Decomposition techniques for distributed optimization: a tutorial overview

Mikael Johansson
KTH – Stockholm – Sweden

(with a voice gone bust ☹)

Aim of these lectures

"To present some of the key techniques for decomposition and distributed optimization in a coherent and comprehensible manner"

Focus on understanding, not all the details
- each lecture could be a full-semester course
- you will have to work with the material yourself!

Focus on fundamentals
- many techniques date back to 60's-80's, ...
- but some are very recent, and research frontier is not far away

References at end of presentation (will be posted on-line later this week)

Why distributed optimization

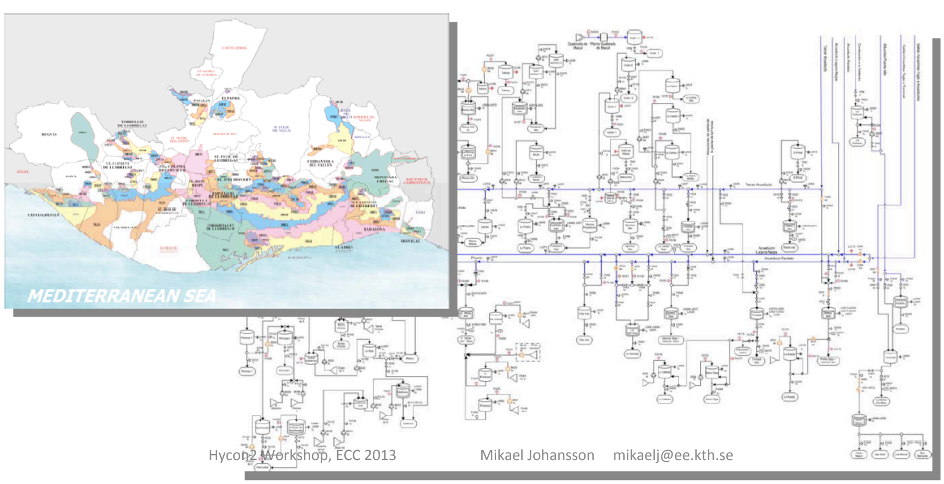
Optimization on a "Google scale"
- information processing on huge data sets

Coordination and control of large-scale systems
- power and water distribution
- vehicle coordination and planning
- sensor, social, and data networks

Theoretical foundation for communication protocol design
- Internet congestion control
- scheduling and power control in wireless systems

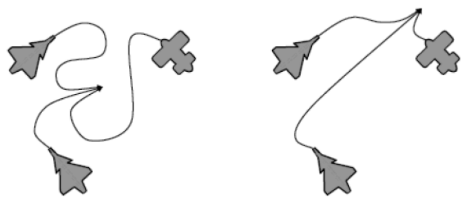
Example: water distribution

Coordinated control of water distribution in city of Barcelona (WIDE)



Example: multi-agent coordination

Cooperate to find jointly optimal controls and rendez-vous point



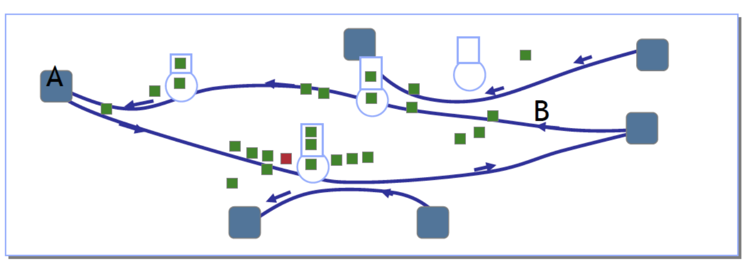
$$\begin{array}{ll} \text{minimize} & \sum_{i \in V} f_i(\theta) \\ \text{subject to} & \theta \in \Theta \end{array}$$

where

$$f_i(\theta) = \begin{array}{ll} \min & \sum_{t=0}^{T} (x_t - \theta)^T Q (x_t - \theta) + u_t^T R u_t \\ \text{s.t.} & x_{t+1} = Ax_t + Bu_t, \quad t = 0, \dots, T-1 \end{array}$$

Example: communication protocol design

Understand how TCP/IP shares network resources between users



$$\begin{array}{ll} \text{maximize} & \sum_i u_i(x_i) \\ \text{subject to} & \sum_{i \in P(l)} x_i \le c_l, \quad l \in L \end{array}$$

Lecture overview

Lecture 1: first-order methods for convex optimization

Lecture 2: decomposition techniques, application to multi-agent optimization

Part I:
Convex optimization using first-order methods

Aim: to understand

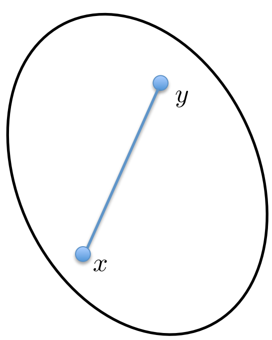
- properties and analysis techniques for basic gradient method
- the interplay between problem structure and convergence rate guarantees
- how we can deal with non-smoothness, noise and constraints

