13. Dual decomposition

- dual gradient methods
- network rate control
- network flow optimization

Lagrange duality

convex problem (with linear componentwise inequality constraints)

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Gx \preceq h \\ & Ax = b \end{array}$$

Lagrangian and dual function

$$\begin{aligned} L(x, \lambda, \nu) &= f(x) + (G^T\lambda + A^T\nu)^T x - h^T\lambda - b^T\nu \\ g(\lambda, \nu) &= \inf_x L(x, \lambda, \nu) \end{aligned}$$

dual problem

$$\begin{array}{ll} \text{maximize} & g(\lambda, \nu) \\ \text{subject to} & \lambda \succeq 0 \end{array}$$

Lagrange dual and conjugates

dual function in terms of conjugate:

$$g(\lambda, \nu) = -h^T\lambda - b^T\nu - f^*(-G^T\lambda - A^T\nu)$$

where

$$f^*(y) = \sup_x \left(y^T x - f(x)\right)$$

potential advantages of dual methods

- dual is unconstrained or has simple constraints (depends on $\mathbf{dom}\, f^*$)
- dual is differentiable (depends on differentiability properties of f^*)
- dual (almost) decomposes into smaller problems

(Sub-)gradients of conjugate function

assume $f : \mathbf{R}^n \to \mathbf{R}$ is closed and convex with conjugate

$$f^*(y) = \sup_x \left(y^T x - f(x)\right)$$

subgradient

- any maximizer in the definition of $f^*(y)$ is a subgradient at y (page 8-6)

$$x \in \partial f^*(y) \qquad \Longleftrightarrow \qquad y^T x - f(x) = f^*(y)$$

- f^* is subdifferentiable on (at least) $\mathbf{int\,dom}\, f^*$ (page 4-6)

gradient: for f strictly convex, maximizer in definition is unique if it exists

$$\nabla f^*(y) = \operatorname*{argmax}_x \left(y^T x - f(x)\right) \quad \text{(if maximum is attained)}$$

Minimum of strongly convex function

if x is a minimizer of a strongly convex function f, then it is unique and

$$f(y) \geq f(x) + \frac{\mu}{2}\|y - x\|_2^2 \quad \forall y \in \mathbf{dom}\, f$$

(μ is the strong convexity constant of f; see page 1-9)

proof: if some y does not satisfy the inequality, then for small positive θ

$$\begin{aligned} f((1-\theta)x + \theta y) &\leq (1-\theta)f(x) + \theta f(y) - \mu\frac{\theta(1-\theta)}{2}\|y - x\|_2^2 \\ &= f(x) + \theta(f(y) - f(x) - \frac{\mu}{2}\|y - x\|_2^2) + \mu\frac{\theta^2}{2}\|x - y\|_2^2 \\ &< f(x) \end{aligned}$$

Conjugate of strongly convex function

for f closed and strongly convex, with parameter $\mu > 0$

- f^* is defined for all y ($i.e.$, $\mathbf{dom}\, f^* = \mathbf{R}^n$)

- f^* is differentiable everywhere, with gradient

$$\nabla f^*(y) = \underset{x}{\operatorname{argmax}} \left(y^T x - f(x) \right)$$

- ∇f^* is Lipschitz continuous with constant $1/\mu$

$$\|\nabla f^*(u) - \nabla f^*(v)\|_2 \leq \frac{1}{\mu} \|u - v\|_2$$

outline of proof

$$f^*(y) = \sup_x \left(y^T x - f(x)\right)$$

- $y^T x - f(x)$ has a unique maximizer for every y

  (follows from closedness of f and strong convexity of $f(x) - y^T x$)

- from page 13-4: $\nabla f^*(y) = \mathrm{argmax}_x \left(y^T x - f(x)\right)$

- from strong convexity and page 13-5 (with $x_u = \nabla f^*(u)$, $x_v = \nabla f^*(v)$)

$$\begin{aligned} f(x_u) - v^T x_u &\geq f(x_v) - v^T x_v + \frac{\mu}{2}\|x_u - x_v\|_2^2 \\ f(x_v) - u^T x_v &\geq f(x_u) - u^T x_u + \frac{\mu}{2}\|x_u - x_v\|_2^2 \end{aligned}$$

  combining the inequalities gives $\mu\|x_u - x_v\|_2^2 \leq (x_u - x_v)^T(u - v)$

- apply the Cauchy-Schwarz inequality to get $\mu\|x_u - x_v\|_2 \leq \|u - v\|_2$

Dual gradient method

primal problem (only equality constraints, for simplicity)

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax = b \end{array}$$

dual problem: maximize $g(\nu)$ where

$$g(\nu) = \inf_x \left(f(x) + (Ax - b)^T \nu \right) = -b^T \nu - f^*(-A^T \nu)$$

dual ascent: solve dual by (sub-)gradient method

$$x^+ = \operatorname*{argmin}_x \left(f(x) + \nu^T Ax \right), \qquad \nu^+ = \nu + t(Ax^+ - b)$$

- sometimes referred to as Uzawa's method
- of interest if calculation of x^+ is inexpensive

