Decomposition Applications

- rate control
- single commodity network flow

Rate control 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: $t \preceq c$

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

Rate control Lagrangian

Lagrangian (for minimizing $-U$) is

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

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

Rate control dual

dual function is

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

dual rate control problem:

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

subgradient of negative dual:

$$R\bar{f} - c \in \partial(-g)(\lambda)$$

where $\bar{f}_j = \operatorname{argmax}\left(U_j(f_j) - (r_j^T\lambda)f_j\right)$

Dual decomposition rate control algorithm

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

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.
   $\lambda := (\lambda - \alpha_k s)_+$.

Dual decomposition rate control algorithm

- decentralized:
  - links only need to know the flows that pass through them
  - flows only need to know prices on links they pass through
- prices converge to optimal; so do flows (since U is strictly concave)
- iterates can be (and often are) infeasible, *i.e.*, $Rf \not\preceq c$
  (but we do have $Rf \preceq c$ in the limit)
- have upper bound $-g(\lambda)$ on optimal utility $U^\star$

Generating feasible flows

- 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

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

- f^{feas} will be feasible, even if f is not

- finding f^{feas} is also decentralized
  (in fact this is a step in primal decomposition)

Example

- $n = 10$ flows, $m = 12$ links; 3 or 4 links per flow

- link capacities chosen randomly, uniform on $[0.1, 1]$

- $U_j(f_j) = \log f_j$ (can be argued to give proportionally fair flows)

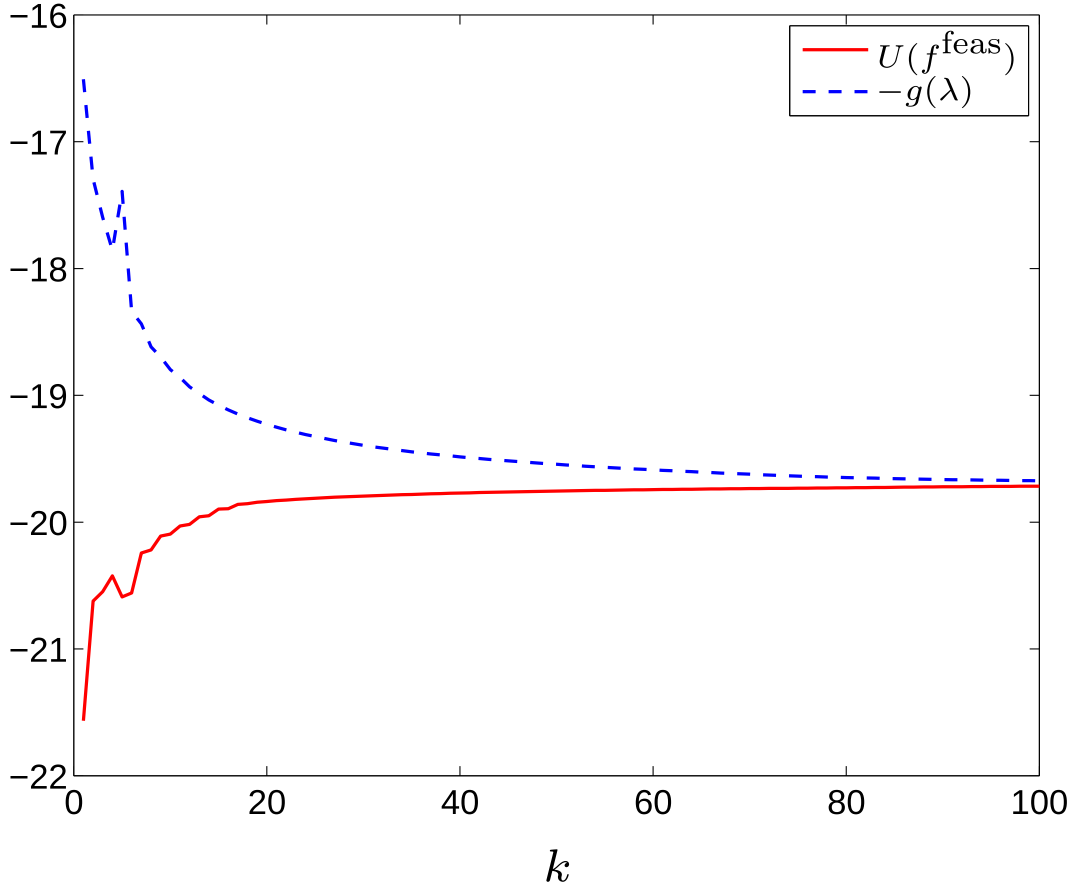
- optimal flow as a function of price:

$$\bar{f}_j = \operatorname{argmax}(U_j(f_j) - \Lambda_j f_j) = 1/\Lambda_j$$

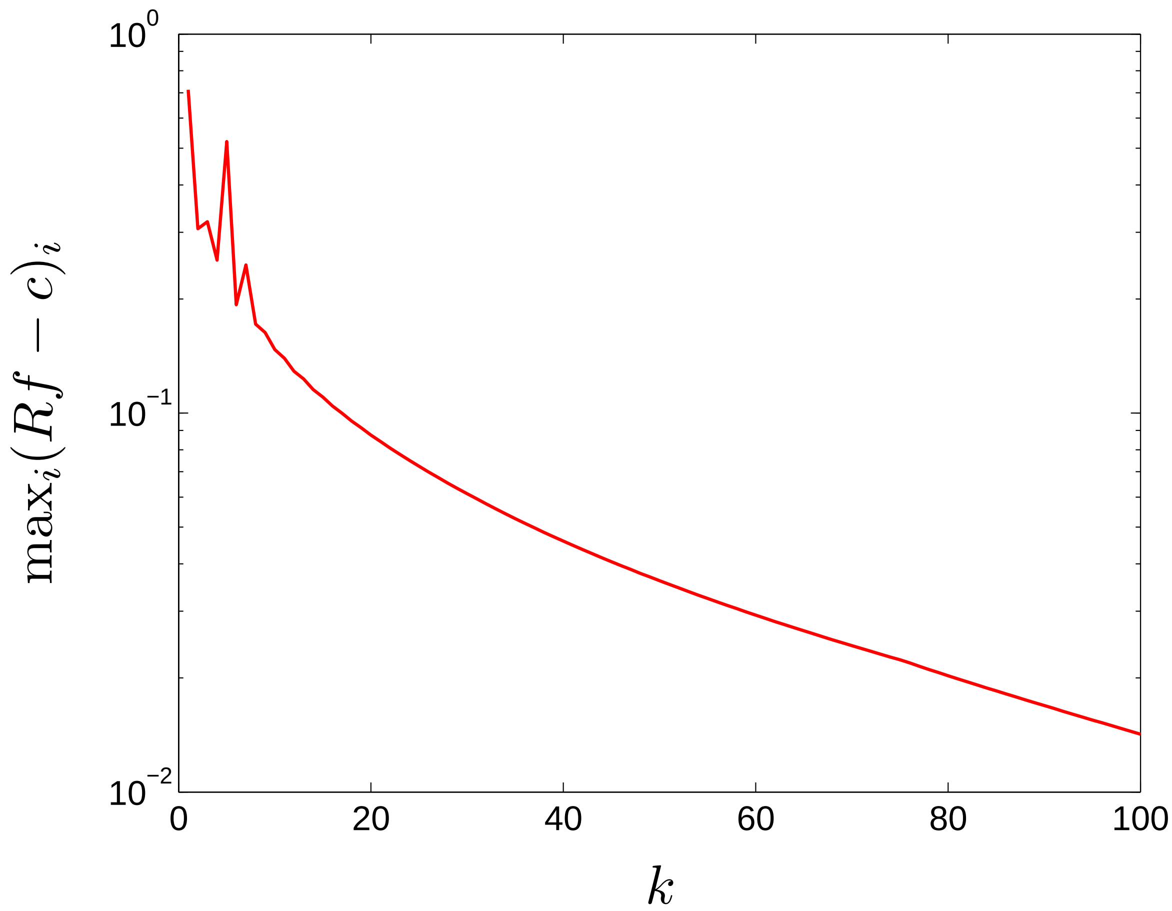
- initial prices: $\lambda = \mathbf{1}$