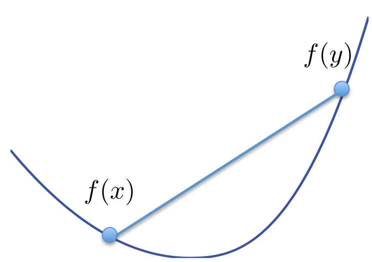
Rationale

Convex optimization:
- minimize convex function subject to convex constraints
- local minima global, strong and useful theory

First-order methods:
- only use function and gradient evaluations (i.e. no Hessians)
- easy to analyze, implement and distribute, yet competitive

Convex functions and convex sets





$\alpha x + (1-\alpha) y \in X,\ \alpha \in [0,1]$

$\alpha f(x) + (1-\alpha) f(y) \geq f(\alpha x + (1-\alpha) y),\ \alpha \in [0,1]$

Affine lower bounds from convexity

Why?

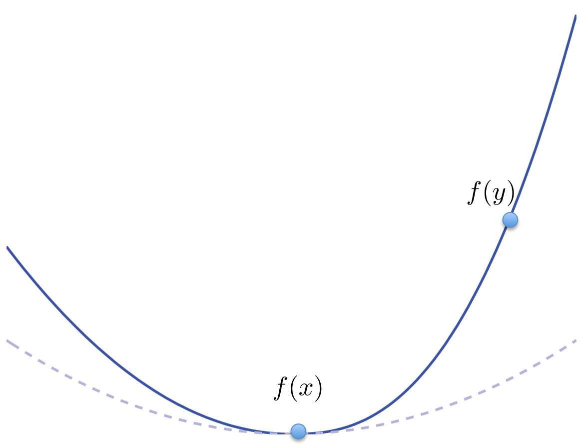
$$\alpha f(x) + (1-\alpha) f(y) \geq f(\alpha x + (1-\alpha) y) \Rightarrow$$

$$\begin{aligned} f(y) &\geq \frac{1}{1-\alpha}\left(f(\alpha x + (1-\alpha)y) - \alpha f(x)\right) = \\ &= f(x) + \frac{1}{1-\alpha}\left(f(\alpha x + (1-\alpha)y) - f(x)\right) \\ &= f(x) + \frac{1}{1-\alpha}\left(f(x + (1-\alpha)(y-x)) - f(x)\right) \end{aligned}$$

Letting $\alpha \to 0$ yields result. Can also go in other direction

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle$$

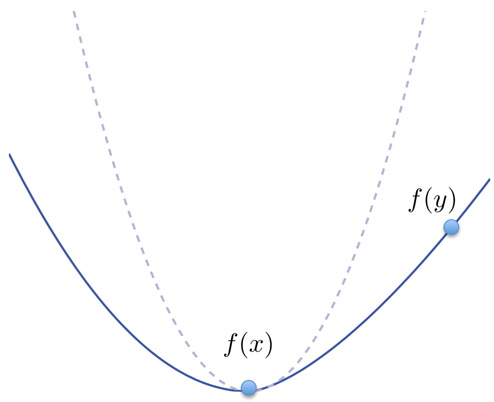
Strong convexity – quadratic lower bounds



$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{c}{2}\|y - x\|^2$$

Lipschitz continuous gradient – upper bounds

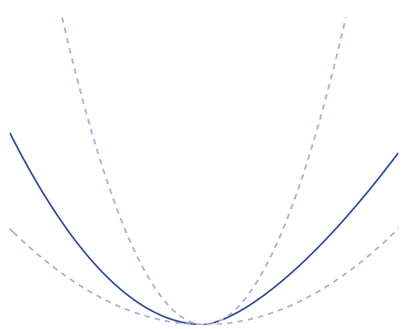
Lipschitz-continuous gradient: $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$



Yields upper quadratic bound: $f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2}\|y - x\|^2$

Strongly convex functions with Lipschitz gradient

Bounded from above and below by quadratic functions



Condition number $\kappa = L/c$ impacts performance of first-order methods.
Note: limited function class when required to hold globally.

The basic gradient method

Basic gradient method

$$x(t+1) = x(t) - \alpha(t)\nabla f(x(t))$$

A **descent** method (for small enough step-size $\alpha(t)$).