Dual decomposition

convex problem with separable objective

$$\begin{array}{ll} \text{minimize} & f_1(x_1) + f_2(x_2) \\ \text{subject to} & G_1 x_1 + G_2 x_2 \preceq h \end{array}$$

constraint is *complicating* (or *coupling*) constraint

dual problem

$$\begin{array}{ll} \text{maximize} & g_1(\lambda) + g_2(\lambda) - h^T \lambda \\ \text{subject to} & \lambda \succeq 0 \end{array}$$

where $g_j(\lambda) = \inf_x \left(f_j(x) + \lambda^T G_j x \right) = -f_j^*(-G_j^T \lambda)$

can be solved by (sub-)gradient projection if $\lambda \succeq 0$ is the only constraint

subproblem: to calculate $g_j(\lambda)$ and a (sub-)gradient, solve the problem

$$\text{minimize (over } x_j\text{)} \quad f_j(x_j) + \lambda^T G_j x_j$$

- optimal value is $g_j(\lambda)$
- if $\hat{x}_j$ solves the subproblem, then $-G_j\hat{x}_j$ is a subgradient of $-g_j$ at λ

dual subgradient projection method

- solve two unconstrained (and independent) subproblems

$$x_j^+ = \underset{x_j}{\operatorname{argmin}} \left(f_j(x_j) + \lambda^T G_j x_j \right), \quad j = 1, 2$$

- make projected subgradient update of λ

$$\lambda^+ = \left(\lambda + t(G_1 x_1^+ + G_2 x_2^+ - h) \right)_+$$

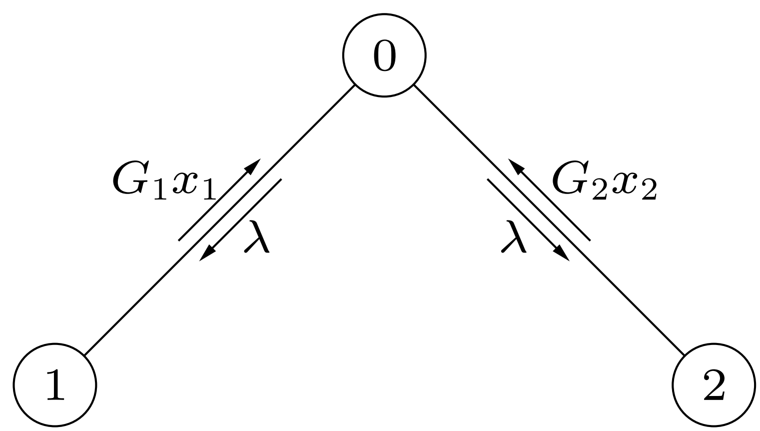
($u_+ = \max\{u, 0\}$, component-wise)

interpretation: price coordination

- $p = 2$ units in a system; unit j chooses decision variable x_j
- constraints are limits on shared resources; λ_i is price of resource i
- dual update $\lambda_i^+ = (\lambda_i - ts_i)_+$ depends on slacks $s = h - G_1x_1 - G_2x_2$
  - increases price λ_i if resource is over-utilized ($s_i < 0$)
  - decreases price λ_i if resource is under-utilized ($s_i > 0$)
  - never lets prices get negative

distributed architecture

- central node 0 sets prices λ
- peripheral node j sets x_j



Quadratic programming example

$$
\begin{array}{ll}
\text{minimize} & \frac{1}{2}\sum_{j=1}^{r} x_j^T P_j x_j + q_j^T x_j \\
\text{subject to} & A_j x_j \preceq b_j \\
& \sum_{j=1}^{p} G_j x_j \preceq h
\end{array}
$$

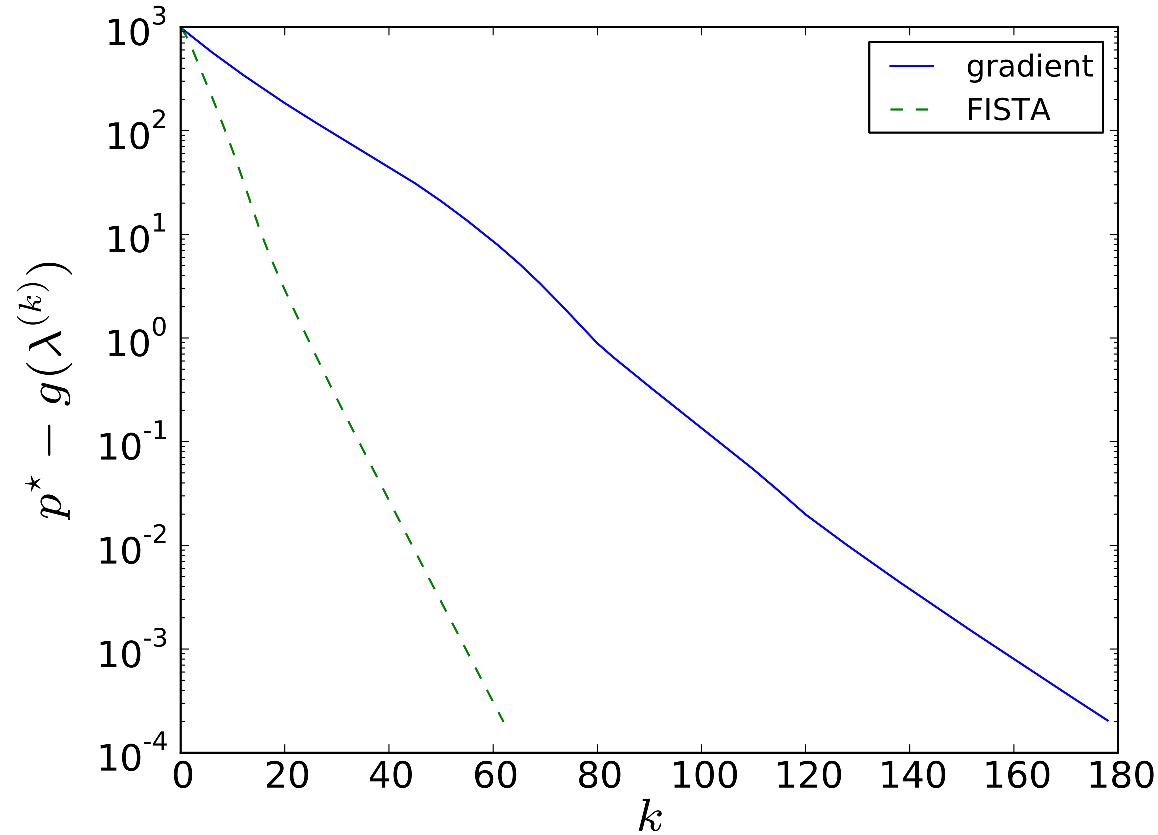
- $r = 10$; variables $x_j \in \mathbf{R}^{100}$; $A_j \in \mathbf{R}^{100\times 100}$, $G_j \in \mathbf{R}^{10\times 100}$
- $P_j \succ 0$; implies dual function has Lipschitz continuous gradient

