A Splitting Method for Embedded Optimal Control

B. O'Donoghue   G. Stathopoulos   **S. Boyd**

Stanford University

EMBOPT, 8/9/14, IMT Lucca

Outline

Convex optimal control problem

Operator splitting method

Examples

Conclusion

Convex optimal control problem

- we consider discrete-time, deterministic, finite-horizon control
- linear-convex optimal control problem:

$$\begin{array}{ll} \text{minimize} & \sum_{t=0}^{T} \ell_t(x_t, u_t) \\ \text{subject to} & x_{t+1} = A_t x_t + B_t u_t + c_t, \quad t = 0, \ldots, T-1 \\ & x_0 = x_{\text{init}} \end{array}$$

- variables: states $x_t \in \mathbf{R}^n$ and actions $u_t \in \mathbf{R}^m$, $t = 0, \ldots, T$
- stage cost $\ell_t : \mathbf{R}^n \times \mathbf{R}^m \to \mathbf{R} \cup \{\infty\}$ convex
- infinite values of ℓ_t encode state/action constraints

Solution methods

- many methods to solve convex optimal control problem
  - interior-point methods
  - accelerated (primal or dual) proximal gradient
  - explicit MPC
  - active set
- each has advantages, disadvantages, limitations

This talk

yet another method for convex control problem, that

- is fast and reliable
- is implementable in light, library free code
- can take advantage of parallelism
- scales to large problems
- can be implemented in fixed point arithmetic (in many cases)

Stage cost decomposition

- stage cost decomposed as
$$\ell_t = \phi_t + \psi_t$$
- ϕ_t convex quadratic
- ψ_t non-quadratic, possibly infinite (but convex)
- (decomposition not unique)

Quadratic stage cost

convex quadratic terms $\phi_t : \mathbf{R}^n \times \mathbf{R}^m \to \mathbf{R}$ have the form

$$\phi_t(x, u) = (1/2) \begin{bmatrix} x \\ u \\ 1 \end{bmatrix}^T \begin{bmatrix} Q_t & S_t & q_t \\ S_t^T & R_t & r_t \\ q_t^T & r_t^T & 0 \end{bmatrix} \begin{bmatrix} x \\ u \\ 1 \end{bmatrix}$$

where

$$\begin{bmatrix} Q_t & S_t \\ S_t^T & R_t \end{bmatrix} \succeq 0$$

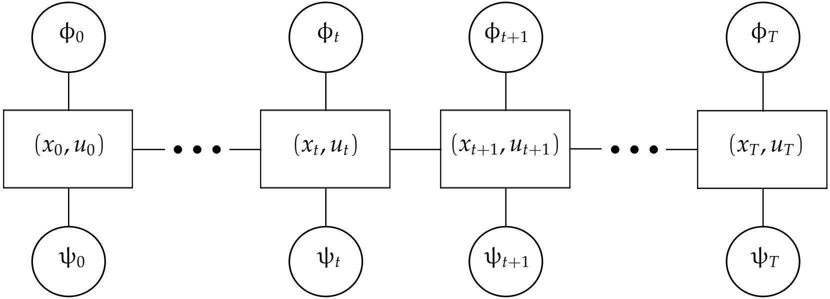
(*i.e.*, symmetric positive semidefinite)

Decomposed problem

$$\begin{array}{ll} \text{minimize} & \sum_{t=0}^{T} \left(\phi_t(x_t, u_t) + \psi_t(x_t, u_t)\right) \\ \text{subject to} & x_{t+1} = A_t x_t + B_t u_t + c_t, \quad t = 0, \ldots, T-1 \\ & x_0 = x_{\text{init}} \end{array}$$

Variable-term graph structure

circles: objective function terms; rectangles: variables



Notation

- $x = (x_0, \ldots, x_T)$, $u = (u_0, \ldots, u_T)$, (x, u) denote whole trajectories
- define trajectory costs

$$\phi(x,u) = \sum_{t=0}^{T} \phi_t(x_t, u_t), \qquad \psi(x,u) = \sum_{t=0}^{T} \psi_t(x_t, u_t)$$

