CS 287 Advanced Robotics (Fall 2019)
Lecture 7: Constrained Optimization

Pieter Abbeel
UC Berkeley EECS

[optional] Boyd and Vandenberghe, Convex Optimization, Chapters 9 – 11
[optional] Nocedal and Wright, Chapter 18

Outline

- Constrained Optimization
- Penalty Formulation
- Convex Programs and Solvers
- Dual Descent

Outline

- ***Constrained Optimization***
- Penalty Formulation
- Convex Programs and Solvers
- Dual Descent

Constrained Optimization

$$\begin{aligned} \min_x \quad & g_0(x) \\ \text{s.t.} \quad & g_i(x) \leq 0 \quad \forall i \\ & h_j(x) = 0 \quad \forall j \end{aligned}$$

Outline

- Constrained Optimization
- ***Penalty Formulation***
- Convex Programs and Solvers
- Dual Descent

Penalty Formulation

Original:

$$\begin{aligned} \min_x \quad & g_0(x) \\ \text{s.t.} \quad & g_i(x) \le 0 \quad \forall i \\ & h_j(x) = 0 \quad \forall j \end{aligned}$$

- constrained

Penalty Formulation:

$$\min_x \quad g_0(x) + \mu \sum_i |g_i(x)|^+ + \mu \sum_j |h_j(x)|$$

- now unconstrained
- same solution for mu large enough

Penalty Method

- *Inner loop: optimize merit function*

$$\min_x \; g_0(x) + \mu \sum_i |g_i(x)|^+ + \mu \sum_j |h_j(x)| \qquad = \min_x \; f_\mu(x)$$

merit function

and increase μ in an outer loop until the two sums equal zero.

- *Inner loop optimization can be done by any of:*
  - Gradient descent
  - Newton or quasi-Newton method
  - Trust region method

Penalty Method w/Trust Region Inner Loop

- *Inner loop: optimize merit function*

merit function

$$\min_x \quad g_0(x) + \mu \sum_i |g_i(x)|^+ + \mu \sum_j |h_j(x)| \qquad = \min_x \quad f_\mu(x)$$

and increase μ in an outer loop until the two sums equal zero.

- *Trust region method repeatedly solves:*

$$\min_x \quad g_0(\bar{x}) + \nabla_x g_0(\bar{x})(x-\bar{x}) + \mu \sum_i |g_i(\bar{x}) + \nabla_x g_i(\bar{x})(x-\bar{x})|^+ + \mu \sum_j |h_j(\bar{x}) + \nabla_x h_j(\bar{x})(x-\bar{x})|$$

$$\text{s.t.} \quad \|x-\bar{x}\|_2 \leq \varepsilon$$ (trust region constraint)

$\bar{x}$: current point

Inputs: $\bar{x}, \mu = 1, \varepsilon_0, \alpha \in (0.5, 1), \beta \in (0, 1), t \in (1, \infty)$

WHILE ($\sum_i |g_i(\bar{x})|^+ + \sum_j |h_j(\bar{x})| \geq \delta$ **AND** $\mu < \mu_{\text{MAX}}$)

$\mu \leftarrow t\mu, \quad \varepsilon \leftarrow \varepsilon_0$ // increase penalty coefficient for constraints; re-init trust region size

WHILE (1) // [2] loop that optimizes

Compute terms of first-order approximations: $g_0(\bar{x}), \nabla_x g_0(\bar{x}), g_i(\bar{x}), \nabla_x g_i(\bar{x}), h_j(\bar{x}), \nabla_x h_j(\bar{x}), \quad \forall i, j$

WHILE (1) // [3] loop that does trust-region size search

Call convex program solver to solve:

$$\bar{f}_\mu(\bar{x}_{\text{next?}}) = \min_x \quad g_0(\bar{x}) + \nabla_x g_0(\bar{x})(x - \bar{x}) + \mu \textstyle\sum_i |g_i(\bar{x}) + \nabla_x g_i(\bar{x})(x - \bar{x})|^+$$
$$+\mu \textstyle\sum_j |h_j(\bar{x}) + \nabla_x h_j(\bar{x})(x - \bar{x})| \qquad \text{s.t.} \quad \|x - \bar{x}\|_2 \leq \varepsilon$$

