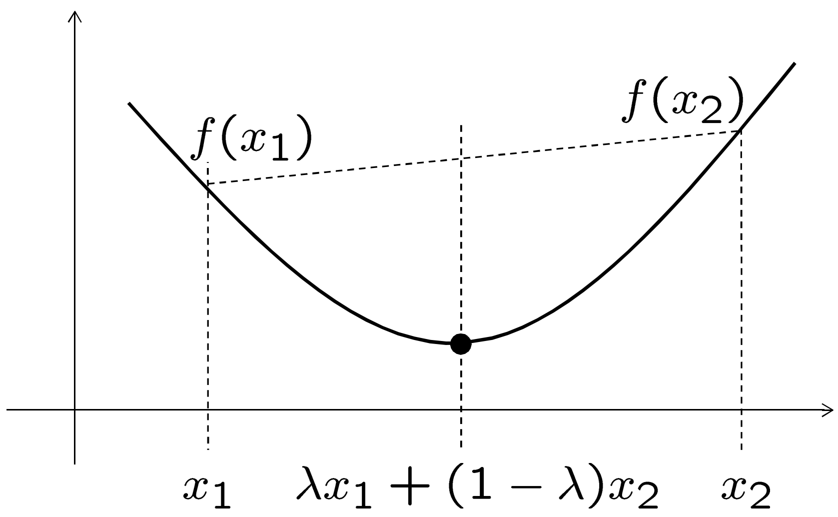
Definition:

$f : S \rightarrow R$ is a **convex function** if S is convex

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$$

$$\forall x_1, x_2 \in S, \lambda \in [0,1]$$

Jensen's inequality



Unconstrained Optimization

Aspects of Gradient Methods for unconstrained optimization

- Convergence: is k_{max} finite for every $\delta, \varepsilon > 0$? ✓(globally)
- Convergence speed: dependence of errors f $(z^{k_{max}}) - f(z^*)$ and dist($z^{k_{max}}$, S)on iteration counter ✓(globally)
- Numerical robustness in presence of finite precision arithmetics
- Warm-starting: can the method take advantage of z^0 being close to z^*? ✓
- Preconditioning: equivalence transformation of (P) into a similar problem(P') that can be solved in fewer iterations? ✓
- Each iteration computationally cheap (matrix-vector multiplication for QPs)

Unconstrained Optimization

Newton's Method

- Idea: Minimize second-order approximation of f at point z^k

$$z^{k+1} = \arg\min_z f(z^k) + \nabla f(z^k)^T(z - z^k) + \frac{1}{2}(z - z^k)^T \nabla^2 f(z^k)(z - z^k)$$

$$\nabla_z \left(f(z^k) + \nabla f(z^k)^T(z - z^k) + \frac{1}{2}(z - z^k)^T \nabla^2 f(z^k)(z - z^k) \right) \Big|_{z=z^{k+1}} = 0$$

$$\Leftrightarrow \nabla f(z^k) + \nabla^2 f(z^k)(z^{k+1} - z^k) = 0$$

$$\Leftrightarrow z^{k+1} = z^k \underbrace{- \left(\nabla^2 f(z^k)\right)^{-1} \nabla f(z^k)}_{\text{Newton direction } d_N(z^k)}$$

Since second-order approximation of f is not an upper bound on f , full Newton step does not necessarily yield descent

Unconstrained Optimization

- Idea: Use step size $h^k>0$ such that Newton step yields descent

$$z^{k+1} = z^k - h^k \left(\nabla^2 f(z^k)\right)^{-1} \nabla f(z^k)$$

- **Line search (LS) methods:**

  - *Exact*: Compute *best* h^k:

    $$h^{k^*} = \arg\min_{h>0} f(z^k + h^k d_N(z^k))$$

    Optimization in 1 variable $\to$ solve by bisection
    Time consuming (requires many evaluations of f)

  - *Inexact*: Find h^k that decreases f by some per cent. Example:
    Backtracking[1] line search. For $\alpha \in (0, 0.5)$ and $\beta \in (0, 1)$:
    Initialize $h^k = 1$.
    while $f(z^k + h^k d_N(z^k)) > f(z^k) + \alpha h^k \nabla f(z^k)^T d_N(z^k)$ **do** $h^k \leftarrow \beta h^k$

