COMP 9602: Convex Optimization

Decomposition Methods

Dr. C Wu

Department of Computer Science
The University of Hong Kong

Roadmap

Theory	convex set convex function standard forms of optimization problems, quasi-convex optimization linear program, integer linear program quadratic program geometric program semidefinite program vector optimization duality
Algorithm	unconstrained optimization equality constrained optimization interior-point method subgradient methods localization methods decomposition methods and more

Decomposition methods

- Break a problem into smaller ones and solve each of the smaller ones separably, either in parallel or sequentially
- Problems
  - Separable problems
  - problems with complicating variables
  - problems with complicating constraints
- Decomposition methods
  - primal decomposition
  - dual decomposition

Separable problems

$$\begin{array}{ll} \text{minimize} & f_1(x_1) + f_2(x_2) \\ \text{subject to} & x_1 \in \mathcal{C}_1, \quad x_2 \in \mathcal{C}_2 \end{array}$$

- we can solve for x_1 and x_2 separately (in parallel)

Problems with complicating variables

consider unconstrained problem,

$$\text{minimize} \quad f(x) = f_1(x_1, y) + f_2(x_2, y)$$

$x = (x_1, x_2, y)$

- y is the **complicating variable** or **coupling variable**; when it is fixed the problem is separable in x_1 and x_2

Primal decomposition method

fix y and define

$$\begin{array}{lll} \text{subproblem 1:} & \text{minimize}_{x_1} & f_1(x_1, y) \\ \text{subproblem 2:} & \text{minimize}_{x_2} & f_2(x_2, y) \end{array}$$

with optimal values $\phi_1(y)$ and $\phi_2(y)$

master problem

$$\text{minimize}_y \quad \phi_1(y) + \phi_2(y)$$

- can solve master problem using
  - bisection (if y is scalar)
  - gradient or Newton method (if ϕ_i differentiable)
  - subgradient, cutting-plane, or ellipsoid method
- each iteration of master problem requires solving the two subproblems

Primal decomposition method (cont'd)

- Algorithm sketch

Given $y^{(0)}$, $k = 0$;

Repeat

solve two subproblems to derive x_1, x_2

update $y^{(k)}$ to $y^{(k+1)}$ based on the algorithm to solve the master problem

Primal decomposition method (cont'd)

- Example (solve master problem using the subgradient method)

Given $y^{(0)}$, $k = 0$;

Repeat

find x_1 that minimizes $f_1(x_1, y)$, $g_1 \in \partial\phi_1(y)$

find x_2 that minimizes $f_2(x_2, y)$, $g_2 \in \partial\phi_2(y)$

update $y^{(k+1)} = y^{(k)} - \alpha_k(g_1 + g_2)$

Problems with complicating constraints

$$
\begin{array}{ll}
\text{minimize} & f_1(x_1) + f_2(x_2) \\
\text{subject to} & x_1 \in \mathcal{C}_1, \quad x_2 \in \mathcal{C}_2 \\
& h_1(x_1) + h_2(x_2) \preceq 0
\end{array}
$$

- f_i, h_i, $\mathcal{C}_i$ convex
- $h_1(x_1) + h_2(x_2) \preceq 0$ is a set of p complicating or coupling constraints, involving both x_1 and x_2
- can interpret coupling constraints as limits on resources shared between two subproblems

Dual decomposition method

form (separable) partial Lagrangian

$$\begin{aligned} L(x_1, x_2, \lambda) &= f_1(x_1) + f_2(x_2) + \lambda^T(h_1(x_1) + h_2(x_2)) \\ &= \big(f_1(x_1) + \lambda^T h_1(x_1)\big) + \big(f_2(x_2) + \lambda^T h_2(x_2)\big) \end{aligned}$$

Lagrange dual

$$g(\lambda) = \underset{x1 \in \mathcal{C}_1,\, x2 \in \mathcal{C}_2}{\text{minimize}} \; (f_1(x_1) + \lambda^T h_1(x_1)) + (f_2(x_2) + \lambda^T h_2(x_2))$$

dual problem

$$\begin{aligned} &\text{maximize } g(\lambda) \\ &\quad \text{s.t. } \lambda \succeq 0 \end{aligned}$$