IF $f_\mu(\bar{x}) - f_\mu(\bar{x}_{\text{next?}}) \geq \alpha \left(\bar{f}_\mu(\bar{x}) - \bar{f}_\mu(\bar{x}_{\text{next?}}) \right)$

THEN: Update $\bar{x} \leftarrow \bar{x}_{\text{next?}}$ **AND** Update (Grow) trust region: $\varepsilon \leftarrow \varepsilon / \beta$ **AND** **BREAK** out of while [3]

ELSE: No update to $\bar{x}$ **AND** Update (Shrink) trust region: $\varepsilon \leftarrow \beta\varepsilon$

IF ε below some threshold **THEN: BREAK** out of while [3] and while [2]

Tweak: Retain Convex Terms Exactly

- Non-convex optimization with convex parts separated:

$$\begin{aligned} \min_x \quad & f_0(x) + g_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 0 \quad \forall i \\ & Ax - b = 0 \quad \forall j \\ & g_k(x) \leq 0 \quad \forall k \\ & h_l(x) = 0 \quad \forall l \end{aligned}$$

with:
f_i convex
g_k non-convex
h_l nonlinear

- Retain convex parts and in inner loop solve:

$$\begin{aligned} \min_x \quad & f_0(x) + g_0(x) + \mu \sum_k |g_k(x)|^+ + \mu \sum_l |h_l(x)| \\ \text{s.t.} \quad & f_i(x) \leq 0 \quad \forall i \\ & Ax - b = 0 \quad \forall j \end{aligned}$$

Outline

- Constrained Optimization
- Penalty Formulation
- ***Convex Programs and Solvers***
- Dual Descent

Convex Optimization Problems

- Convex optimization problems are a special class of optimization problems, of the following form:

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} f_0(x) \\ \text{s.t.} \quad & f_i(x) \le 0 \qquad i = 1, \ldots, n \\ & Ax = b \end{aligned}$$

with $f_i(x)$ convex for i = 0, 1, …, n

Convex Functions

- A function f is convex if and only if

$$\forall x_1, x_2 \in \text{Domain}(f), \forall t \in [0,1]:$$
$$f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2)$$

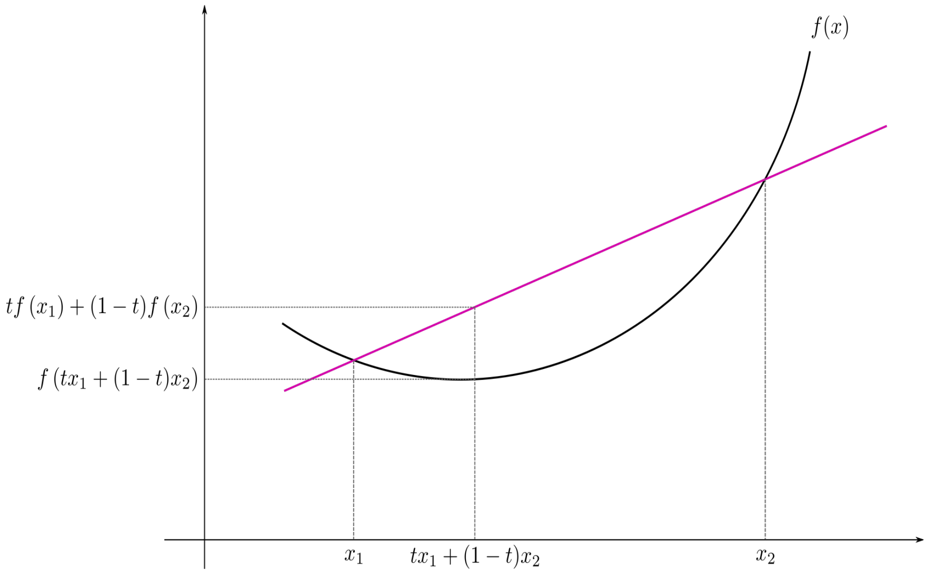
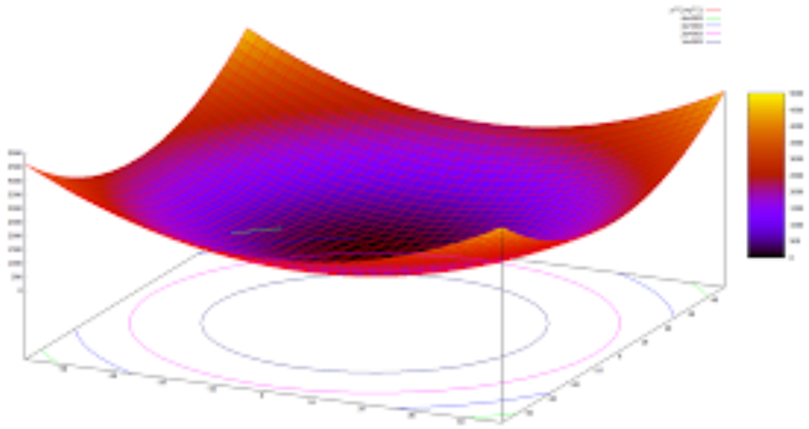