Convergence proof.

$$\begin{aligned}\|x(t+1) - x^\star\|_2^2 &= \|x(t) - x^\star\|_2^2 - 2\alpha(t)\langle \nabla f(x(t)), x(t) - x^\star\rangle + \alpha(t)^2\|\nabla f(x(t))\|_2^2 \\ &\leq \|x(t) - x^\star\|_2^2 - 2\alpha(t)\left(f(x(t)) - f^\star\right) + \alpha(t)^2\|\nabla f(x(t))\|_2^2\end{aligned}$$

Where the inequality follows from convexity of f

Gradient method convergence proof

Applying recursively, we find

$$\|x(T) - x^\star\|_2^2 \leq \|x(0) - x^\star\|_2^2 - 2\sum_{t=0}^{T-1}\alpha(t)(f(x(t)) - f^\star) + \sum_{t=0}^{T-1}\alpha^2(t)\|\nabla f(x(t))\|_2^2$$

Since gradient method is descent, and norms are non-negative

$$2(f(x(T)) - f^\star)\sum_{t=0}^{T-1}\alpha(t) \leq \|x(0) - x^\star\|_2^2 + \sum_{t=0}^{T-1}\alpha^2(t)\|\nabla f(x(t))\|_2^2$$

Hence, with $R_0 = \|x(0) - x^\star\|$

$$f(x(T)) - f^\star \leq \frac{R_0^2 + \sum_{t=0}^{T-1}\alpha^2(t)\|\nabla f(x(t))\|_2^2}{2\sum_{t=0}^{T-1}\alpha(t)}$$

Further assumptions needed to guarantee convergence!

Gradient method discussion

If we assume that f is Lipschitz, *i.e.* $\|\nabla f(x(t))\| \leq L_f$

$$f(x(T)) - f^\star \leq \frac{R_0^2 + L_f^2 \sum_{t=0}^{T-1} \alpha^2(t)}{2 \sum_{t=0}^{T-1} \alpha(t)}$$

Then,

- For fixed step-size $\alpha(t) = \alpha$

$$\lim_{T \to \infty} f(x(T)) \leq f^\star + \frac{\alpha L_f^2}{2}$$

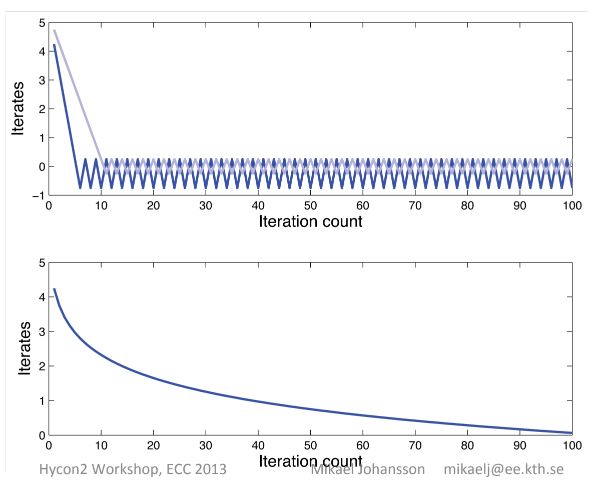
- For diminishing stepsizes $\sum_{t=0}^{\infty} \alpha^2(t) < \infty,\ \sum_{t=0}^{\infty} \alpha(t) = \infty$

$$\lim_{T \to \infty} f(x(T)) = f^\star$$

- Accuracy ε can be obtained in $(R_0 L_f)^2 / \varepsilon^2$ steps

Example

Smaller residual error for smaller stepsize, convergence for diminishing



Strongly convex functions with Lipschitz gradient

As in the basic gradient method proof

$$\|x(t+1)-x^\star\|_2^2 = \|x(t)-x^\star\|_2^2 - 2\alpha(t)\langle \nabla f(x(t)), x(t)-x^\star\rangle + \alpha^2(t)\|\nabla f(x(t))\|_2^2$$

For strongly convex functions with Lipschitz-continuous gradient, it holds

$$\langle \nabla f(x(t)), x(t)-x^\star\rangle \geq \frac{cL}{c+L}\|x(t)-x^\star\|_2^2 + \frac{1}{c+L}\|\nabla f(x(t))\|^2$$

so

$$\|x(t+1)-x^\star\|_2^2 \leq \left(1+\frac{2\alpha(t)cL}{c+L}\right)\|x(t)-x^\star\|_2^2 + \alpha(t)\left(\alpha(t)-\frac{2}{c+L}\right)\|\nabla f(x(t))\|_2^2$$

Hence, if $\alpha(t) \leq 2/(c+L)$ we obtain **linear convergence** rate

$$\|x(t+1)-x^\star\|_2^2 \leq \left(1-\frac{2cL}{c+L}\alpha(t)\right)\|x(t)-x^\star\|_2^2$$

Order-optimal methods

The basic gradient method is **not** the optimal first-order method.

