11. Dual decomposition

- introduction: dual methods
- gradient and subgradient of conjugate
- dual decomposition
- network utility maximization
- network flow optimization

Duality and conjugates

primal problem ($A \in \mathbf{R}^{m\times n}$, f and g convex)

$$\text{minimize} \quad f(x) + g(Ax)$$

Lagrangian (after introducing new variable $y = Ax$)

$$f(x) + g(y) + z^T(Ax - y)$$

dual function

$$\inf_x \left(f(x) + z^T Ax\right) + \inf_y \left(g(y) - z^T y\right) = -f^*(-A^T z) - g^*(z)$$

dual problem

$$\text{maximize} \quad -f^*(-A^T z) - g^*(z)$$

Examples

equality constraints: g is indicator for $\{b\}$

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax = b \end{array} \qquad\qquad \begin{array}{ll} \text{maximize} & -b^T z - f^*(-A^T z) \end{array}$$

linear inequality constraints: g is indicator for $\{y \mid y \preceq b\}$

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax \preceq b \end{array} \qquad\qquad \begin{array}{ll} \text{maximize} & -b^T z - f^*(-A^T z) \\ \text{subject to} & z \succeq 0 \end{array}$$

norm regularization: $g(y) = \|y - b\|$

$$\begin{array}{ll} \text{minimize} & f(x) + \|Ax - b\| \end{array} \qquad\qquad \begin{array}{ll} \text{maximize} & -b^T z - f^*(-A^T z) \\ \text{subject to} & \|z\|_* \leq 1 \end{array}$$

Dual methods

apply first-order method to dual problem

$$\text{maximize} \quad -f^*(-A^T z) - g^*(z)$$

reasons why dual problem may be easier for first-order method:

- dual problem is unconstrained or has simple constraints
- dual objective is differentiable or has a simple nondifferentiable term
- decomposition: exploit separable structure

Outline

- introduction: dual methods
- **gradient and subgradient of conjugate**
- dual decomposition
- network utility maximization
- network flow optimization

(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

- f^* is subdifferentiable on (at least) $\mathbf{int\,dom}\, f^*$ (page 4-6)
- maximizers in the definition of $f^*(y)$ are subgradients at y (page 8-13)

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

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

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

Conjugate of strongly convex function

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

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

- 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 $1/\mu$

$$\|\nabla f^*(y) - \nabla f^*(y')\|_2 \leq \frac{1}{\mu}\|y - y'\|_2 \quad \forall y, y'$$

proof: if f is strongly convex and closed

- $y^T x - f(x)$ has a unique maximizer x for every y
- x maximizes $y^T x - f(x)$ if and only if $y \in \partial f(x)$; from page 8-13

$$y \in \partial f(x) \qquad \Longleftrightarrow \qquad x \in \partial f^*(y) = \{\nabla f^*(y)\}$$

hence $\nabla f^*(y) = \mathrm{argmax}_x \left(y^T x - f(x)\right)$

- from convexity of $f(x) - (\mu/2)x^T x$:

$$(y - y')^T (x - x') \geq \mu \|x - x'\|_2^2 \quad \text{if } y \in \partial f(x),\, y' \in \partial f(x')$$

- this is co-coercivity of ∇f^* (which implies Lipschitz continuity)

$$(y - y')^T (\nabla f^*(y) - \nabla f^*(y')) \geq \mu \|\nabla f^*(y) - \nabla f^*(y')\|_2^2$$

Outline

- introduction: dual methods
- gradient and subgradient of conjugate
- **dual decomposition**
- network utility maximization
- network flow optimization

Equality constraints

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax = b \end{array} \qquad\qquad \begin{array}{ll} \text{minimize} & -f^*(-A^T z) - b^T z \end{array}$$

dual gradient ascent (assuming $\mathbf{dom}\, f^* = \mathbf{R}^n$):

$$\hat{x} = \underset{x}{\operatorname{argmin}} \left(f(x) + z^T A x \right), \qquad z^+ = z + t(A\hat{x} - b)$$

- $\hat{x}$ is a subgradient of f^* at $-A^T z$ (*i.e.*, $\hat{x} \in \partial f^*(-A^T z)$)
- $b - A\hat{x}$ is a subgradient of $b^T z + f^*(-A^T z)$ at z

of interest if calculation of $\hat{x}$ is inexpensive (for example, f is separable)