- constant stepsize $\alpha_k = 3$

Convergence of primal and dual objectives



Maximum capacity violation



Single commodity network flow setup

- connected, directed graph with n links, p nodes
- variable x_j denotes flow (traffic) on arc j
- given external source (or sink) flow s_i at node i, $\mathbf{1}^T s = 0$
- 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 conservation: $Ax + s = 0$
- $\phi(x) = \sum_{j=1}^{n} \phi_j(x_j)$ is separable convex flow cost function

Network flow problem

optimal single commodity network flow problem:

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

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

Network flow Lagrangian

Lagrangian is

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

- a_j is jth column of A
- we'll interpret ν_i as potential at node i
- we use $\Delta\nu_j$ to denote $a_j^T\nu$, which is potential difference across edge j

Network flow dual

dual function:

$$\begin{aligned} g(\nu) &= \inf_x L(x,\nu) \\ &= \nu^T s + \sum_{j=1}^n \inf_{x_j} \left(\phi_j(x_j) + (\Delta\nu_j)x_j\right) \\ &= \nu^T s - \sum_{j=1}^n \phi_j^*(-\Delta\nu_j) \end{aligned}$$

dual problem: maximize $g(\nu)$

Recovering primal from dual

- strictly convex ϕ_j means unique minimizer $x_j^*(y)$ of $\phi_j(x_j) - yx_j$
- if ϕ_j is differentiable, $x_j^*(y) = (\phi_j')^{-1}(y)$ (inverse of derivative function)
- optimal flows, from optimal potentials: $x_j^\star = x_j^*(-\Delta\nu_j^\star)$
- subgradient of negative dual function:

$$-(Ax^*(\Delta\nu) + s) \in \partial(-g)(\nu)$$

(negative of flow conservation residual)

Dual decomposition network flow algorithm

given initial potential vector ν.
repeat

1. Determine link flows from potential differences.
   $x_j := x_j^*(-\Delta\nu_j), \quad j = 1, \ldots, n.$
2. Compute flow surplus at each node.
   $S_i := a_i^T x + s_i, \quad i = 1, \ldots, p.$
3. Update node potentials.
   $\nu_i := \nu_i + \alpha_k S_i, \quad i = 1, \ldots, p.$

α_k is an appropriate step size

Dual decomposition network flow algorithm

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

- $g(\nu)$ gives lower bound on $p^\star$

- flow conservation $Ax + s = 0$ only holds in limit

Electrical network analogy

- electrical network with node incidence matrix A, nonlinear resistors in branches
- variable x_j is the current flow in branch j
- source s_i is external current injected at node i (must sum to zero)
- flow conservation equation $Ax + s = 0$ is Kirkhoff Current Law (KCL)
- dual variables are node potentials; $\Delta\nu_j$ is jth branch voltage
- branch current-voltage characteristic is $x_j = x_j^*(-\Delta\nu_j)$