- optimal first-order methods typically use memory, e.g.

$$x(t+1) = y(t) - L^{-1}\nabla f(y(t))$$
$$y(t+1) = x(t+1) + \frac{1-\sqrt{\kappa}}{1+\sqrt{\kappa}}(x(t+1)-x(t))$$

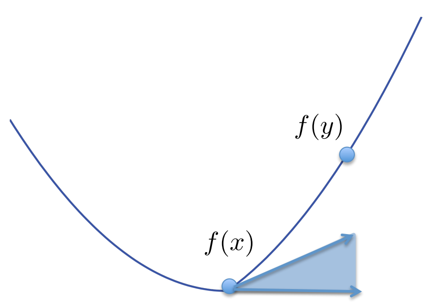
Particularly useful when f is convex and has Lipschitz-continuous gradient

- from $\mathcal{O}(1/\varepsilon)$ to $\mathcal{O}(1/\sqrt{\varepsilon})$
- achieves optimal rate (same as basic gradient) also in other cases
- not always fastest first-order method

Gradient methods: limits of performance

Problem class	First-order method	Complexity	e=1%
Lipschitz-continuous function	Gradient	$\mathcal{O}(1/\varepsilon^2)$	10,000
Lipschitz-continuous gradient	Gradient	$\mathcal{O}(1/\varepsilon)$	100
	Optimal gradient	$\mathcal{O}(1/\sqrt{\varepsilon})$	10
Strongly convex, Lipschitz gradient	Gradient	$\ln(1/\varepsilon)$	2.3
	Optimal gradient	$\ln(1/\varepsilon)$	

Non-smooth convex functions: subgradients



Subgradient s_x gives affine lower bound on convex function at x

$$f(y) \geq f(x) + \langle s_x, x - y \rangle$$

Subdifferential: set of all subgradients

The subgradient method

As the gradient method, but using subgradients instead

$$x(t+1) = x(t) - \alpha(t)s(t), \quad s(t) \in \partial f(x(t))$$

Not a descent method.

Hence, cannot bound $\sum_{t=0}^{T} \alpha(t)(f(x(t)) - f^\star)$ as before . Rather, we find

$$\inf_t f(x(t)) \leq f^\star + \frac{R_0^2 + \sum_{t=0}^{T} \alpha^2(t)\|s(t)\|_2^2}{2\sum_{t=0}^{T} \alpha(t)}$$

If subgradients are bounded, then same conclusions as for gradient method. (step-size, convergence rates, ...)

Averages behave better...

The running averages of iterates

$$\overline{x}(t) = \frac{1}{t}\sum_{k=0}^{t} x(k)$$

are often better-behaved than iterates themselves.

Specifically, if subgradients are bounded $\|s_x\| \leq L$, then averages satisfy

$$f(\overline{x}(T)) \leq f^\star + \frac{\sqrt{2}R_0L}{\sqrt{T}}$$

(note how "inf" is gone)

Hycon2 Workshop, ECC 2013 Mikael Johansson mikaelj@ee.kth.se

Gradient method for constrained optimization

Constrained minimization problem

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in X \end{array}$$

If projections onto X are easy to compute, can use **projected gradient**

$$x(t+1) = P_X\{x(t) - \alpha(t)\nabla f(x(t))\}$$

Same convergence proof as before, since projections are non-expansive

$$\|P_X\{x\} - P_X\{y\}\|^2 \le \|x-y\|^2$$

Beyond the basic methods

Smooth optimization of non-smooth functions

- epsilon-optimal solution to non-smooth problem requires many iterations
- often better to smooth function and apply order-optimal method

Exploiting structure

- when problem is smooth problem + easily-solvable non-smooth
- many current applications in compressed sensing, sparse optimization

…

Duality

Associated with every convex optimization problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0 \\ & Ax = b \end{array}$$

is an associated dual problem

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

where

$$g(\lambda, \mu) = \inf_x \left\{ f_0(x) + \sum_i \lambda_i f_i(x) + \mu^T (Ax - b) \right\}$$

Advantage: dual problem convex, has simple constraint set

Key properties of dual function

Dual function g is always concave, may be non-smooth.

Dual function is a lower bound of optimal value when $\lambda \succeq 0$

$$g(\lambda, \mu) = \inf_x f_0(x) + \sum_i \lambda_i f_i(x) + \mu^T (Ax - b) \leq f_0(x^\star)$$

For convex problems, primal optimal value agrees with dual optimal value

$$g^\star = \sup_{\lambda \succeq 0, \mu} g(\lambda, \mu) = g(\lambda^\star, \mu^\star) = f_0(x^\star)$$

e.g. when there is a feasible point satisfying inequality constraints strictly ("Slater condition")

Solving the dual problem

Dual function concave, but possibly non-smooth.

Dual problem often solved by projected (sub)-gradient method

$$\lambda(t+1) = \mathcal{P}_+ \{\lambda(t) + \alpha(t)s_\lambda(t)\} \qquad s_\lambda \in \partial_\lambda g(\lambda(t), \mu(t))$$
$$\mu(t+1) = \mu(t+1) + \alpha(t)s_\mu(t) \qquad s_\mu \in \partial_\mu g(\lambda(t), \mu(t))$$

Can do better when dual function is strongly concave, has Lipschitz gradient!
(conditions for this will follow in next lecture...)