Unconstrained Optimization

Aspects of **Newton's Methods** for unconstrained optimization

- Convergence: is k_{max} finite for every $\delta, \varepsilon > 0$? ✓(globally with line search)
- Convergence speed: dependence of errors $f(z^{k_{max}}) - f(z^*)$ and dist($z^{k_{max}}$, S)on iteration counter ✓(locally quadratically converging)
- Numerical robustness in presence of finite precision arithmetics ✓
- Warm-starting: can the method take advantage of z^0 being close to z^*? ✓
- Preconditioning: equivalence transformation of (P) into a similar problem(P') that can be solved in fewer iterations? No
- Each iteration computationally expensive (requires solving a system of linear equations)

Constrained Optimization

- Constrained Optimization
  - Gradient Methods
  - Interior Point Methods
  - Active Set Methods

Constrained Optimization

Consider the following constrained convex optimization problem:

$$\min_{z} f_o(z)$$

$$\textit{subject to} \quad z \in S$$

where S is convex and $f_o(z)$ is convex

The problem has several ingredients:

- The vector z collects the decision variables (optimization variables)
- $f_o(z) \quad \mathbb{R}^n \rightarrow \mathbb{R}$ objective function
- We can solve the unconstrained problem $S \equiv R^n$ efficiently by the gradient method
- **Question**: How to handle constraints?

Constrained Optimization

A useful Reformulation of the Gradient update

- **Unconstrained case:** Gradient update results from minimizing a quadratic function:

$$z^{k+1} = z^k - h^k \nabla f(z^k)$$

- **Constrained case:** Incorporate constraints in minimization:

$$\boxed{z^{k+1} = \pi_S\left(z^k - h^k \nabla f(z^k)\right)}$$

where π_S is a **projection**:

$$\pi_S(y) \triangleq \arg\min_z \frac{1}{2}\|z - y\|_2^2 \quad \text{s.t. } z \in S$$

Constrained Optimization

➢ Interior Point Methods

Consider the following problem with inequality constraints

$$\min_z f(z)$$
$$s.t \quad g_i(z) \le 0, i = 1, \cdots, m$$

- f, g_i convex, twice continuously differentiable
- We assume $f(z^*)$ is finite and attained
- We assume problem is strictly feasible: there exists a $\tilde{z}$ with

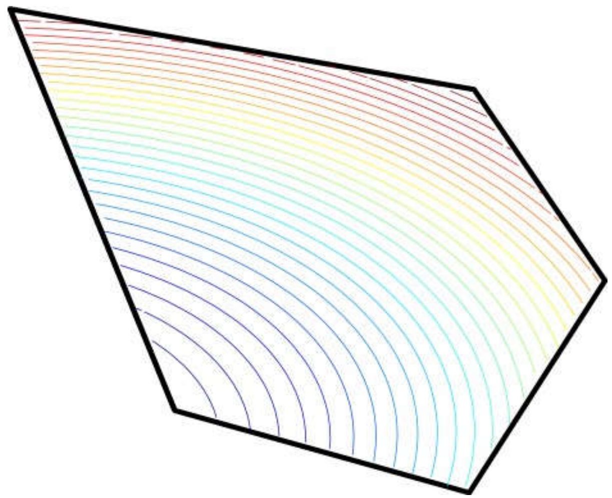
$$\tilde{z} \in \text{dom } f, \quad g_i(\tilde{z}) < 0, i = 1, \dots, m$$

- **Idea:** There exist many methods for unconstrained minimization
  $\Rightarrow$ Reformulate problem as an unconstrained problem

Constrained Optimization

- **Graphical ilustration**

  Define function as ∞ if constraints violated.



Minimize this function over R^n

Constrained Optimization

Barrier Method

$$\begin{array}{ll} \text{min} & f(z) \\ \text{s.t.} & g_i(z) \leq 0,\ i = 1, \ldots, m \end{array} \quad \Leftrightarrow \quad \min \quad f(z) + \mu\phi(z)$$