then, current and potentials in circuit are optimal flows and dual variables

Example: Minimum queueing delay

flow cost function

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

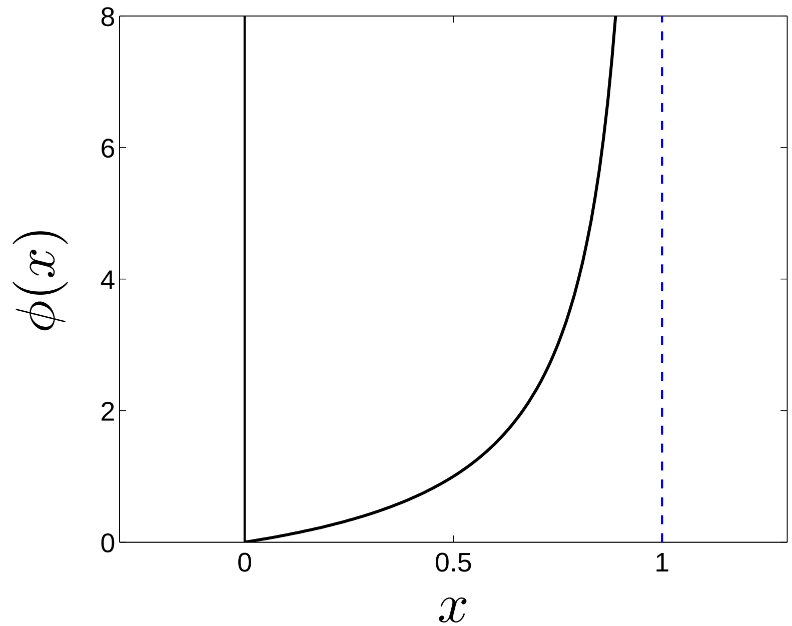
where $c_j > 0$ are given *link capacities*

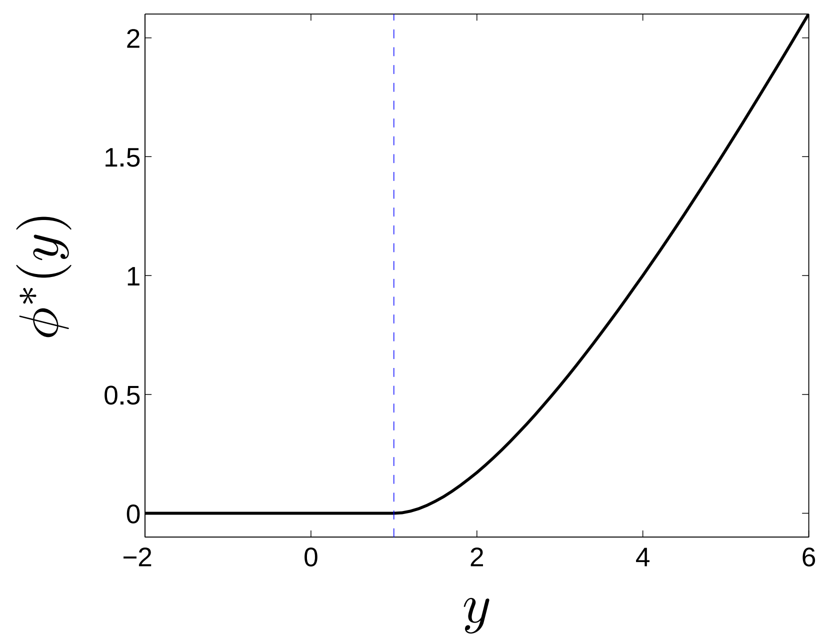
($\phi_j(x_j)$ gives expected waiting time in queue with exponential arrivals at rate x_j, exponential service at rate c_j)