Summary of Lecture 1

First-order methods for convex optimization:
- gradient method: convergence proof and convergence rate estimates
- optimal methods: more states, but still only gradient information
- easy to implement, strong performance for certain problem classes

Non-smooth optimization
- subgradient method
- not a descent method, averaging gives better properties

Duality and the dual optimization problem

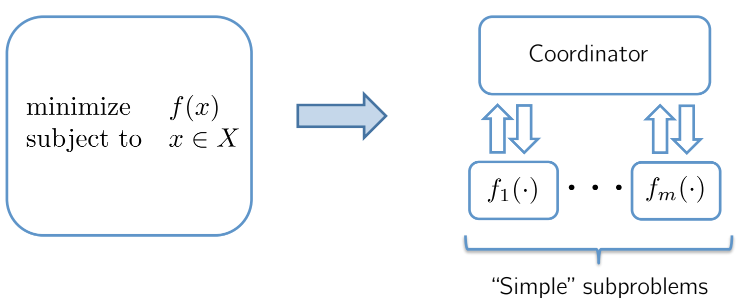
Part II:
Decomposition techniques

Aim: to understand

- The basic idea of decomposition, coupling variables/constraints
- Dual decomposition: principle, advantages and challenges
- Application to multi-agent optimization

Basic idea of decomposition techniques

Decompose one complex problem into many small:



The trivial case

Separable objectives and constraints

$$\begin{array}{ll} \text{minimize} & \sum_i f_i(x_i) \\ \text{subject to} & x_i \in X_i \end{array}$$

Trivially separates into n decoupled subproblems

$$\begin{array}{ll} \text{minimize} & f_i(x_i) \\ \text{subject to} & x_i \in X_i \end{array}$$

that can be solved in parallel and combined.

The more interesting ones

Problems with **coupling constraints**

$$\begin{array}{ll} \text{minimize} & f_1(x_1) + f_2(x_2) \\ \text{subject to} & x_1 + x_2 \leq c \end{array}$$

Problems with **coupled objectives**

$$\text{minimize} \quad f_1(x_1, x_{12}) + f_2(x_{12}, x_2)$$

Coupled objectives can be cast as a problem of coupling constraints:

$$\begin{array}{ll} \text{minimize} & f_1(x_1, z_{12}) + f_2(z_{21}, x_2) \\ \text{subject to} & z_{12} = z_{21} \end{array}$$

so this case will be our focus.

Dual decomposition

Basic idea: decouple problem by relaxing coupling constraints.

$$\begin{array}{ll} \text{minimize} & f_1(x_1) + f_2(x_2) \\ \text{subject to} & x_1 + x_2 \leq c \end{array}$$

Formally, introduce Lagrange multiplier for the constraint, form Lagrangian

$$L(x, \lambda) = f_1(x_1) + f_2(x_2) + \lambda(x_1 + x_2 - c)$$

with associated dual function

$$g(\lambda) = \inf_x L(x, \lambda) = -\lambda c + \inf_{x_1}\{f_1(x_1) + \lambda x_1\} + \inf_{x_2}\{f_2(x_2) + \lambda x_2\}$$

and solve the dual problem.

Dual decomposition cont'd

Dual problem has the form

$$\begin{array}{ll} \text{maximize} & g_1(\lambda) + g_2(\lambda) \\ \text{subject to} & \lambda \geq 0 \end{array}$$

additive (hence, can be evaluated in parallel) and simple constraints.

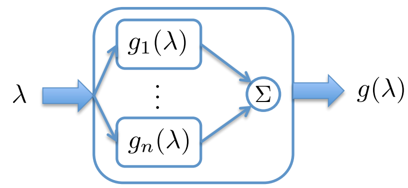
The dual function is always concave, and a subgradient of g is given by

$$x_1^\star(\lambda) + x_2^\star(\lambda) - c$$

Hence, dual problem is convex. Can solve using projected subgradient method.

Dual and distributed optimization

Dual decomposition often results in additive dual function



but might still need coordinator to solve dual optimization problem.

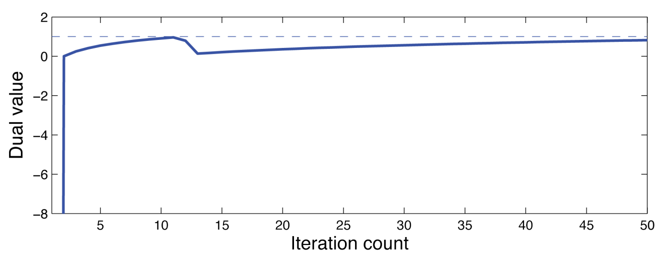
Dual problem fully distributed if (sub)gradient of dual locally available

Dual decomposition example

Simple example:

$$\begin{array}{ll} \text{minimize} & |x_1 - 1| + |x_2 - 1| \\ \text{subject to} & x_1 + x_2 \leq 1 \\ & x_1 \in [-10, 10] \end{array}$$

Optimal value $f_0^\star = 1$ for $x_1^\star = 1 - x_2^\star, \quad x_2^\star \in [0, 1]$



Drawback of dual decomposition

Optimizes dual variables, to find optimal value of dual function.

$$\begin{array}{ll} \text{maximize} & g(\lambda) \\ \text{subject to} & \lambda \succeq 0 \end{array} \quad \Rightarrow \lambda^\star,\ d^\star = g(\lambda^\star)$$

In general, primal iterates might be suboptimal, violate constraints.

$$x^\star(\lambda) = \arg\inf_x L(x, \lambda)$$

Under strong convexity of primal, and the existence of a Slater point:

- feasibility and primal optimality recovered in the limit.