subproblems are QPs

$$
\begin{array}{ll}
\text{minimize (over } x_j\text{)} & (1/2)x_j^T P_j x_j + (q_j + G_j^T \lambda)^T x_j \\
\text{subject to} & A_j x_j \preceq b_j
\end{array}
$$

gradient projection and fast gradient projection

- fixed step size (equal in the two methods)
- plot shows convergence of master problem



Outline

- dual gradient methods
- **network rate control**
- network flow optimization

Network rate control

network flows

- n flows, with fixed routes, in a network with m links
- variable $x_j \geq 0$ denotes the rate of flow j
- flow utility is $U_j : \mathbf{R} \to \mathbf{R}$, concave, increasing

capacity constraints

- traffic y_i on link i is sum of flows passing through it
- $y = Rx$, 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: $y \preceq c$

Rate control problem

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

a convex problem; dual decomposition gives decentralized method

Lagrangian (for minimizing $-U$)

$$\begin{aligned} L(x,\lambda) &= -U(x) + \lambda^T(Rx - c) \\ &= -\lambda^T c + \sum_{j=1}^{n} \left(-U_j(x_j) + (r_j^T\lambda)x_j\right) \end{aligned}$$

- λ_i is price (per unit flow) for using link i
- $r_j^T\lambda$ is the sum of prices along route j (r_j is jth column of R)

dual function

$$\begin{aligned} g(\lambda) &= -\lambda^T c + \sum_{j=1}^{n} \inf_{x_j} \left(-U_j(x_j) + x_j r_j^T \lambda\right) \\ &= -\lambda^T c - \sum_{j=1}^{n} (-U_j)^*(-r_j^T \lambda) \end{aligned}$$

dual rate control problem

$$\begin{array}{ll} \text{maximize} & -\lambda^T c - \sum_{j=1}^{n} (-U_j)^*(-r_j^T \lambda) \\ \text{subject to} & \lambda \succeq 0 \end{array}$$

(Sub-)gradients of dual function

$$-g(\lambda) = \lambda^T c + \sum_{j=1}^{n} \sup_{x_j} \big(U_j(x_j) - x_j r_j^T \lambda\big)$$

- subgradient of $-g(\lambda)$

$$c - R\bar{x} \in \partial(-g)(\lambda) \quad \text{where} \quad \bar{x}_j = \operatorname{argmax} \big(U_j(x_j) - x_j r_j^T \lambda\big)$$

  if U_j is strictly concave, this is a gradient

- $r_j^T \lambda$ is the sum of link prices along route j

- $c - R\bar{x}$ is vector of link capacity margins for flow $\bar{x}$

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 $z_j = r_j^T \lambda$
2. optimize flows (separately) using flow prices

$$x_j := \operatorname{argmax} \left(U_j(x_j) - z_j x_j \right)$$

3. calculate link capacity margins $s := c - Rx$
4. update link prices using projected (sub-)gradient step with step t

$$\lambda := (\lambda - ts)_+$$

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

Generating feasible flows

primal iterates are not necessarily feasible ($i.e.$, $Rx \not\preceq c$)

- define $\eta_i = (Rx)_i / c_i$

  $\eta_i < 1$ means link i is under capacity; $\eta_i > 1$ means link is over capacity