Image source: wikipedia

Convex Functions



- Unique minimum
- Set of points for which f(x) <= a is convex

Source: Thomas Jungblut's Blog

Outline

- Constrained Optimization
- Penalty Formulation
- ***Convex Programs and Solvers***
  - ***Equality Constraints***
  - Inequality Constraints
- Dual Descent

Convex Problems: Equality Constrained Minimization

- Problem to be solved:

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s.t.} \quad & Ax = b \end{aligned}$$

- We will cover three solution methods:
  - Elimination
  - Newton’s method
  - Infeasible start Newton method

Method 1: Elimination

- From linear algebra we know that there exist a matrix F (in fact infinitely many) such that:

$$\{x | Ax = b\} = \{x | x = \hat{x} + Fz\}$$

$\hat{x}$: any solution to Ax = b

F: spans the null-space of A

A way to find an F: compute SVD of A, A = U S V', for A having k nonzero singular values, set F = U(:, k+1:end)

- So we can solve the equality constrained minimization problem by solving an ***unconstrained minimization problem over a new variable z***:

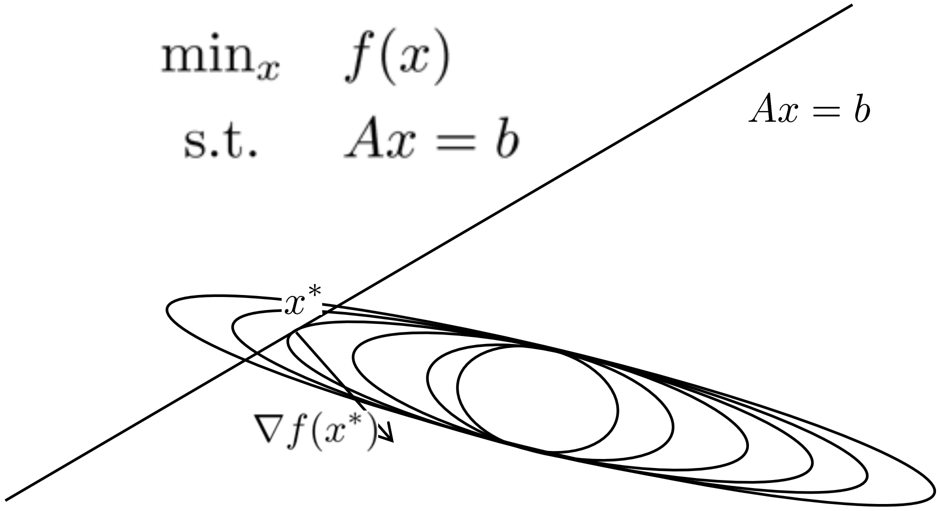
$$\min_z f(\hat{x} + Fz)$$

- Potential cons: (i) need to first find a solution to Ax=b, (ii) need to find F, (iii) elimination might destroy sparsity in original problem structure