conjugate is

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

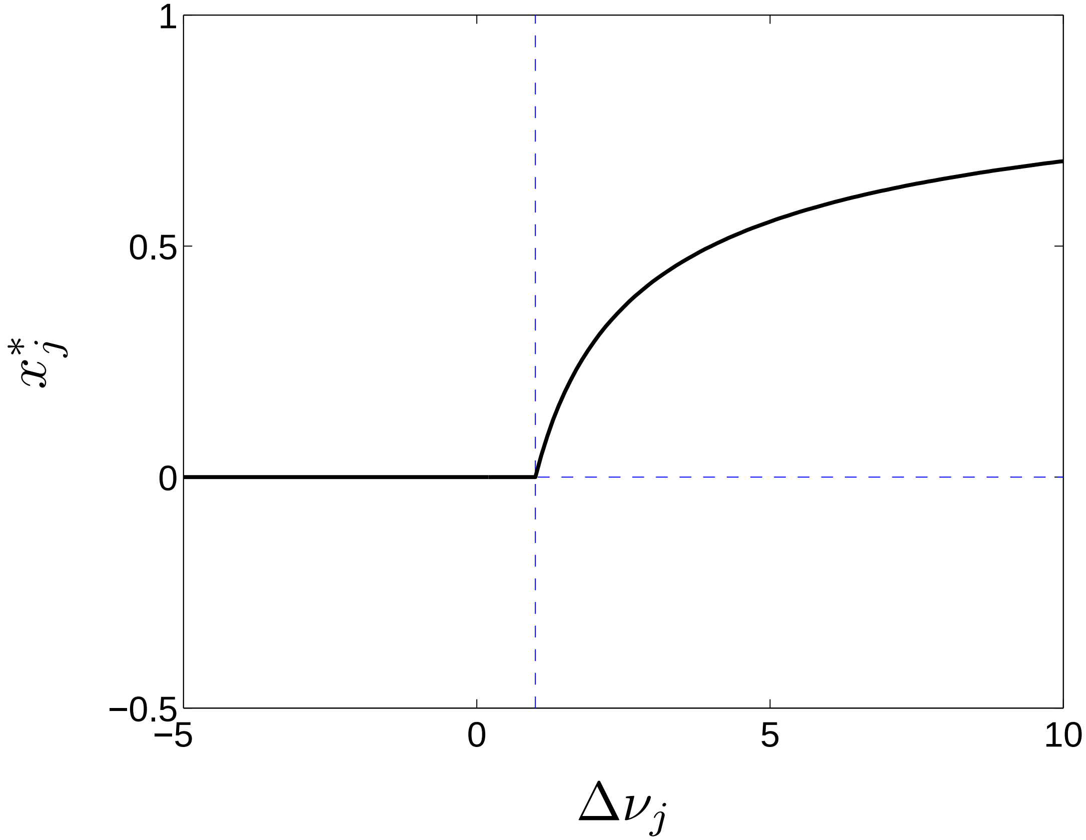
cost function $\phi(x)$ (left) and its conjugate $\phi^*(y)$ (right), $c = 1$





(note that conjugate is differentiable)