- define x^{feas} as

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

  x^{feas} is feasible, even if x is not

- finding x^{feas} is also decentralized

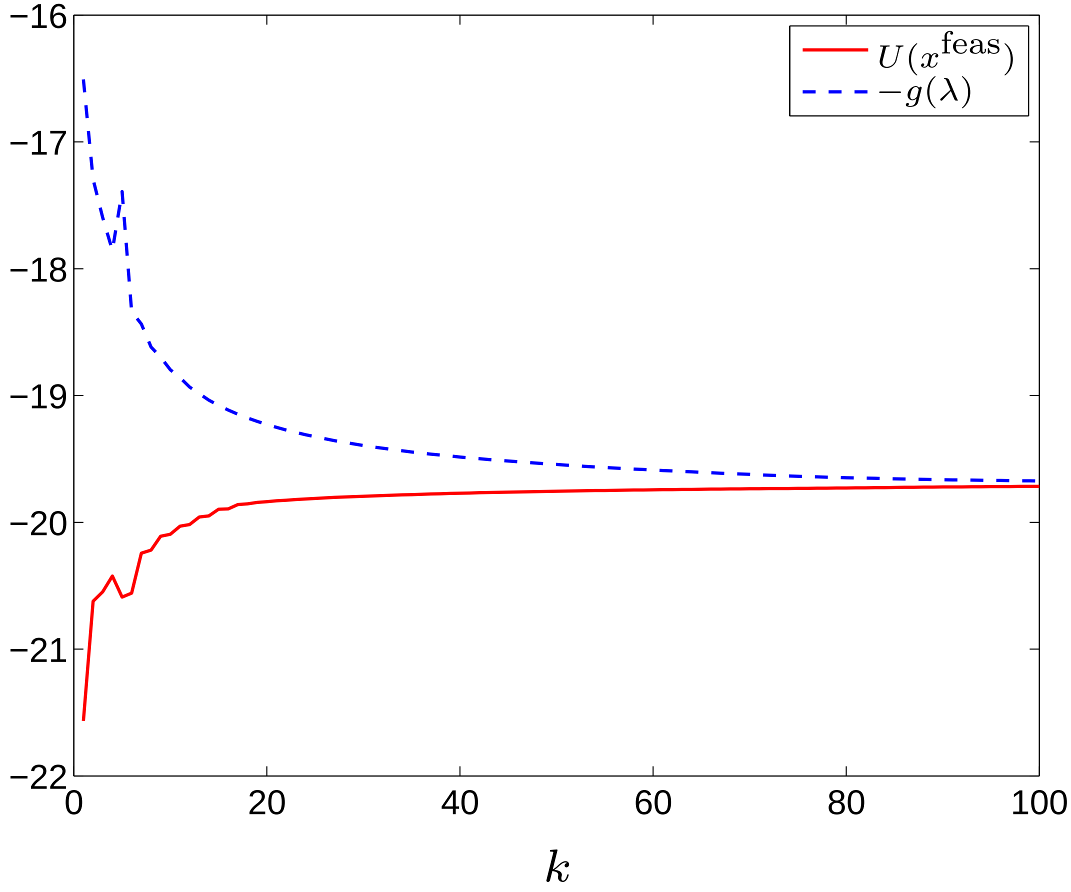
Example

- $n = 10$ flows, $m = 12$ links; 3 or 4 links per flow
- link capacities chosen randomly, uniform on $[0.1, 1]$
- $U_j(x_j) = \log x_j$; optimal flow as a function of price is

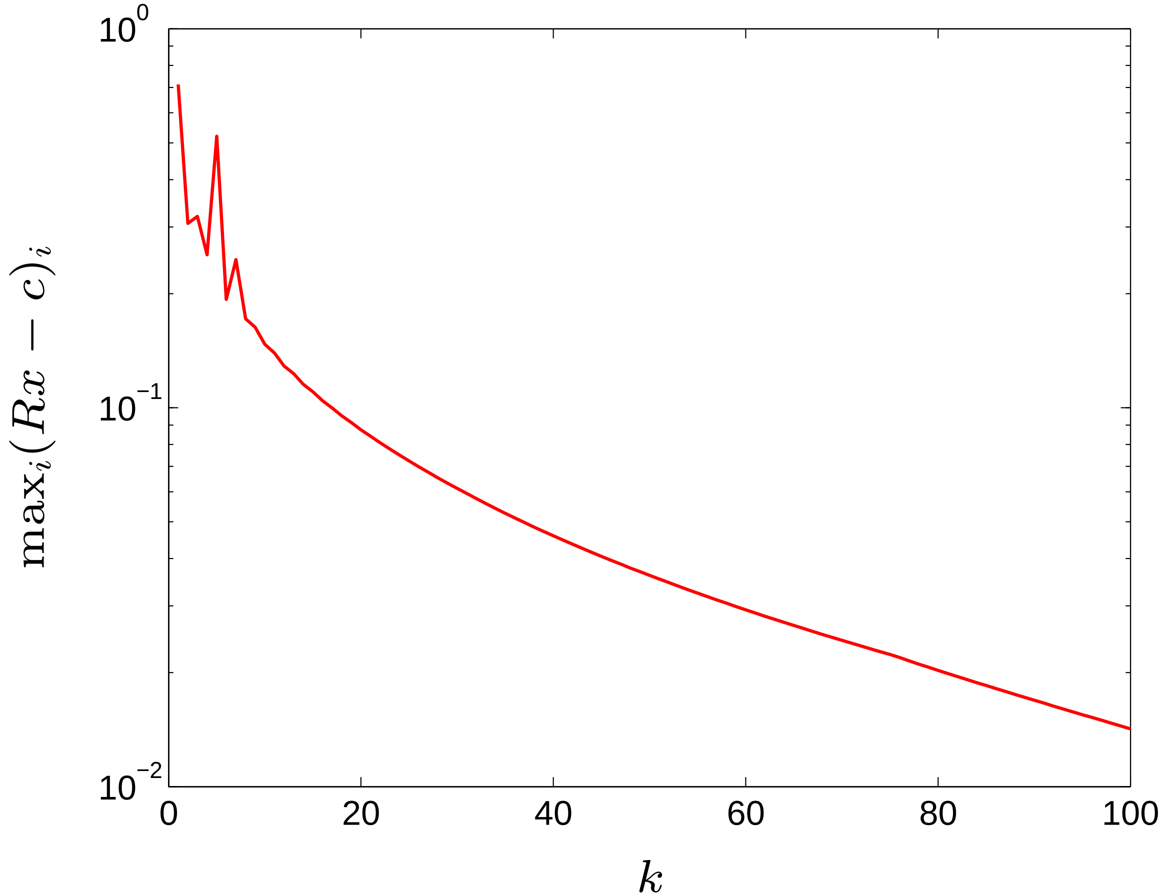
$$\bar{x}_j = \operatorname{argmax} \left(U_j(x_j) - z_j x_j \right) = \frac{1}{z_j}$$