Methods 2 and 3 --- First Consider Optimality Condition

- Recall problem to be solved:

$$\begin{array}{ll} \min_x & f(x) \\ \text{s.t.} & Ax = b \end{array}$$



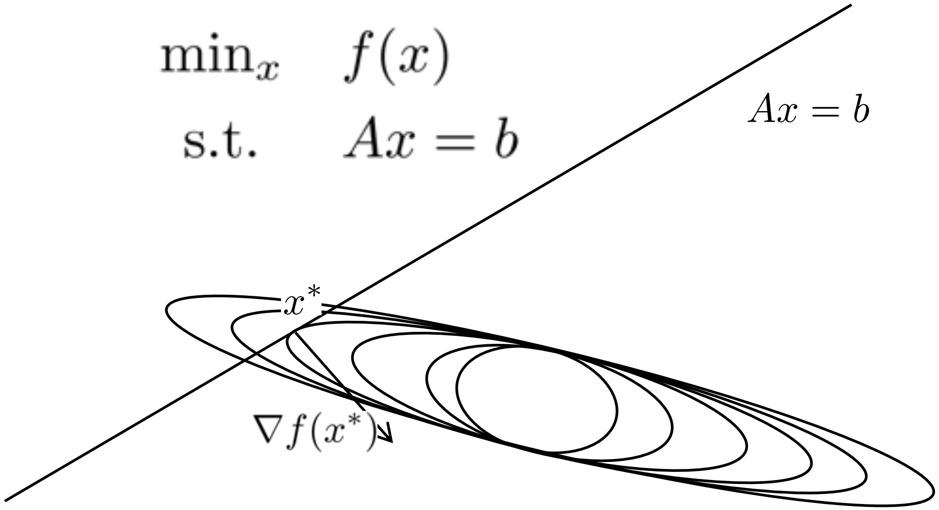
x* with Ax*=b is (local) optimum if and only if: $\forall \Delta x \quad \text{if } A\Delta x = 0 \text{ then } \nabla f(x^*)^\top \Delta x = 0.$

Equivalently: $\nabla f(x^*)^\top = \nu^\top A$

Methods 2 and 3 --- First Consider Optimality Condition

- Recall problem to be solved:

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s.t.} \quad & Ax = b \end{aligned}$$



Optimality Condition: $Ax^* = b$ and $\nabla f(x^*) + A^\top \nu = 0$

Method 2: Newton's Method

- Problem to be solved:

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s.t.} \quad & Ax = b \end{aligned}$$

- Optimality Condition: $Ax^* = b$ and $\nabla f(x^*) + A^\top \nu = 0$

- Assume x is feasible, i.e., satisfies Ax = b, now use 2nd order approximation of f:

$$\begin{aligned} \min_{\Delta x} \quad & f(x) + \nabla f(x)^\top \Delta x + \frac{1}{2} \Delta x^\top \nabla^2 f(x) \Delta x \\ \text{s.t.} \quad & A(x + \Delta x) = b \end{aligned}$$

- Optimality condition for 2nd order approximation:

$$\begin{bmatrix} \nabla^2 f(x) & A^\top \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \nu \end{bmatrix} = \begin{bmatrix} -\nabla f(x) \\ 0 \end{bmatrix}$$

Method 2: Newton's Method

given starting point $x \in \mathbf{dom}\, f$ with $Ax = b$, tolerance $\epsilon > 0$.

repeat

1. Compute the Newton step and decrement Δx_{nt}, $\lambda(x)$.
2. *Stopping criterion.* **quit** if $\lambda^2/2 \leq \epsilon$.
3. *Line search.* Choose step size t by backtracking line search.
4. *Update.* $x := x + t\Delta x_{\mathrm{nt}}$.