Dual decomposition method (cont'd)

fix dual variable λ and define

subproblem 1: $\begin{array}{ll} \text{minimize} & f_1(x_1) + \lambda^T h_1(x_1) \\ \text{subject to} & x_1 \in \mathcal{C}_1 \end{array}$

subproblem 2: $\begin{array}{ll} \text{minimize} & f_2(x_2) + \lambda^T h_2(x_2) \\ \text{subject to} & x_2 \in \mathcal{C}_2 \end{array}$

with optimal values $g_1(\lambda)$, $g_2(\lambda)$ $\quad (g(\lambda) = g_1(\lambda) + g_2(\lambda))$

- $-h_i(\bar{x}_i) \in \partial(-g_i)(\lambda)$, where $\bar{x}_i$ is any solution to subproblem i
- $-(h_1(\bar{x}_1) + h_2(\bar{x}_2)) \in \partial(-g)(\lambda)$

Dual decomposition method (cont'd)

- Lagrangian relaxation and subgradient method

repeat

1. Solve the subproblems.
   Solve subproblem 1, finding an optimal $\bar{x}_1$.
   Solve subproblem 2, finding an optimal $\bar{x}_2$.
2. Update dual variables (prices).
   $\lambda := (\lambda + \alpha_k(h_1(\bar{x}_1) + h_2(\bar{x}_2)))_+$.

- α_k is an appropriate step size
- iterates need not be feasible
- can again construct feasible primal variables using projection

Same idea as projected subgradient method on dual problem (pp. 24, 13_Subgradient_C9602_Fall2018.pdf)

Example: rate control

Problem setup

- n flows, with fixed routes, in a network with m links
- variable $f_j \geq 0$ denotes the rate of flow j
- flow utility is $U_j : \mathbf{R} \to \mathbf{R}$, strictly concave, increasing
- traffic t_i on link i is sum of flows passing through it
- $t = Rf$, where R is the routing matrix

$$R_{ij} = \begin{cases} 1 & \text{flow } j \text{ passes over link } i \\ 0 & \text{otherwise} \end{cases}$$

- link capacity constraint: $t \preceq c$

Example: rate control

- Rate control problem:

$$\begin{array}{ll} \text{maximize} & U(f) = \sum_{j=1}^{n} U_j(f_j) \\ \text{subject to} & Rf \preceq c \end{array}$$

- convex problem
- dual decomposition gives decentralized method

Example: rate control

- Dual decomposition rate control algorithm:

given initial link price vector $\lambda \succ 0$ ($e.g.$, $\lambda = \mathbf{1}$).
repeat

1. Sum link prices along each route.
   Calculate $\Lambda_j = r_j^T \lambda$.
2. Optimize flows (separately) using flow prices.
   $f_j := \mathrm{argmax}\,(U_j(f_j) - \Lambda_j f_j)$.
3. Calculate link capacity margins.
   $s := c - Rf$.
4. Update link prices.
   $\lambda := (\lambda - \alpha_k s)_+$.

- decentralized:
  - links only need to know the flows that pass through them
  - flows only need to know prices on links they pass through