Dual decomposition

convex problem with separable objective

$$\begin{array}{ll} \text{minimize} & f_1(x_1) + f_2(x_2) \\ \text{subject to} & A_1 x_1 + A_2 x_2 \preceq b \end{array}$$

constraint is *complicating* or *coupling* constraint

dual problem

$$\begin{array}{ll} \text{maximize} & -f_1^*(-A_1^T z) - f_2^*(-A_2^T z) - b^T z \\ \text{subject to} & z \succeq 0 \end{array}$$

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

Dual subgradient projection

subproblem: to calculate $f_j^*(-A_j^T z)$ and a (sub-)gradient for it,

$$\text{minimize (over } x_j\text{)} \quad f_j(x_j) + z^T A_j x_j$$

optimal value is $f_j^*(-A_j^T z)$; minimizer $\hat{x}_j$ is in $\partial f_j^*(-A_j^T z)$

dual subgradient projection method

$$\begin{aligned} \hat{x}_j &= \underset{x_j}{\operatorname{argmin}} \left(f_j(x_j) + z^T A_j x_j \right), \quad j = 1, 2 \\ z^+ &= (z + t(A_1\hat{x}_1 + A_2\hat{x}_2 - b))_+ \end{aligned}$$

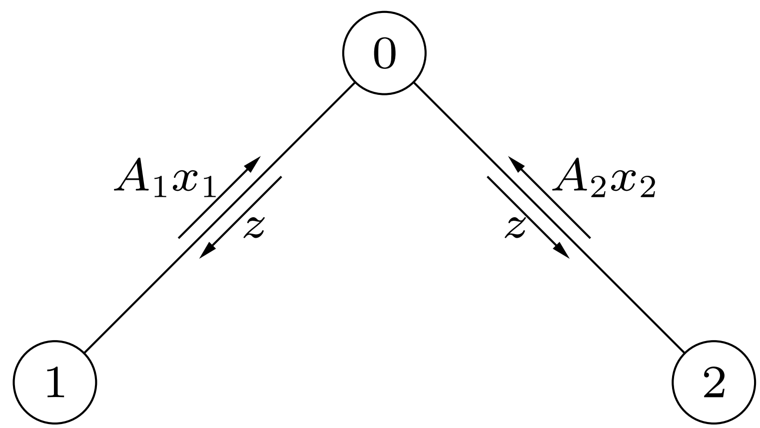
- minimization problems over x_1, x_2 are independent
- z-update is projected subgradient step ($u_+ = \max\{u, 0\}$ elementwise)

Interpretation as price coordination

- $p = 2$ units in a system; unit j chooses decision variable x_j
- constraints are limits on shared resources; z_i is price of resource i
- dual update $z_i^+ = (z_i - ts_i)_+$ depends on slacks $s = b - A_1x_1 - A_2x_2$
  - increases price z_i if resource is over-utilized ($s_i < 0$)
  - decreases price z_i if resource is under-utilized ($s_i > 0$)
  - never lets prices get negative

distributed architecture

- central node sets prices z
- peripheral node j sets x_j



Quadratic programming example

$$
\begin{array}{ll}
\text{minimize} & \sum_{j=1}^{r} (x_j^T P_j x_j + q_j^T x_j) \\
\text{subject to} & B_j x_j \preceq d_j, \quad j = 1, \ldots, r \\
& \sum_{j=1}^{p} A_j x_j \preceq b
\end{array}
$$