With Newton step obtained by solving a linear system of equations:

$$\begin{bmatrix} \nabla^2 f(x) & A^\top \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x_{\mathrm{nt}} \\ \nu \end{bmatrix} = \begin{bmatrix} -\nabla f(x) \\ 0 \end{bmatrix}$$

Feasible descent method: $x^{(k)}$ feasible and $f(x^{(k+1)}) \leq f(x^{(k)})$

Method 3: Infeasible Start Newton Method

- Problem to be solved:
$$\begin{array}{ll} \min_x & f(x) \\ \text{s.t.} & Ax = b \end{array}$$

- Optimality Condition: $Ax^* = b$ and $\nabla f(x^*) + A^\top \nu = 0$

- Use 1st order approximation of the optimality conditions at current x:

$$\begin{aligned} A(x + \Delta x) &= b \\ \nabla f(x) + \nabla^2 f(x)\Delta x + A^\top \nu &= 0 \end{aligned}$$

- Equivalently:

$$\begin{bmatrix} \nabla^2 f(x) & A^\top \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \nu \end{bmatrix} = \begin{bmatrix} -\nabla f(x) \\ b - Ax \end{bmatrix}$$

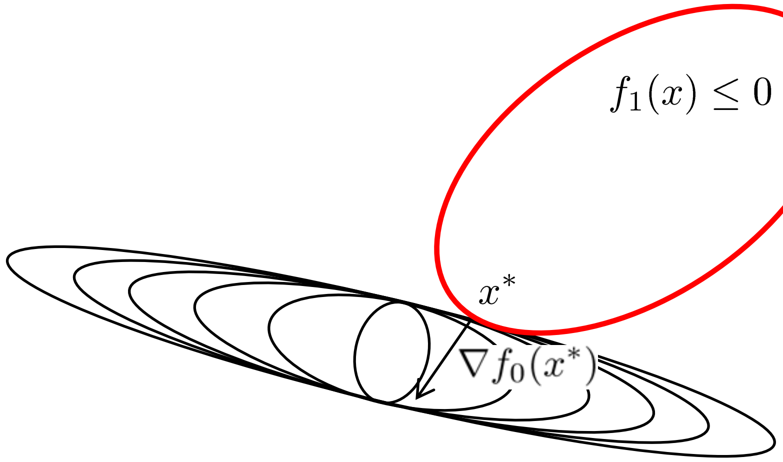
Outline

- Constrained Optimization
- Penalty Formulation
- ***Convex Programs and Solvers***
  - Equality Constraints
  - ***Inequality Constraints: Barrier Method***
- Dual Descent

Convex Problems: Equality and Inequality Constrained Minimization

- Recall the problem to be solved:

$$
\begin{aligned}
\min_x \quad & f_0(x) \\
\text{s.t.} \quad & f_i(x) \le 0, \quad i = 1, \ldots, m \\
& Ax = b
\end{aligned}
$$



Equality and Inequality Constrained Minimization

- Problem to be solved:

$$\begin{aligned} \min_x \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 0, \quad i = 1, \dots, m \\ & Ax = b \end{aligned}$$

- Reformulation via indicator function

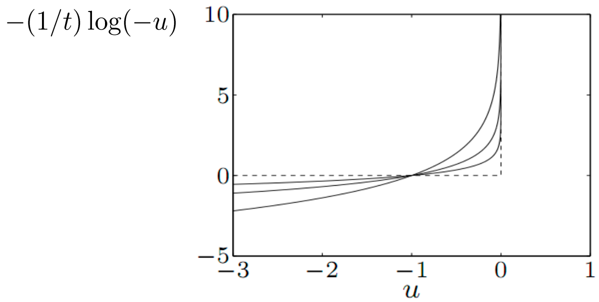
$$\begin{aligned} \min_x \quad & f_0(x) + \sum_{I=1}^{m} I_-(f_i(x)) \\ & Ax = b \end{aligned}$$

→ No inequality constraints anymore, but very poorly conditioned objective function