Example: rate control

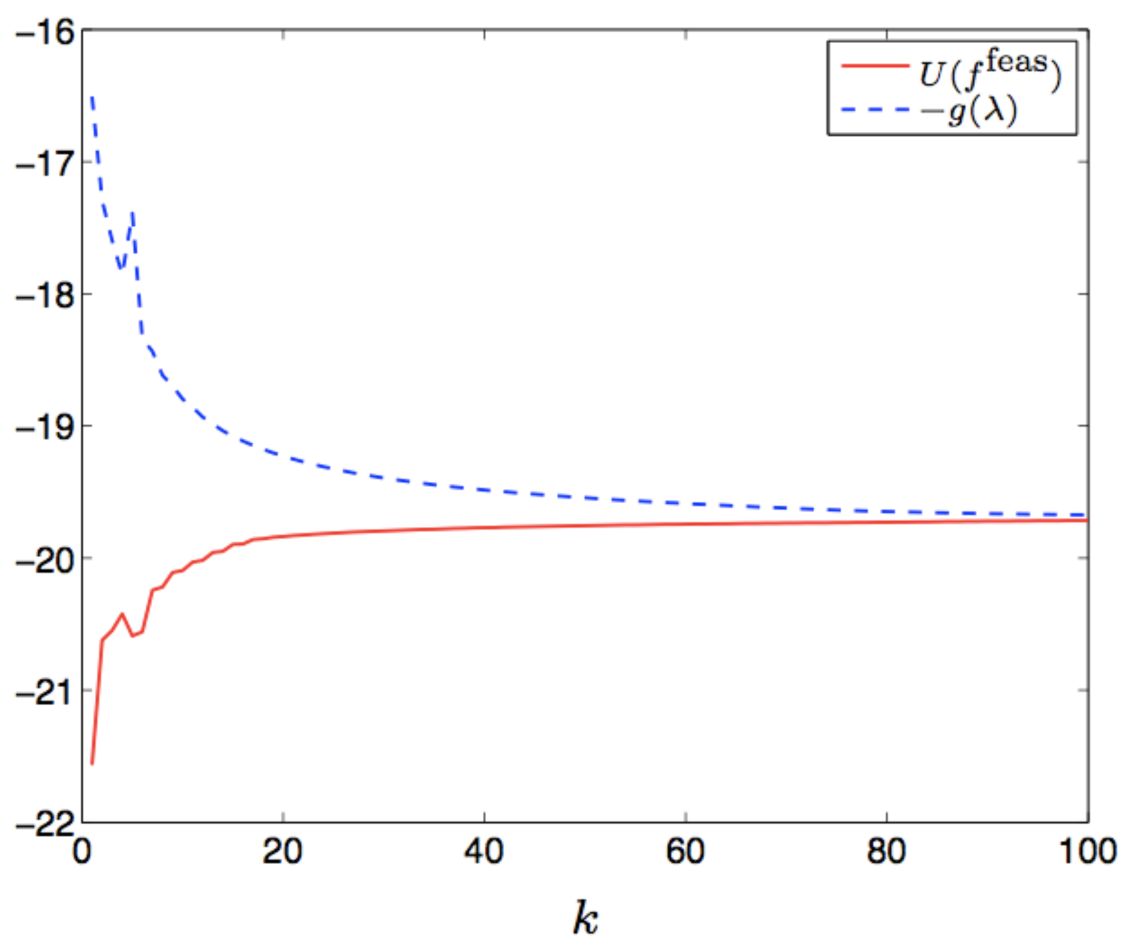
- Generating feasible flow rates:
  - iterates can be (and often are) infeasible, $i.e.$, $Rf \not\preceq c$ (but we do have $Rf \preceq c$ in the limit)
  - define $\eta_i = t_i/c_i = (Rf)_i/c_i$
    - $\eta_i < 1$ means link i is under capacity
    - $\eta_i > 1$ means link i is over capacity
  - define f^{feas} as

$$f_j^{\mathrm{feas}} = \frac{f_j}{\max\{\eta_i \mid \text{flow } j \text{ passes over link } i\}}$$

Example: rate control

Convergence

- $n = 10$ flows, $m = 12$ links; 3 or 4 links per flow
- link capacities chosen randomly, uniform on $[0.1, 1]$
- $U_j(f_j) = \log f_j$
- optimal flow as a function of price: $\bar{f}_j = \text{argmax}(U_j(f_j) - \Lambda_j f_j) = 1/\Lambda_j$
- initial prices: $\lambda = \mathbf{1}$
- constant stepsize $\alpha_k = 3$



- Reference
  - Decomposition methods:
    decomposition_notes.pdf (reference 8 on Moodle)
    Chapter 7.6, Dimitri P. Bertsekas, Nonlinear Programming (3rd edition), Athena Scientific, 2016
- Acknowledgement
  - Some materials are extracted from the slides created by Prof. Stephen Boyd for his course EE364b in Stanford University