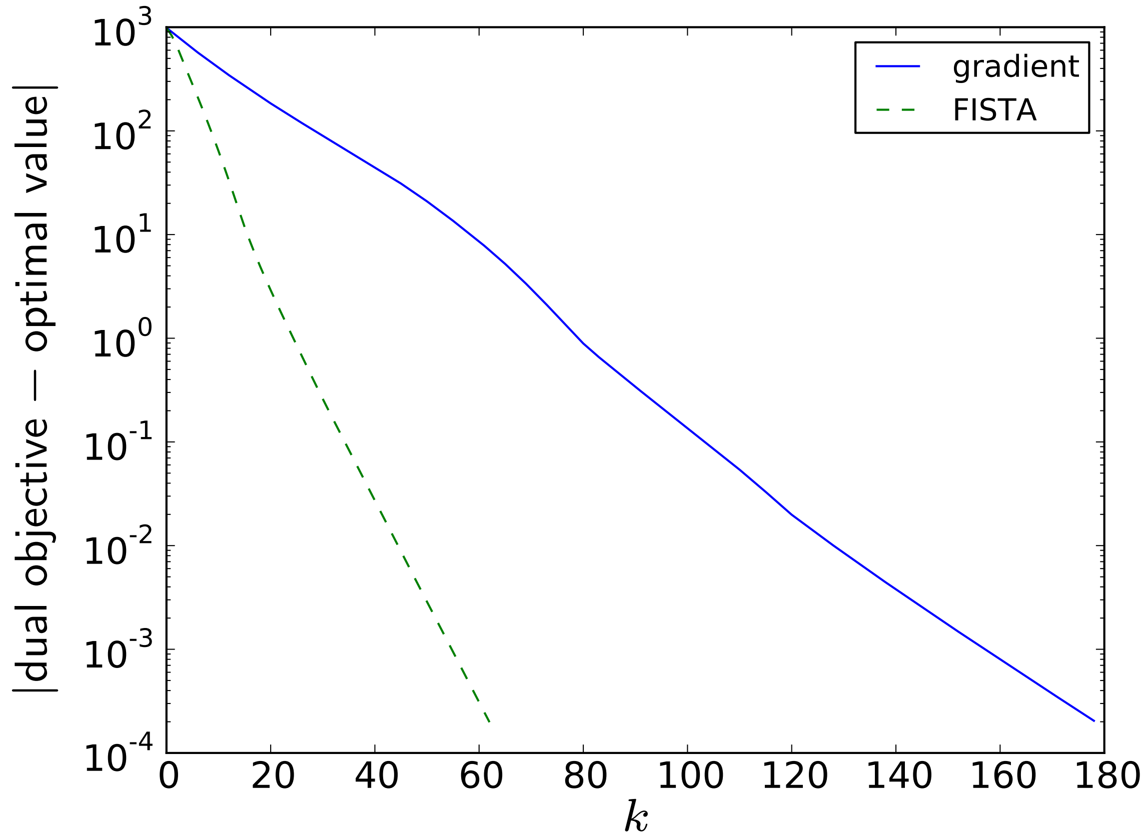
- $r = 10$, variables $x_j \in \mathbf{R}^{100}$, 10 coupling constraints ($A_j \in \mathbf{R}^{10 \times 100}$)
- $P_j \succ 0$; implies dual function has Lipschitz continuous gradient

subproblems: each iteration requires solving 10 decoupled QPs

$$
\begin{array}{ll}
\text{minimize (over } x_j\text{)} & x_j^T P_j x_j + (q_j + A_j^T z)^T x_j \\
\text{subject to} & B_j x_j \preceq d_j
\end{array}
$$

gradient projection and fast gradient projection

- fixed step size (equal in the two methods)
- plot shows dual objective gap



Outline

- introduction: dual methods
- gradient and subgradient of conjugate
- dual decomposition
- **network utility maximization**
- network flow optimization

Network utility maximization

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: $y \preceq c$

Dual network utility maximization problem

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

a convex problem; dual decomposition gives decentralized method

dual problem

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

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

(Sub-)gradients of dual function

dual objective

$$\begin{aligned} f(z) &= c^T z + \sum_{j=1}^{n} (-U_j)^*(-r_j^T z) \\ &= c^T z + \sum_{j=1}^{n} \sup_{x_j} \big(U_j(x_j) - (r_j^T z) x_j\big) \end{aligned}$$

subgradient

$$c - R\hat{x} \in \partial f(z) \quad \text{where} \quad \hat{x}_j = \operatorname*{argmax}_{x_j} \big(U_j(x_j) - (r_j^T z) x_j\big)$$

- if U_j is strictly concave, this is a gradient
- $r_j^T z$ is the sum of link prices along route j
- $c - R\hat{x}$ is vector of link capacity margins for flow $\hat{x}$

Dual decomposition algorithm

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

1. sum link prices along each route: calculate $\lambda_j = r_j^T z$ for $j = 1, \ldots, n$
2. optimize flows (separately) using flow prices

$$\hat{x}_j = \underset{x_j}{\operatorname{argmax}} \left(U_j(x_j) - \lambda_j x_j \right), \quad j = 1, \ldots, n$$

3. calculate link capacity margins $s = c - R\hat{x}$
4. update link prices using projected (sub-)gradient step with step t

$$z := (z - ts)_+$$

decentralized:

- to find λ_j, $\hat{x}_j$ source j only needs to know the prices on its route
- to update s_i, z_i, link i only needs to know the flows that pass through it

Outline

- introduction: dual methods
- gradient and subgradient of conjugate
- dual decomposition
- network utility maximization
- **network flow optimization**

Single commodity network flow

network

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

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

flow vector and external sources

- variable x_j denotes flow (traffic) on arc j
- b_i is external demand (or supply) of flow at node i (satisfies $\mathbf{1}^T b = 0$)
- flow conservation: $Ax = b$

Network flow optimization problem

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

- ϕ is a separable sum of convex functions
- dual decomposition yields decentralized solution method

dual problem (a_j is jth column of A)

$$\text{maximize} \quad -b^T z - \sum_{j=1}^{n} \phi_j^*(-a_j^T z)$$

- dual variable z_i can be interpreted as potential at node i
- $y_j = -a_j^T z$ is the potential difference across arc j
  (potential at start node minus potential at end node)

(Sub-)gradients of dual function

negative dual objective

$$f(z) = b^T z + \sum_{j=1}^{n} \phi_j^*(-a_j^T z)$$

subgradient

$$b - A\hat{x} \in \partial f(z) \quad \text{where} \quad \hat{x}_j = \operatorname{argmin}\left(\phi_j(x_j) + (a_j^T z)x_j\right)$$

- this is a gradient if the functions ϕ_j are strictly convex
- if ϕ_j is differentiable, $\phi_j'(\hat{x}_j) = -a_j^T z$

Dual decomposition network flow algorithm

given initial potential vector z, repeat

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

$$\hat{x}_j = \underset{x_j}{\operatorname{argmin}} \left(\phi_j(x_j) - y_j x_j\right), \quad j = 1, \ldots, n$$

2. compute flow residual at each node: $s := b - A\hat{x}$
3. update node potentials using (sub-)gradient step with step size t

$$z := z - ts$$

decentralized:

- flow is calculated from potential difference across arc
- node potential is updated from its own flow surplus

Electrical network interpretation

network flow optimality conditions (with differentiable ϕ_j)

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

network with node incidence matrix A, nonlinear resistors in branches

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

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

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

z_j is node potential; $y_j = -a_j^T z$ 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 and conjugate ($c_j > 0$ are link capacities):

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

(with $\mathbf{dom}\, \phi_j = [0, c_j)$)

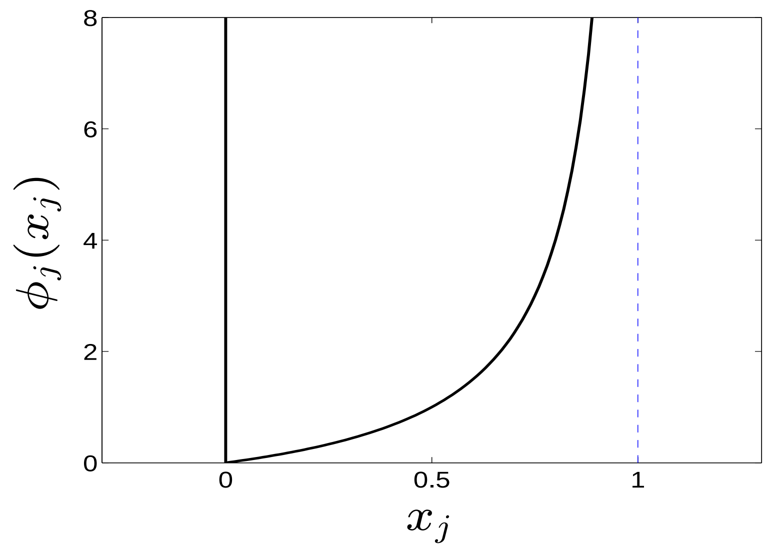
- ϕ_j is differentiable except at $x_j = 0$