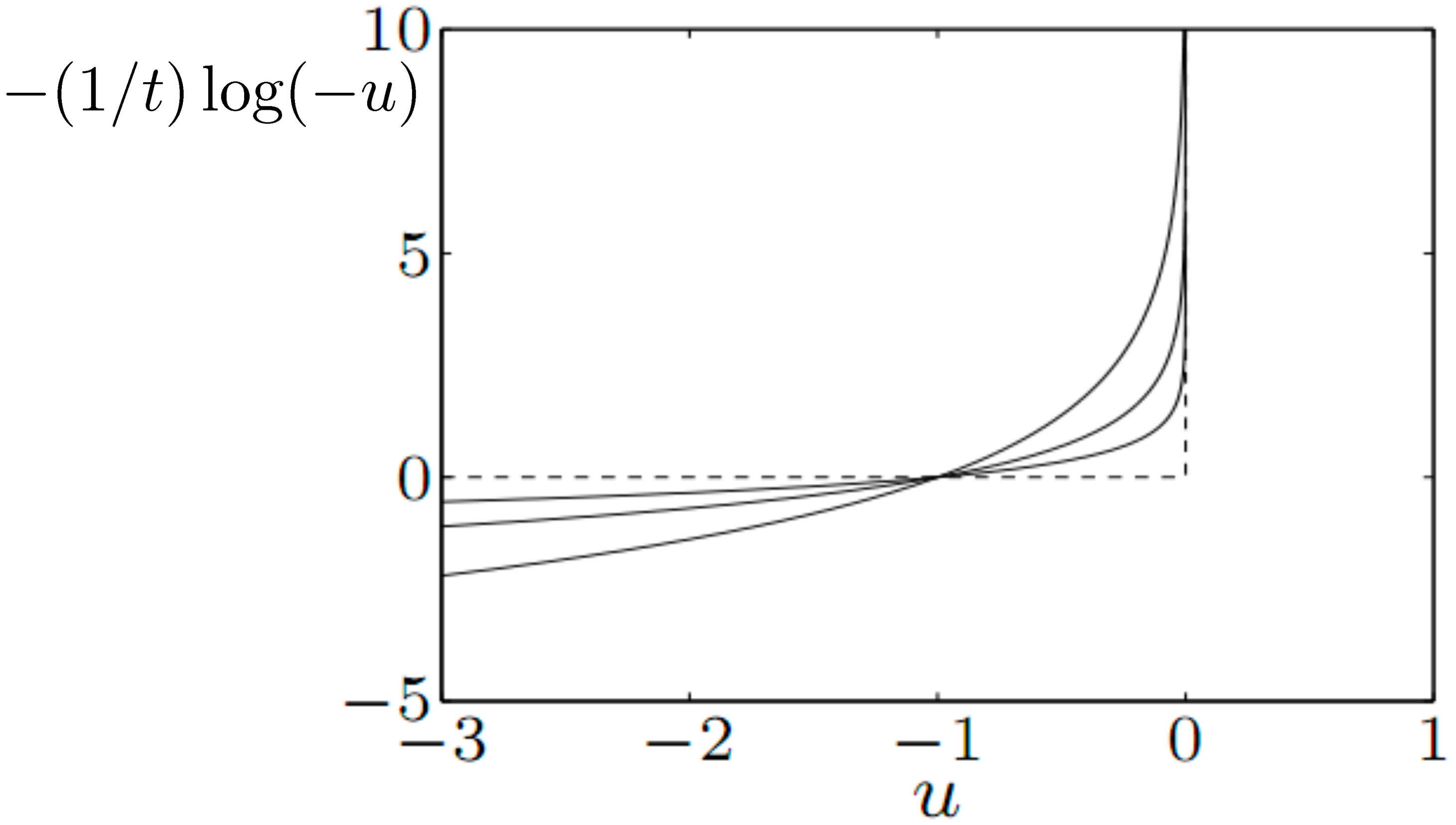
- Approximation via logarithmic barrier:

$$\begin{aligned} \min_x \quad & f_0(x) - (1/t) \sum_{i=1}^{m} \log(-f_i(x)) \\ \text{s.t.} \quad & Ax = b \end{aligned}$$

* for t > 0, -(1/t) log(-u) is a smooth approximation of *I*_(u)
* approximation improves for *t* → infinity
* better conditioned for smaller *t*



Equality and Inequality Constrained Minimization



Barrier Method

- Given: strictly feasible x, $t=t^{(0)} > 0$, $\mu > 1$, tolerance $\varepsilon > 0$
- Repeat