Constraints have been moved to objective via *indicator function:*

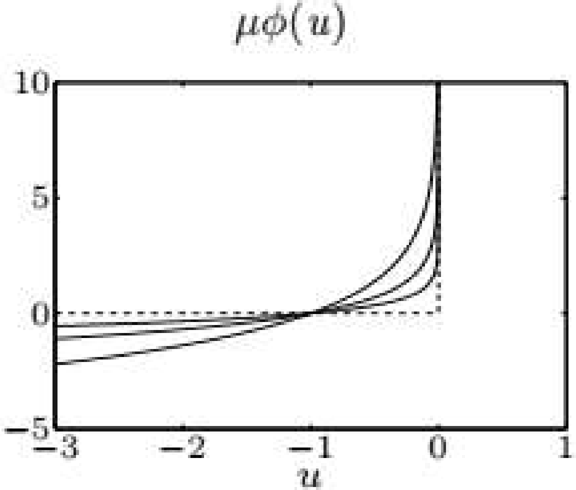
$$\phi(z) = \sum_{i=1}^{m} I_{-}(g_i(z)), \quad \mu = 1$$

where $I_{-}(u) = 0$ if $u \leq 0$ and $I_{-} = \infty$ otherwise

- Augmented cost is not differentiable
  $\rightarrow$ approximation by *logarithmic barrier:*

$$\phi(z) = -\sum_{i=1}^{m} \log(-g_i(z))$$

- For $\mu > 0$ smooth approximation of indicator function
- Approximation improves as $\mu \to 0$



Constrained Optimization

Logarithmic Barrier Function

$$\phi(z) = -\sum_{i=1}^{m} log(-g_i(z)), \quad \text{dom}\ \phi = \{z \mid g_1(z) < 0, \ldots, g_m(z) < 0\}$$

- Convex, smooth on its domain
- $\phi(z) \to \infty$ as z approaches boundary of domain
- $\arg\min_z \phi(z)$ is called **analytic center** of the set defined by inequalities $g_1 < 0, \ldots, g_m < 0$
- Twice continuously differentiable with derivatives

$$\nabla\phi(z) = \sum_{i=1}^{m} \frac{1}{-g_i(z)} \nabla g_i(z)$$

$$\nabla^2\phi(z) = \sum_{i=1}^{m} \frac{1}{g_i(z)^2} \nabla g_i(z) \nabla g_i(z)^T + \frac{1}{-g_i(z)} \nabla^2 g_i(z)$$

Constrained Optimization

Central Path

- Define $z^*(\mu)$ as the solution of

$$\min_z f(z) + \mu\phi(z)$$

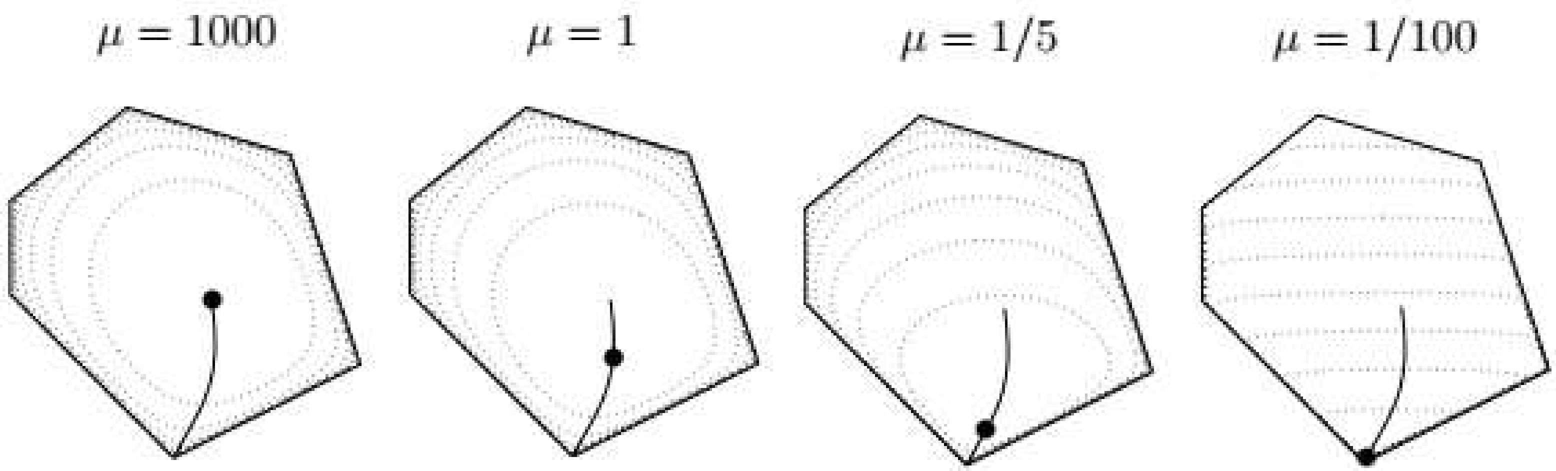
  (assume minimizer exists and is unique for each $\mu > 0$)
- Barrier parameter μ determines relative weight between objective and barrier
- Barrier 'traps' $z(\mu)$ in strictly feasible set
- *Central path* is defined as $\{z^*(\mu) \mid \mu > 0\}$
- For given μ can compute $z^*(\mu)$ by solving smooth unconstrained minimization problem
- Intuitively $z^*(\mu)$ converges to optimal solution as $\mu \to 0$ (easy to prove under mild conditions)