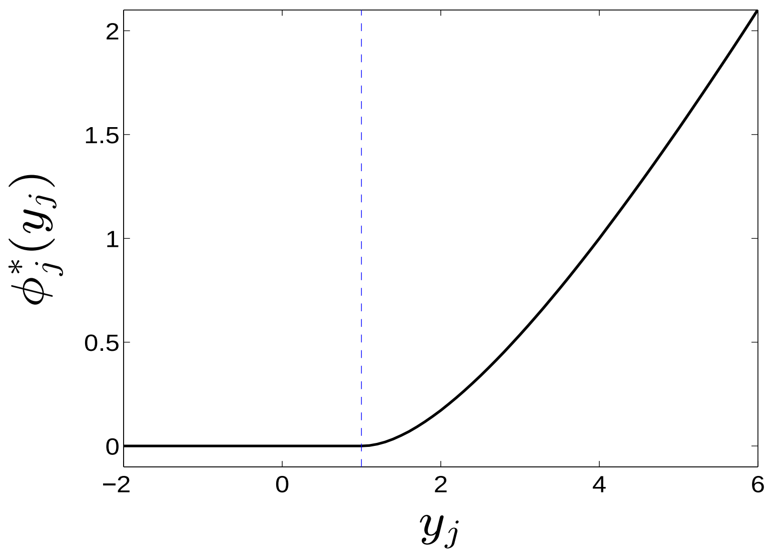
$$\partial\phi_j(0) = (-\infty, 0], \qquad \phi_j'(x_j) = \frac{c_j}{(c_j - x_j)^2} \quad (0 < x_j < c_j)$$

- ϕ_j^* is differentiable

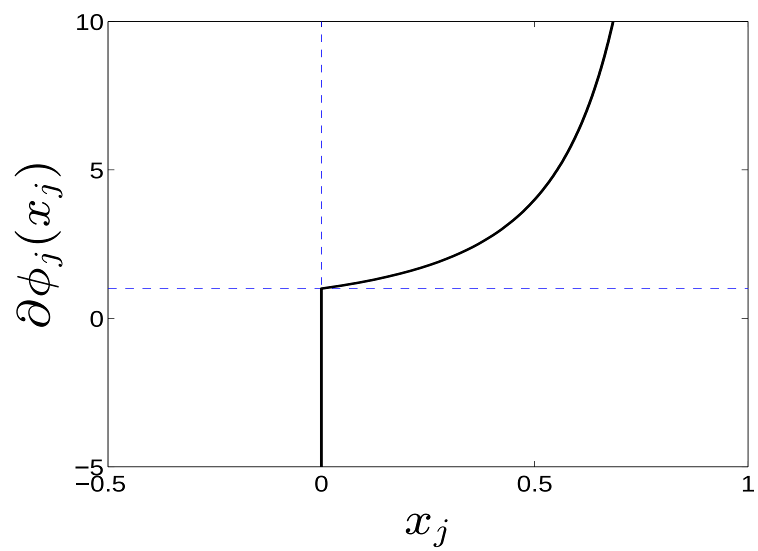
$$\phi_j^{*\prime}(y_j) = \begin{cases} 0 & y_j \leq 1/c_j \\ c_j - \sqrt{c_j/y_j} & y_j > 1/c_j \end{cases}$$

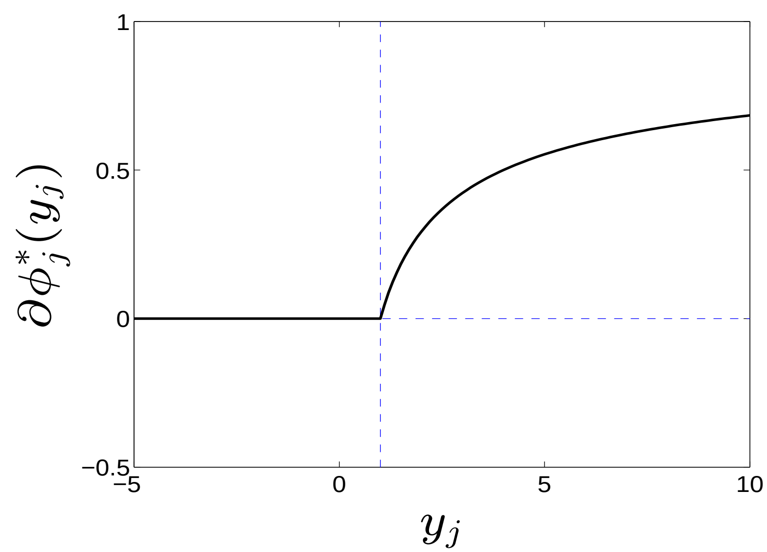
flow cost function and conjugate $(c_j = 1)$





derivatives





References

- S. Boyd, course notes for EE364b. Lectures and 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)