1. *Centering Step.* Compute $x^*(t)$ by solving

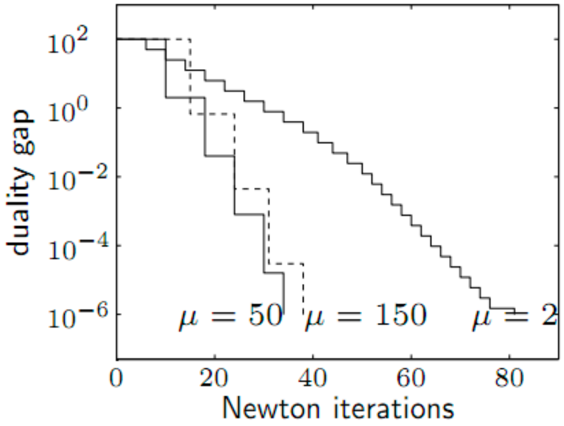
$$\begin{aligned} \min_x \quad & f_0(x) - (1/t)\sum_{i=1}^{m}\log(-f_i(x)) \\ \text{s.t.} \quad & Ax = b \end{aligned}$$

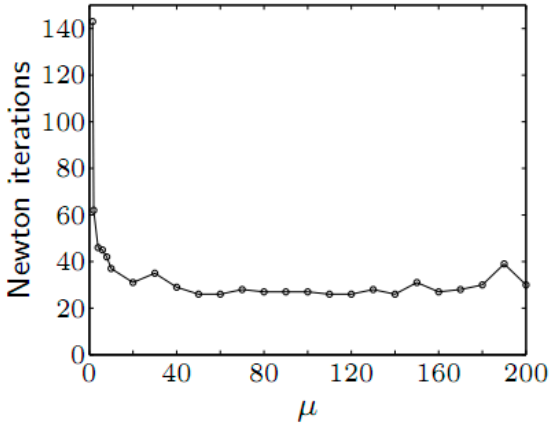
   starting from x

2. *Update.* $x := x^*(t)$.
3. *Stopping Criterion.* Quit if $m/t < \varepsilon$
4. *Increase t.* $t := \mu t$

Example 1: Inequality Form LP

inequality form LP ($m = 100$ inequalities, $n = 50$ variables)



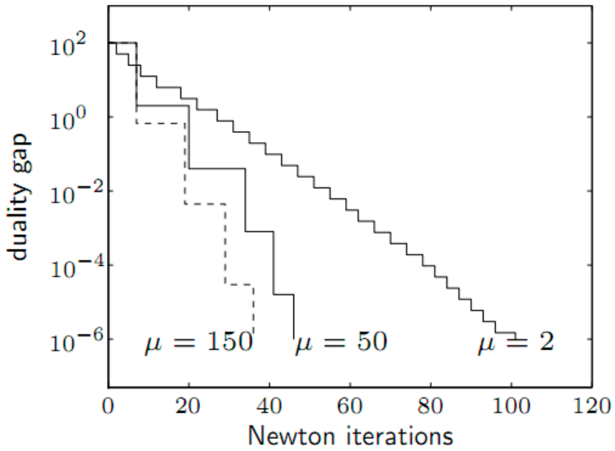


- starts with x on central path ($t^{(0)} = 1$, duality gap 100)
- terminates when $t = 10^8$ (gap 10^{-6})
- centering uses Newton's method with backtracking
- total number of Newton iterations not very sensitive for $\mu \geq 10$

Example 2: Geometric Program

geometric program ($m = 100$ inequalities and $n = 50$ variables)

$$\begin{array}{ll} \text{minimize} & \log\left(\sum_{k=1}^{5} \exp(a_{0k}^T x + b_{0k})\right) \\ \text{subject to} & \log\left(\sum_{k=1}^{5} \exp(a_{ik}^T x + b_{ik})\right) \le 0, \quad i = 1, \ldots, m \end{array}$$