➔Constraints and demands on subsystem consistency should be "soft"

Primal convergence in dual methods

Several techniques for enforcing primal convergence, *e.g.* averaging iterates

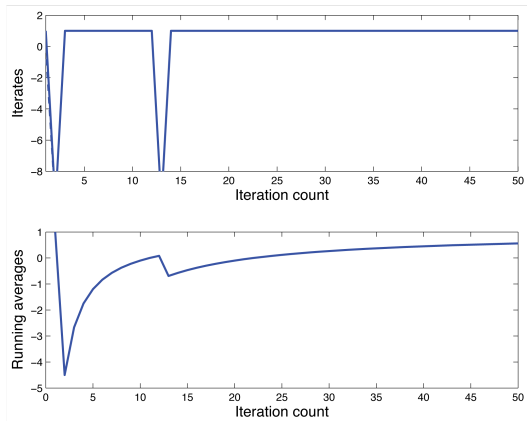
$$\overline{x}^\star(t) = \frac{1}{t}\sum_{k=0}^{t} x^\star(\lambda(t))$$

Under Slater, iterate average satisfies constraints asymptotically and

$$f_0(\overline{x}^\star(t)) \leq f^\star + \frac{\alpha\Delta^2}{2} + \frac{\|\lambda(0)\|_2^2}{2\alpha t}$$

Example

Simple example from before. Iterates and running averages:



Primal convergence in dual methods

Stronger properties when dual function is differentiable, strongly concave.

Fact. Consider the linearly constrained convex optimization problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & Ax = b \\ & Cx \preceq d \end{array}$$

If objective is strongly convex w. modulus c, has Lipschitz-continuous gradient, and there exists a Slater point. Then, if

$$\mathbf{A} = [A^T \; C^T]^T$$

has full row rank, iterates $u = (\lambda, \mu)$ produced by dual projected gradient satisfy

$$\|u(t) - u^\star\| \leq q^t \|u(0) - u^\star\|$$

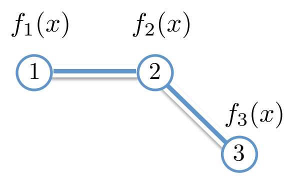
$$\|x^\star(u(t)) - x^\star\| \leq q^t \frac{\sigma_{\max}(A)}{c} \|u(0) - u^\star\|$$

Application to multi-agent optimization

A network of agents collaborate to solve the optimization problem

$$\text{minimize} \quad \sum_{i \in \mathcal{V}} f_i(x)$$

Agents can only exchange information with neighbors in graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$



Three techniques in some detail:

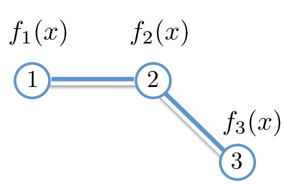
- dual decomposition, consensus-gradient, alternating direction of multipliers

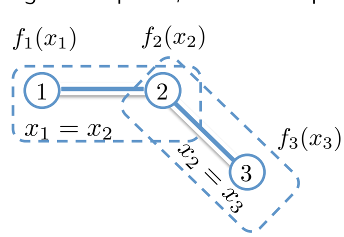
Method 1: dual decomposition

Introduce local copy x_i of decision variable, re-write problem on the form

$$\begin{array}{ll} \text{minimize} & \sum_{i \in \mathcal{V}} f_i(x_i) \\ \text{subject to} & x_i = x_j \qquad \forall (i,j) \in \mathcal{E} \end{array}$$

Relax consistency constraints using Lagrange multipliers, solve dual problem.





The dual decomposition approach

Convenient to write problem as

$$\begin{array}{ll} \text{minimize} & \sum_{i\in\mathcal{V}} f_i(x_i) \\ \text{subject to} & M\mathbf{x} = 0 \end{array}$$

where M is the edge-node incidence matrix of $\mathcal{G}$,

$$[M]_{e,i} = \begin{cases} 1 & \text{if } i \text{ is the start node of edge } e \\ -1 & \text{if } i \text{ is the end node of edge } e \\ 0 & \text{otherwise} \end{cases}$$

The dual decomposition approach

Introducing Lagrange multiplier vector $\mu \in \mathbb{R}^{|\mathcal{E}|}$, form Lagrangian

$$L(x,\mu) = \sum_{i\in\mathcal{V}} f_i(x_i) + \mu^T M x = \sum_{i\in\mathcal{V}} f_i(x_i) + \sum_{j:(i,j)\in\mathcal{E}} \mu_{ij}(x_i - x_j)$$

Dual decomposition updates become

$$x_i(t+1) = \underset{x_i}{\operatorname{argmin}}\, L(x,\mu) = \underset{x_i}{\operatorname{argmin}} \left\{ f_i(x_i) + \sum_{j:(i,j)\in\mathcal{E}} \mu_{ij}(t)x_i - \sum_{j:(j,i)\in\mathcal{E}} \mu_{ji}(t)x_i \right\}$$

$$\mu_{ij}(t+1) = \mu_{ij}(t) + \alpha(t)\left(x_i(t+1) - x_j(t+1)\right)$$

Data exchange only between neighbors.