- initial prices: $\lambda = \mathbf{1}$
- constant stepsize $t_k = 3$

Convergence of primal and dual objectives



Maximum capacity violation



Outline

- dual gradient methods
- network rate control
- **network flow optimization**

Single commodity network flow

network

- connected, directed graph with n links, p nodes
- node incidence matrix $A \in \mathbf{R}^{p \times n}$ is

$$A_{ij} = \begin{cases} 1 & \text{arc } j \text{ enters } i \\ -1 & \text{arc } j \text{ leaves node } i \\ 0 & \text{otherwise} \end{cases}$$

flow vector and external sources

- variable x_j denotes flow (traffic) on arc j
- given external source (or sink) flow b_i at node i, $\mathbf{1}^T b = 0$
- flow conservation: $Ax + b = 0$

Network flow optimization problem

$$
\begin{array}{ll}
\text{minimize} & \sum_{j=1}^{n} \phi_j(x_j) \\
\text{subject to} & Ax + b = 0
\end{array}
$$

$\phi(x) = \sum_{j=1}^{n} \phi_j(x_j)$ is separable convex flow cost function

- convex, readily solved with standard methods
- dual decomposition yields decentralized solution method

Network flow dual

Lagrangian

$$\begin{aligned} L(x,\nu) &= \phi(x) + \nu^T(Ax+b) \\ &= b^T\nu + \sum_{j=1}^n \left(\phi_j(x_j) + x_j a_j^T \nu\right) \end{aligned}$$

- a_j is jth column of A
- dual variable ν_i can be interpreted as as potential at node i
- $y_j = -a_j^T\nu$ is the potential difference across edge j (potential at start node minus potential at end node)

dual problem: maximize $g(\nu)$

$$g(\nu) = \inf_x L(x,\nu) = b^T\nu - \sum_{j=1}^n \phi_j^*(-a_j^T\nu)$$

Recovering primal from dual

assume cost functions ϕ_j are strictly convex

- strictly convex ϕ_j means unique minimizer

$$\hat{x}_j(y) = \underset{x_j}{\operatorname{argmin}} \left(\phi_j(x_j) - y_j x_j\right)$$

- gradient of $-g$ at ν is:

$$-(A\hat{x}(y) + b) \quad \text{where} \quad y = -A^T \nu$$

  gradient is negative of flow conservation residual

- if ϕ_j is differentiable, $\hat{x}_j(y) = (\phi_j')^{-1}(y_j)$ (inverse of derivative function)

- optimal flows, from optimal potentials, are $\hat{x}_j(y_j^\star)$ where $y^\star = -A^T \nu^\star$

Dual decomposition network flow algorithm

given initial potential vector ν

repeat

1. determine link flows from potential differences $y = -A^T\nu$

$$x_j := \hat{x}_j(y_j), \quad j = 1, \ldots, n$$

2. compute flow surplus at each node: $s_i := a_i^T x + b_i$, $i = 1, \ldots, p$
3. update node potentials using (sub-)gradient step with step size t

$$\nu_i := \nu_i + ts_i, \quad i = 1, \ldots, p$$

decentralized: flow is calculated from potential difference across edge; node potential is updated from its own flow surplus

Electrical network interpretation

network flow optimality conditions (with differentiable ϕ_j)

$$Ax + b = 0, \qquad y + A^T\nu = 0, \qquad y_j = \phi_j'(x_j), \quad j = 1, \ldots, n$$