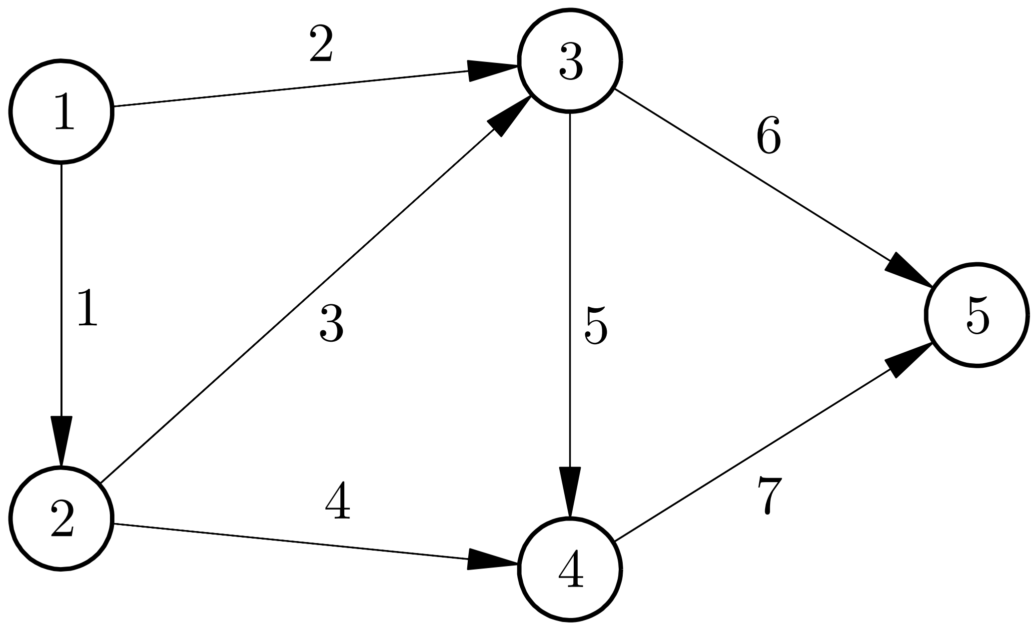
$x_j^*(-\Delta\nu_j)$, for $c_j = 1$



gives flow as function of potential difference across link

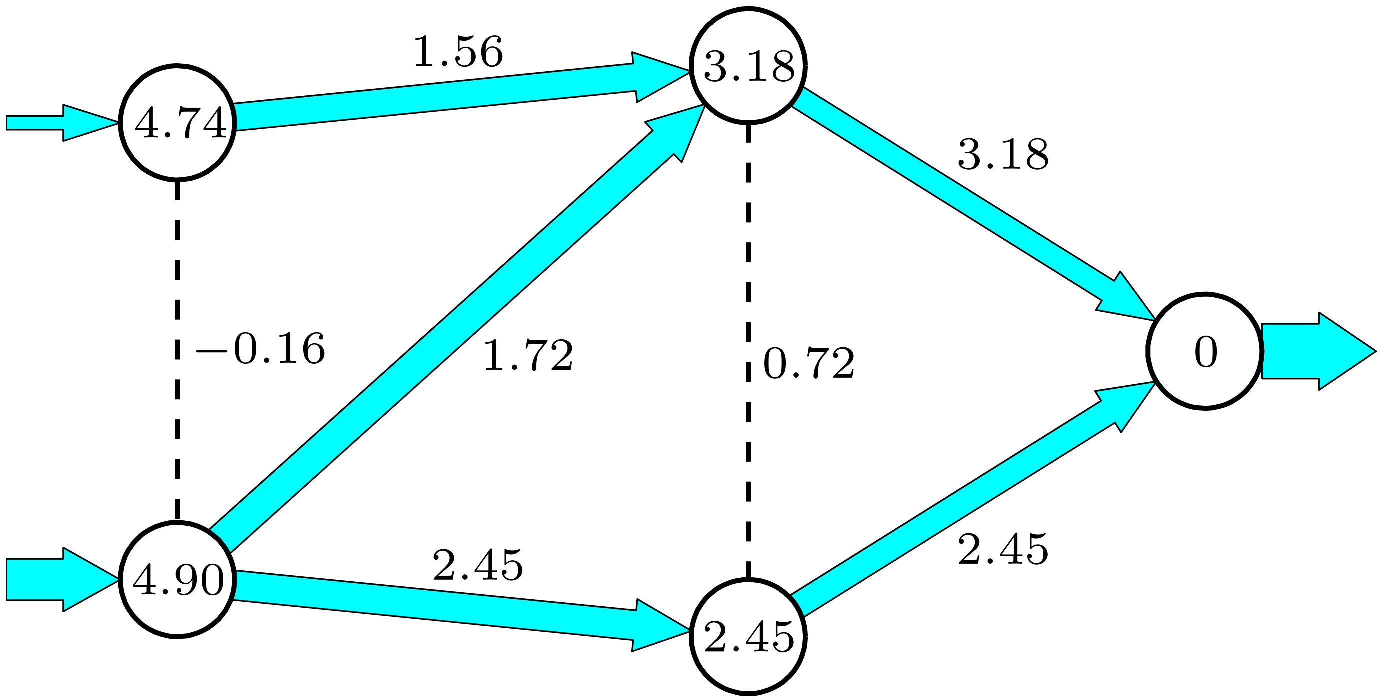
A specific example

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