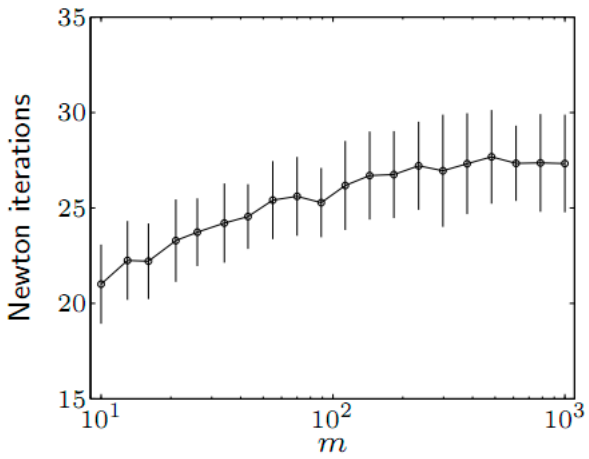


Example 3: Standard LPs

family of standard LPs $(A \in \mathbf{R}^{m \times 2m})$

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b, \quad x \succeq 0 \end{array}$$

$m = 10, \ldots, 1000$; for each m, solve 100 randomly generated instances



number of iterations grows very slowly as m ranges over a $100:1$ ratio

Initialization

- Basic phase I method:

Initialize by first solving:

$$\begin{aligned} \min_{x,s} \quad & s \\ \text{s.t.} \quad & f_i(x) \le s, \quad i = 1, \ldots, m \\ & Ax = b \end{aligned}$$

- Easy to initialize above problem, pick some x such that Ax = b, and then simply set $s = \max_i f_i(x)$
- Can stop early---whenever $s < 0$

Initialization

- Sum of infeasibilities phase I method:
- Initialize by first solving:

$$\begin{aligned} \min_{x,s} \quad & \sum_{I=1}^{m} s_i \\ \text{s.t.} \quad & f_i(x) \leq s_i, \quad i = 1, \ldots, m \\ & s_i \geq 0, \quad i = 1, \ldots, m \\ & Ax = b \end{aligned}$$

- Easy to initialize above problem, pick some x such that Ax = b, and then simply set $s_i = \max(0, f_i(x))$
- For infeasible problems, produces a solution that satisfies many more inequalities than basic phase I method

Other methods for convex problems

- We have covered a primal interior point method / barrier method
  - one of several optimization approaches
- Examples of others:
  - Primal-dual interior point methods
  - Primal-dual infeasible interior point methods

Outline

- Constrained Optimization
- Penalty Formulation
- Convex Programs and Solvers
  - Equality Constraints
  - Inequality Constraints: Barrier Method
- ***Dual Descent***

Formulation

Original:

$$\begin{aligned} \min_x \quad & g_0(x) \\ \text{s.t.} \quad & g_i(x) \leq 0 \quad \forall i \\ & h_j(x) = 0 \quad \forall j \end{aligned}$$

Penalty Formulation:

$$\min_x \quad g_0(x) + \mu \sum_i |g_i(x)|^+ + \mu \sum_j |h_j(x)|$$

Penalty Method iterates:

- Optimize over x
- Increase mu as needed

$$\mu \leftarrow t * \mu$$

Dual-Descent Formulation:

$$\max_{\lambda \geq 0, \nu} \min_x g_0(x) + \sum_i \lambda_i g_i(x) + \sum_j \nu_j h_j(x)$$

Dual Descent iterates:

- Optimize over x
- Gradient descent step for lambda and nu

$$\lambda_i \leftarrow \lambda_i + \alpha g_i(x)$$

$$\nu_j \leftarrow \nu_j + \alpha h_j(x)$$

New, equivalent problem with same solution:

$$\begin{aligned} \min_x \quad & g_0(x) \\ \text{s.t.} \quad & |g_i(x)|^+ \leq 0 \ \forall i \\ & |h_j(x)| = 0 \ \forall j \end{aligned}$$

$$\max_{\lambda \geq 0, \nu} \min_x g_0(x) + \sum_i \lambda_i |g_i(x)|^+ + \sum_j \nu_j |h_j(x)|$$

Dual-Descent Formulation of new, equivalent problem almost identical to penalty formulation, but individual additive updates to lambda and nu, rather than scaling up of a single mu

Next Lecture

Optimization-based Optimal Control! ☺