Does iterations converge? Under what assumptions? Good stepsizes?

Method 2: consensus-gradients

Use same modeling idea, i.e. consider

$$\begin{array}{ll} \text{minimize} & \sum_{i \in \mathcal{V}} f_i(x_i) \\ \text{subject to} & M\mathbf{x} = 0 \end{array}$$

Replace strict equalities with penalty term

$$\text{minimize} \quad p(\mathbf{x}) := \sum_{i \in \mathcal{V}} f_i(x_i) + \frac{\eta}{2}\|Mx\|_2^2$$

Note: an optimality-consistency trade-off

Gradient descent on penalty function

The gradient iterations become

$$x(t+1) = x(t) - \alpha(t)\frac{\partial}{\partial x_i}p(\mathbf{x}) = x(t) - \alpha(t)(\nabla f(x(t)) + \eta M^T Mx)$$

which we can re-write as

$$x_i(t+1) = \underbrace{x_i(t) + \sum_{j:(i,j)\in\mathcal{E}} \alpha(t)\eta(x_j(t) - x_i(t))}_{\text{“consensus”}} - \alpha(t)\nabla f_i(x_i(t))$$

A combination of fixed-weight consensus and gradient descent.

Consensus-subgradient method

Originally proposed for non-smooth optimization

$$x_i(t+1) = \left\{ W_{ii}x_i(t) + \sum_{j:(i,j)\in\mathcal{E}} W_{ij}x_j(t) \right\} - \alpha_i s(t), \qquad s(t) \in \partial f(x(t))$$

Studied under general consensus weights, time-varying graphs.

For fixed step-sizes, iterations do not converge to true optimum

- need average iterates, use diminishing stepsizes

Method 3: ADMM

Alternating direction of multipliers (ADMM) considers problem on the form

$$\begin{array}{ll} \text{minimize} & f(x)+g(z) \\ \text{subject to} & Ex+Fz=h \end{array} \Leftrightarrow \begin{array}{ll} \text{minimize} & f(x)+g(z)+\frac{\rho}{2}\|Ex+Fz-h\|_2^2 \\ \text{subject to} & Ex+Fz=h \end{array}$$

Finds optimal solution by alternating minimization of **augmented Lagrangian**

$$L_\rho(x,z,\mu) = f(x)+g(z)+\mu^T(Ex+Fz-h)+\frac{\rho}{2}\|Ex+Fz-h\|_2^2$$

followed by Lagrange multiplier update, i.e.:

$$x(t+1) = \underset{x}{\operatorname{argmin}}\, L_\rho(x, z(t), \mu(t))$$

$$z(t+1) = \underset{z}{\operatorname{argmin}}\, L_\rho(x(t+1), z, \mu(t))$$

$$\mu(t+1) = \mu(t) + \rho(Ex(t+1) + Fz(t+1) - h)$$

ADMM properties

Under mild conditions, ADMM converges for all values of $\rho > 0$
(in contrast to dual methods, where large step-size can cause divergence)

Convergence rates of ADMM is a topic of intense current research.

The penalty parameter ρ affects the convergence factors of the iterates.

- optimal parameter selection rules exist for some problem classes

ADMM for quadratic problems

Quadratic programming problem

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}x^T P x + q^T x \\ \text{subject to} & Ax \leq b \end{array}$$

re-written on ADMM standard form

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}x^T P x + q^T x + \mathcal{I}_+(z) \\ \text{subject to} & Ax - b + z = 0 \end{array}$$

yields iterations

$$\begin{aligned} x(t+1) &= -(Q + \rho A^T A)^{-1}[q + \rho A^T(z(t) + u(t) - b)] \\ z(t+1) &= \max\{0, -A(t+1) - u(t) + b\} \\ u(t+1) &= u(t) + Ax(t+1) - b + z(t+1) \end{aligned}$$

What can we say about convergence, optimal ρ?

ADMM for quadratic problems

Fact. For all $\rho > 0$ ADMM iterations converge to optimum at linear rate.

Fact. If A is invertible or has full row rank, then

$$\rho = \frac{1}{\sqrt{\lambda_1(AQ^{-1}A^T)\lambda_n(AQ^{-1}A^T)}}$$

yields the smallest convergence factor (fastest convergence times).