network with node incidence matrix A, nonlinear resistors in branches

Kirchhoff current law (KCL): $Ax + b = 0$

x_j is the current flow in branch j; b_i is external current injected at node i

Kirchhoff voltage law (KVL): $y + A^T\nu = 0$

ν_j is node potential; $y_j = -a_j^T\nu$ is jth branch voltage

current-voltage characterics: $y_j = \phi_j'(x_j)$

for example, $\phi_j(x_j) = R_j x_j^2/2$ for linear resistor R_j

current and potentials in circuit are optimal flows and dual variables

Example: minimum queueing delay

flow cost function

$$\phi_j(x_j) = \frac{x_j}{c_j - x_j}, \qquad \mathbf{dom}\,\phi_j = [0, c_j)$$

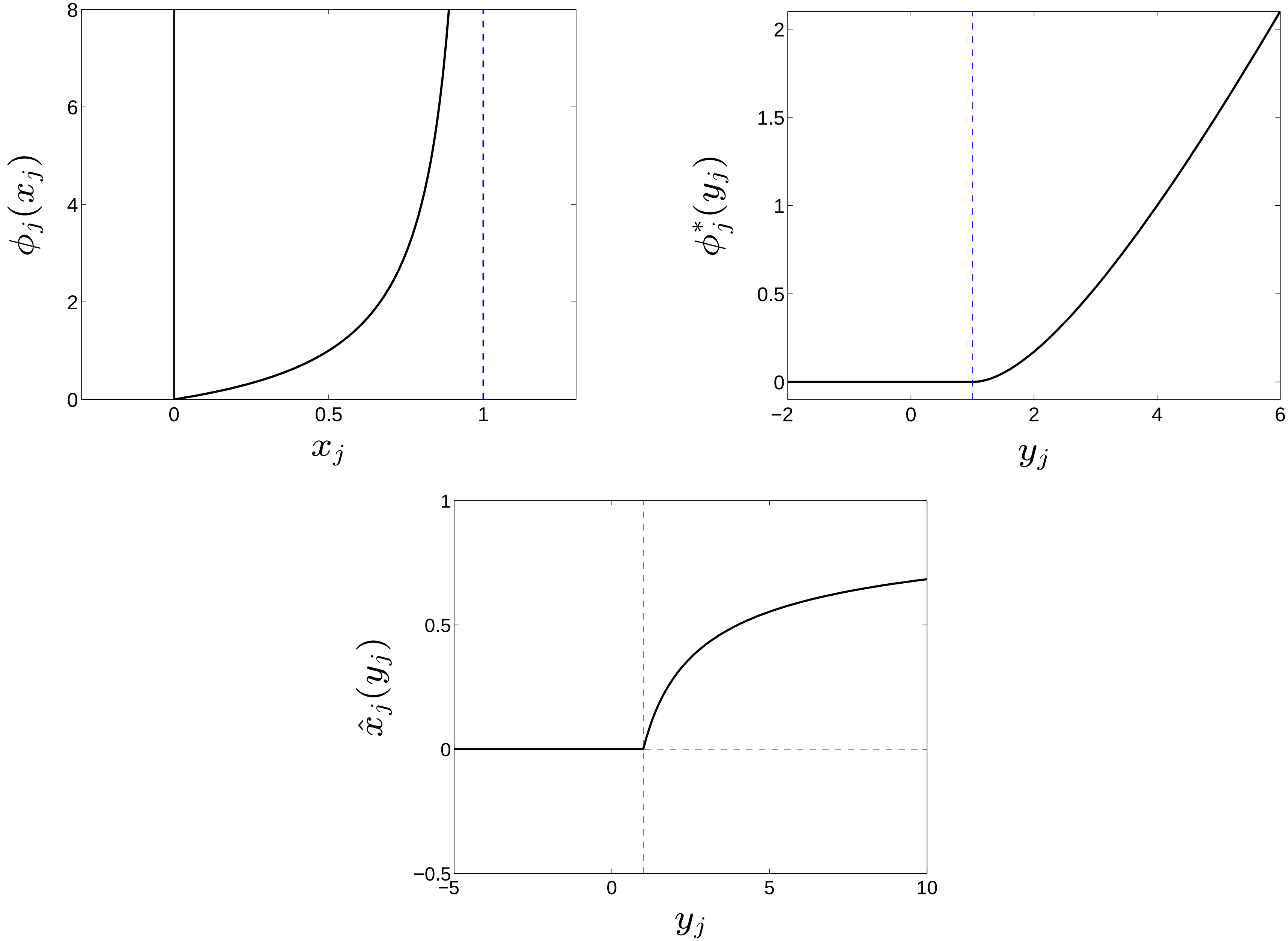
where $c_j > 0$ are given *link capacities*

conjugate

$$\phi_j^*(y_j) = \begin{cases} \left(\sqrt{c_j y_j} - 1\right)^2 & y_j > 1/c_j \\ 0 & y_j \le 1/c_j \end{cases}$$

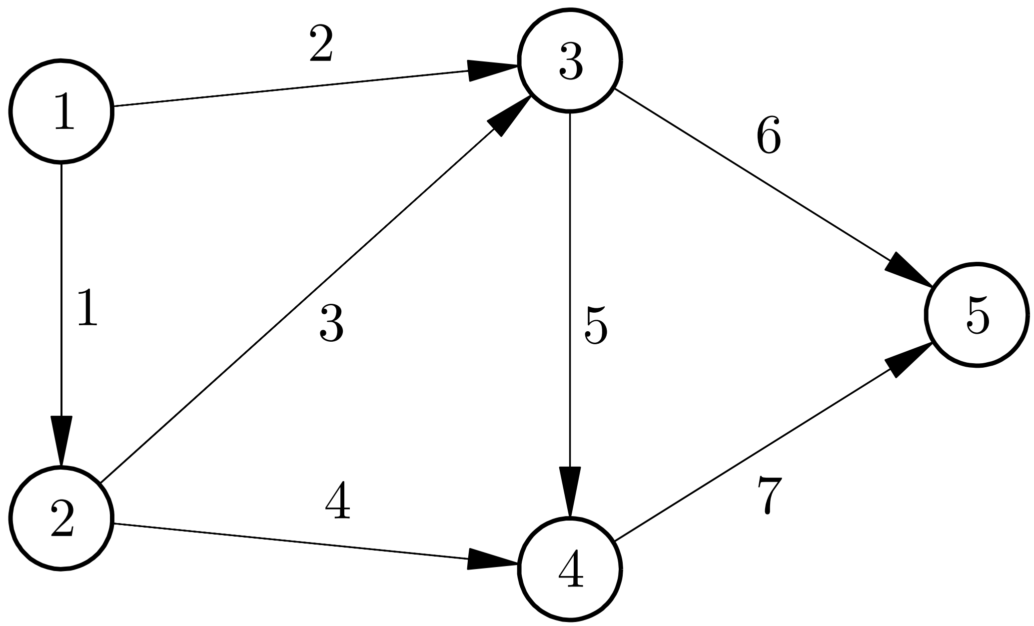
inverse derivative map

$$\phi_j'(x_j) = y_j \quad \Longleftrightarrow \quad x_j = \hat{x}_j(y_j) = c_j - \sqrt{c_j/y_j}$$