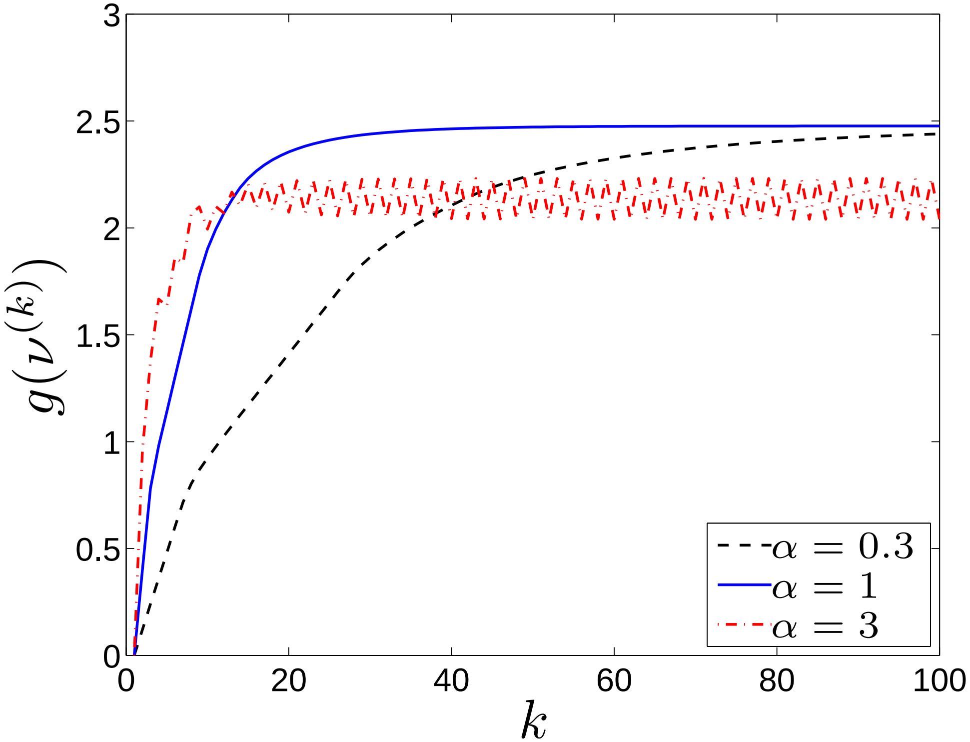
Optimal flow

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



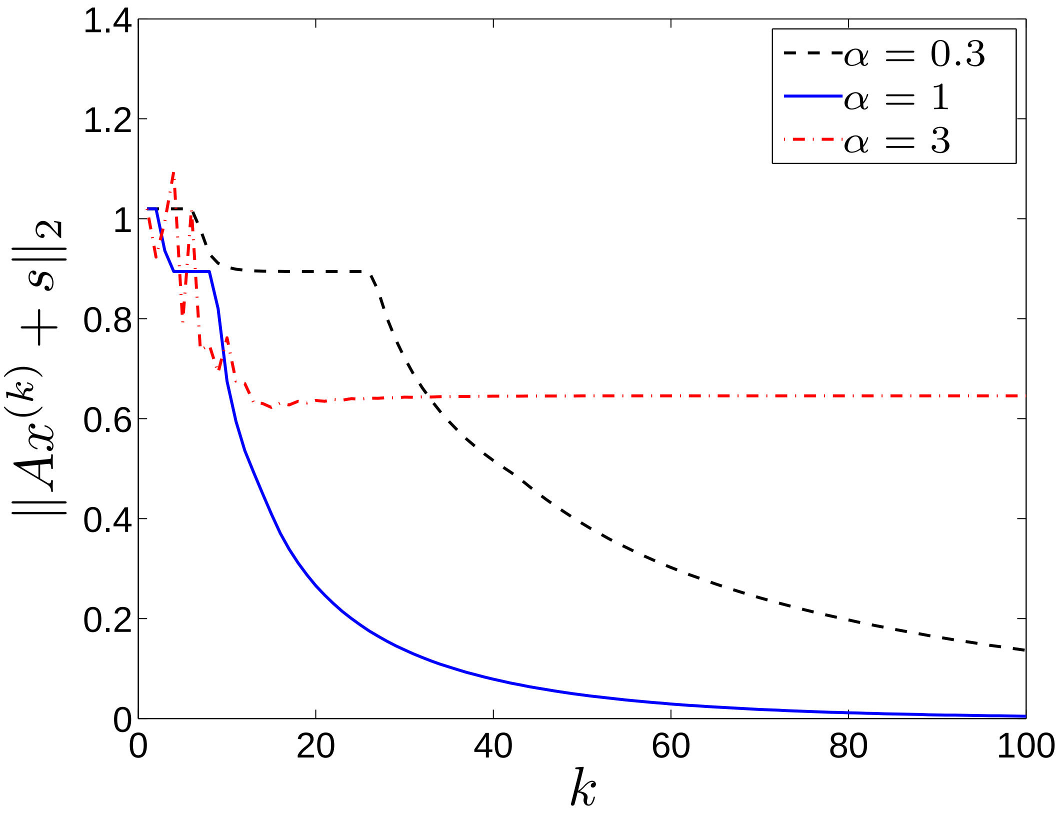
Convergence of dual function

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