(tends to work well also when A does not have these properties)

ADMM for multi-agent optimization

Introduce "agreement variable" $z_{(i,j)}$ on each edge $(i,j) \in \mathcal{E}$, consider

$$\begin{array}{ll} \text{minimize} & \sum_{i\in V} f_i(x_i) \\ \text{subject to} & x_i = z_{(i,j)} \quad \forall (i,j) \in \mathcal{E} \\ & x_j = z_{(i,j)} \quad \forall (i,j) \in \mathcal{E} \end{array}$$

Can be re-written as

$$\begin{array}{ll} \text{minimize} & \sum_{i\in \mathcal{V}} f_i(x_i) \\ \text{subject to} & \underbrace{\begin{bmatrix} M_+ \\ M_- \end{bmatrix}}_{E} x - \underbrace{\begin{bmatrix} I \\ I \end{bmatrix}}_{F} z = 0 \end{array}$$

where $M_+ = \max\{M, 0\},\ M_- = -\min\{M, 0\}$

ADMM for multi-agent optimization

ADMM iterations become

$$
\begin{aligned}
x_i(t+1) &= \operatorname*{argmin}_x f_i(x) + (\mu_{ij} + \mu_{ji})x + \frac{\rho}{2}\left((x - z_{ij})^2 + (x - z_{ji})^2\right) \\
z_{ij}(t+1) &= \rho x_i(t+1) + \mu_{ij}(t) \\
\mu_{ij}(t+1) &= \mu_{ij}(t) + \rho(x_i(t+1) - z_{ij}(t+1))
\end{aligned}
$$

Converge for all values of penalty parameter.

Many variations, extensions (e.g. different penalty parameters per edge)

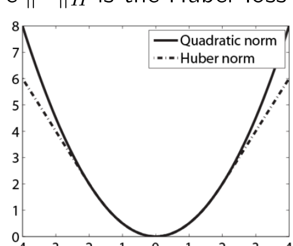
Example: robust estimation

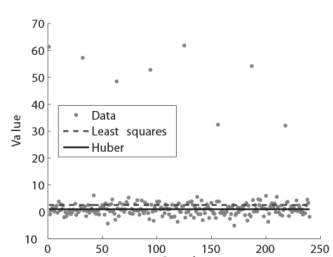
Nodes measure different noisy versions $y_i(t)$ of the same quantity.

Would like to agree on common estimate $\hat{x}$ that minimizes

$$
\begin{array}{ll}
\text{minimize} & \sum_{i\in\mathcal{V}} \|y_i - x\|_H \\
\text{subject to} & x \in X \\
& \mathcal{G} = (\mathcal{V}, \mathcal{E})
\end{array}
$$

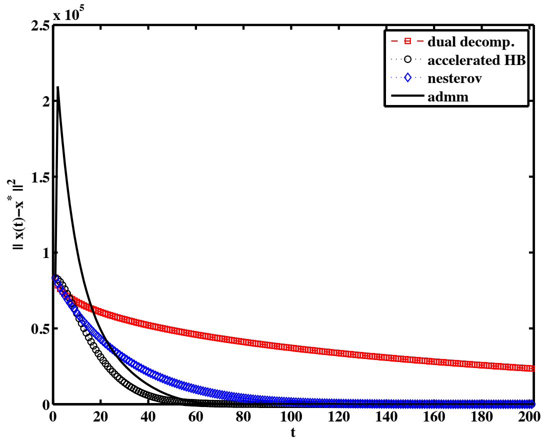
where $\|\cdot\|_H$ is the Huber loss





Example: robust optimization

Representative results, 100-node ring network



Summary of Lecture 2

Dual decomposition: idea and properties.

Multi-agent optimization:

- collaborative optimization under information exchange constraints

Three techniques in (some) detail

- Dual decomposition
- ADMM
- Gradient/consensus method

Many alternative techniques not covered.

So what did we see?

Lecture 1: first-order methods for convex optimization

Lecture 2: dual decomposition and optimization over graphs

References for Lecture 1

Lecture one is covered, at least in parts, in many textbooks. The books

B. Polyak, “Introduction to optimization”, 1987
Y. Nesterov, “Introductiory lectures on convex optimization: a basic course”, 2004

are particularly beautiful accounts. A good reference for duality theory is

D. Bertsekas, A. Ozdaglar, A. Nedich, “Convex analysis and optimization”

References for Lecture 2

The material on dual decomposition is based on the chapter

B. Yang and M. Johansson, "Distributed optimization, a tutorial overview"

from the "Networked Control" book of an earlier Hycon Summer School. The book covers many individual references to original work by a wide range of authors.

The lecture notes

S. Boyd, L. Xiao, A. Mutapcic and J. Mattingley "Notes on decomposition methods", Stanford University, 2007

has a nice introduction to modelling for distributed optimization.

References for lecture 2

The survey paper

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers", 2010

covers theory and applications of ADDM. Optimal penalty parameter selection is studied in

E. Ghadimi, A. Teixeira, I. Shames and M. Johansson, "Optimal parameter selection for the alternating direction of multipliers method (ADMM): quadratic problems", arXiv preprint.

Subgradient-consensus techniques were proposed in

A. Nedich and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization", IEEE Transactions on Automatic Control, 2009.