$\phi_j(x_j)$
x_j
0
2
4
6
8
0
0.5
1
$\phi_j^*(y_j)$
y_j
0
0.5
1
1.5
2
−2
0
2
4
6
$\hat{x}_j(y_j)$
y_j
−0.5
0
0.5
1
−5
0
5
10

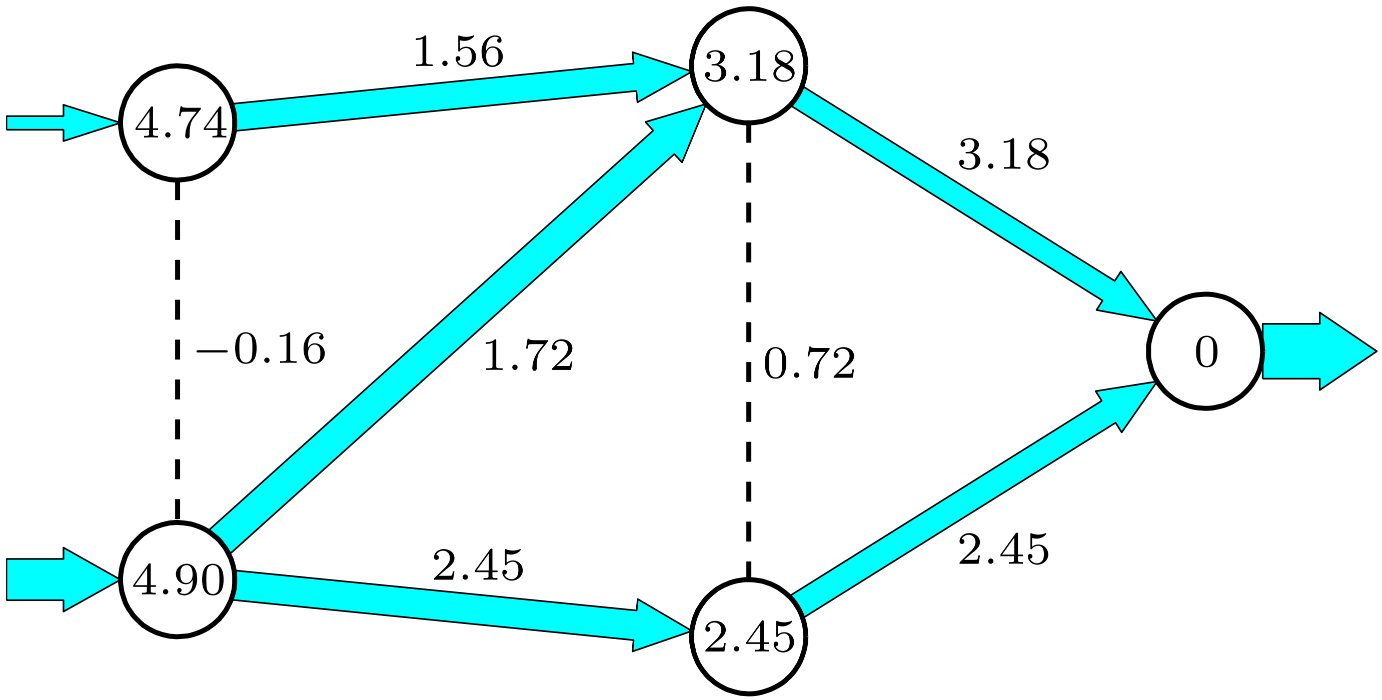
A specific example

network with 5 nodes, 7 links, capacities $c_j = 1$



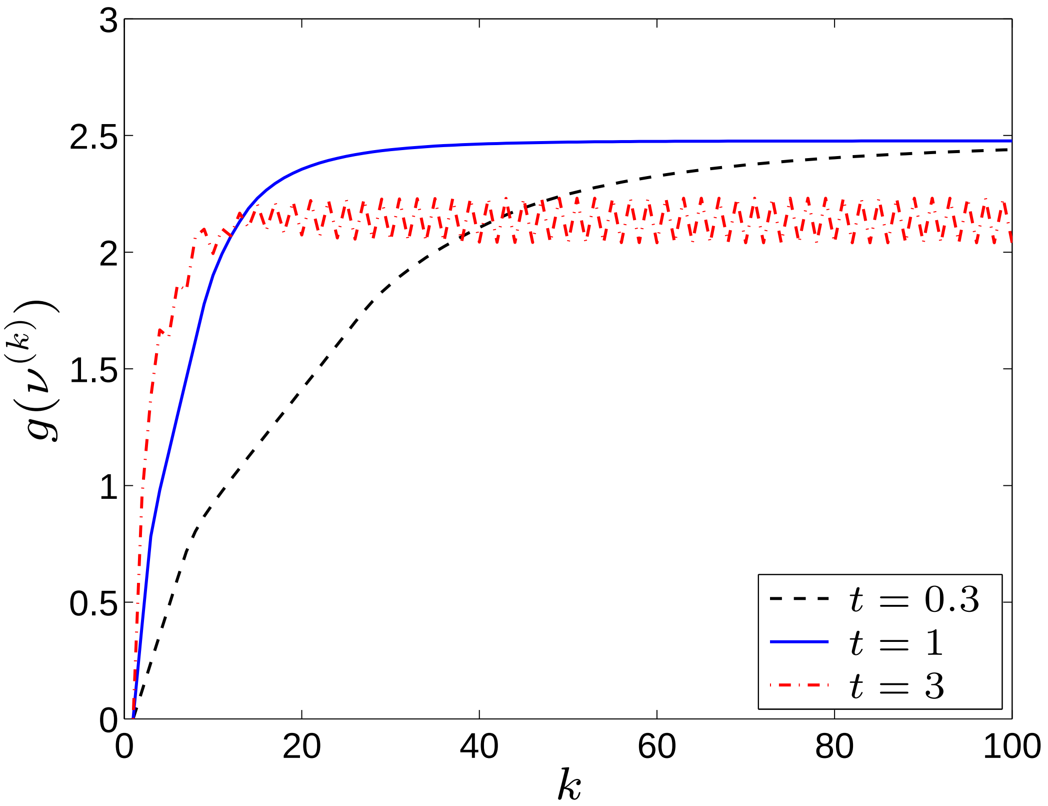
Optimal flow

optimal flows shown as width of arrows; optimal dual variables shown in nodes; potential differences shown on links



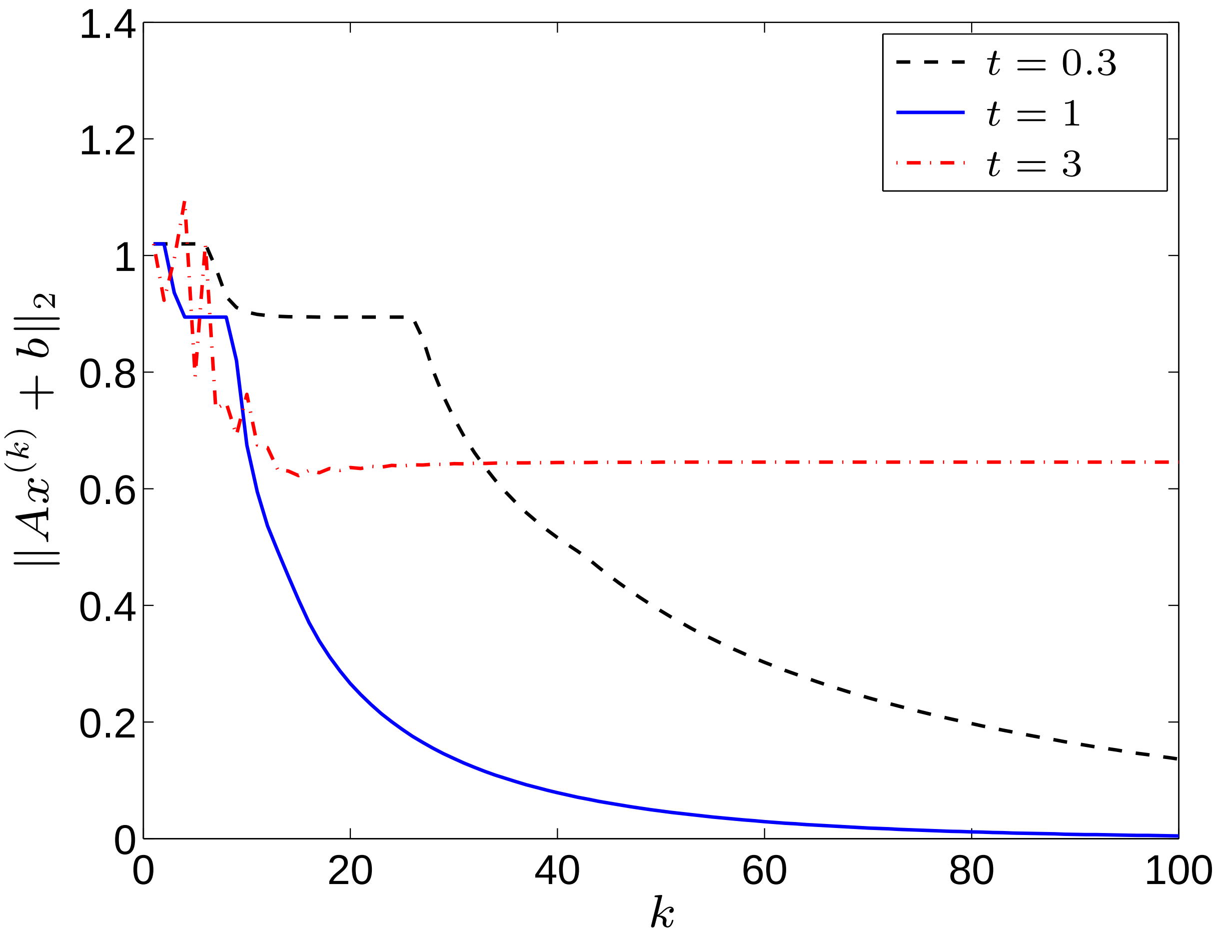
Convergence of dual function

fixed step size rules, $t = 0.3,\ 1,\ 3$



for $t = 1$, converges to $p^\star = 2.48$ in around 40 iterations

Convergence of primal residual



References

- S. Boyd, course notes for EE364b, Convex Optimization II

  the rate control and network flow examples are from EE364b lecture 9 and the notes on decomposition
- M. Chiang, S.H. Low, A.R. Calderbank, J.C. Doyle, *Layering as optimization decomposition: A mathematical theory of network architectures*, Proceedings IEEE (2007)
- D.P. Bertsekas and J.N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods* (1989)
- D.P. Bertsekas, *Network Optimization. Continuous and Discrete Models* (1998)
- L.S. Lasdon, *Optimization Theory for Large Systems* (1970)