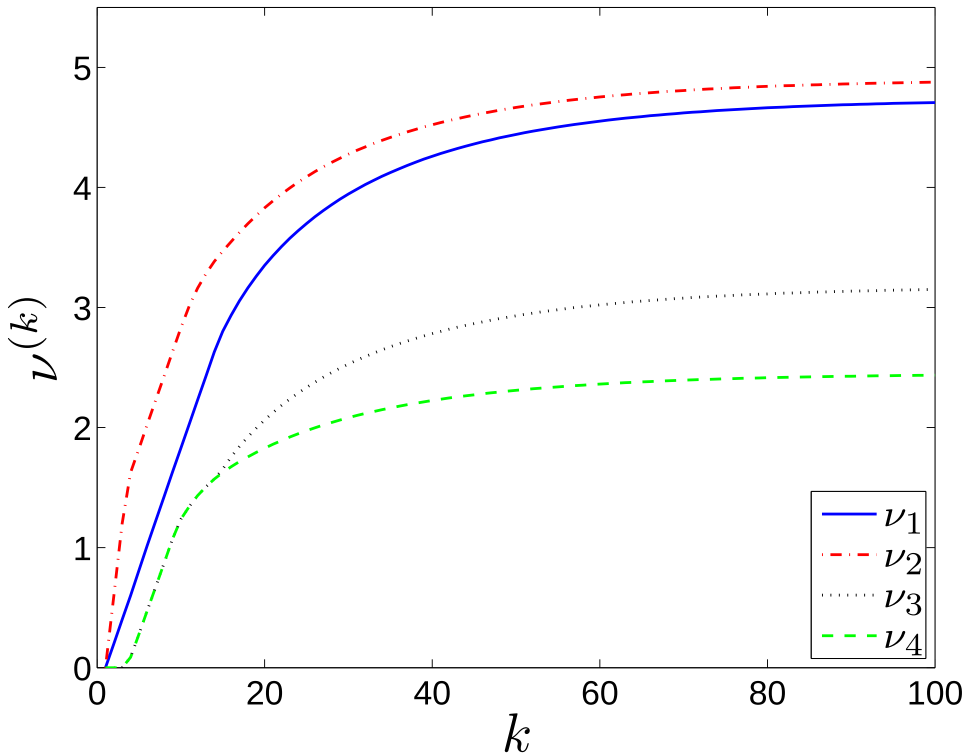
for $\alpha = 1$, converges to $p^\star = 2.48$ in around 40 iterations

Convergence of primal residual



Convergence of dual variables

$\nu^{(k)}$ versus iteration number k, fixed step size rule $\alpha = 1$



(ν_5 is fixed as zero)