Constrained Optimization

Example: Central Path for an LP

$$\begin{array}{ll} \text{min} & c^T z \\ \text{s.t.} & a_i^T z \le b_i, i = 1, \ldots, 6 \end{array}$$

$z \in \mathbb{R}^2$, c points upward



Constrained Optimization

Path-following Method

Idea: Follow central path to the optimal solution

Solve sequence of smooth unconstrained problems:

$$z^*(\mu) = \arg\min_z f(z) + \mu\phi(z)$$

- Assume current solution is on the central path $z^i = z^*(\mu^i)$
- Obtain μ^{i+1} by decreasing μ^i by some amount
- Solve for $z^*(\mu^{i+1})$ starting from $z^*(\mu^i)$ (unconstrained optimization)
- Method converges to the optimal solution, i.e., $z^i \to z^*$ for $\mu \to 0$

Constrained Optimization

Active Set Idea

Consider

$$\begin{array}{ll} \min_z & f(z) \\ \text{subj. to} & z \in S, \end{array}$$

where the feasible set $S \subset \mathbb{R}^s$ is a polyhedron, i.e. a set defined by linear equalities and inequalities, and the objective f is a linear function (*linear programming* (LP)) or a convex quadratic function (*quadratic programming* (QP)).

- Active set methods aim to identify the set of active constraints at the solution. Once this set is known, a solution to the problem can be easily identified.
- Since the number of potentially visited active sets depends combinatorially on the number of decision variables and constraints, these methods have a worst case complexity that is exponential in the problem size (as opposed to first-order and interior point methods). However, active set methods have proved to work quite well in practice.

Constrained Optimization

Active Set for LPs

$$\begin{array}{ll} \min_z & c'z \\ \text{subj. to} & Gz \leq w \end{array} \tag{3}$$

- Central to active set methods for LP is the observation that a solution is always attained at a vertex of the polyhedral feasible set.
- A simple strategy would be to enumerate all vertices of the polyhedron and declare the vertex with the smallest cost as solution.
- However, one can do better by only visiting those vertices that improve the cost at the previous ones.
- This is the main idea behind active set methods for LP.

Modeling languages for optimization problems

- **YALMIP** MATLAB-based modeling language
- **CVX (CVXPY)** Modeling language for convex problems in (MATLAB , python)
- **CASADI + IPOPT** Nonlinear modeling + automatic differentiation, nonlinear programming solver (MATLAB, python ,C++)
- **Optimization Toolbox**’ modeling language (part of MATLAB since R2017b)
- **PYOMO** python -based modeling language
- **PuLP** An linear programming modeler for python
- **GAMS** (www.gams.com): commercial high-level modeling system for large-scale optimization. Supports many different types of problems (LPs, QCQPs, MILPs, MINLPs, ...) and solvers

Solving optimization problems

A simple optimization problem:

$$\min_{x \in R} |x_1 + 6| + |x_2 - 4|$$

$$s.t \quad 3 \le x_1 \le 5, \quad -2 \le x_2 \le 2$$

The **YALMIP toolbox** for Matlab (from ETH / Linkoping):

```
%make variables
sdpvar x1 x2;
%define cost function
f = abs(x1 + 6) + abs(x2 - 4);
%define constraints
X = set(3 <= x1 <= 5) + …
set( -2 <= x2 <= 2);
%solve
solvesdp(X,f);
```

Solving optimization problems

A simple optimization problem:

$$\min_{x \in R} |x_1 + 6| + |x_2 - 4|$$

$$s.t \quad 3 \le x_1 \le 5, \quad -2 \le x_2 \le 2$$

The **CVX toolbox** for Matlab (from Stanford):

```
cvx_begin
variables x1 x2 % define variables
%define cost function and constraints
minimize(abs(x1 + 6) + abs(x2 - 4))
subject to
3 <= x1 <= 5
-2 <= x2 <= 2
cvx _end %solves automatically
```