- $\mathcal{D}$ is set of trajectories that satisfy dynamics

$$\mathcal{D} = \{(x,u) \mid x_0 = x_{\text{init}},\ x_{t+1} = A_t x_t + B u_t + c_t,\ t = 0, \ldots, T-1\}$$

- $I_{\mathcal{D}}$ is indicator function of $\mathcal{D}$

$$I_{\mathcal{D}}(x,u) = \begin{cases} 0 & (x,u) \in \mathcal{D} \\ \infty & \text{otherwise} \end{cases}$$

Optimal control problem

$$\text{minimize} \quad I_{\mathcal{D}}(x, u) + \phi(x, u) + \psi(x, u)$$

- $I_{\mathcal{D}}(x, u)$ encodes linear equality (dynamics) constraints
- $\phi(x, u)$ is separable convex quadratic
- $\psi(x, u)$ is separable non-quadratic convex

Outline

Convex optimal control problem

Operator splitting method

Examples

Conclusion

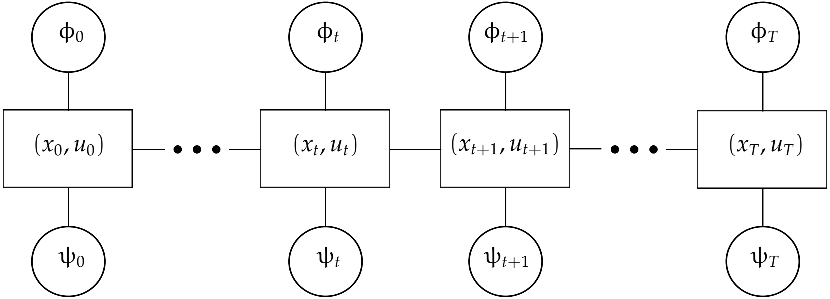
Consensus form

- replicate x and u, and add *consensus* constraints:

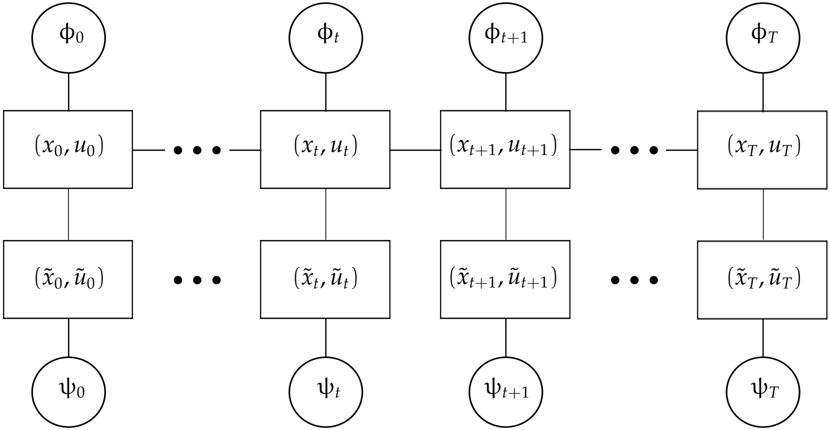
$$\begin{array}{ll} \text{minimize} & (I_{\mathcal{D}}(x,u) + \phi(x,u)) + \psi(\tilde{x},\tilde{u}) \\ \text{subject to} & (x,u) = (\tilde{x},\tilde{u}) \end{array}$$

over $(x,u) \in \mathbf{R}^{(n+m)(T+1)}$ and $(\tilde{x},\tilde{u}) \in \mathbf{R}^{(n+m)(T+1)}$

Graph structure (original problem)



Graph structure (consensus form)



Proximal operator

- define prox operator

$$\mathbf{prox}_f(v) = \underset{x}{\mathrm{argmin}} \left(f(x) + (\rho/2)\|x - v\|_2^2 \right)$$

  with parameter $\rho > 0$

- generalizes notion of projection
- prox operators of many functions have simple forms

Douglas-Rachford splitting for consensus convex optimization

- consensus convex optimization problem

$$\begin{array}{ll} \text{minimize} & f(x) + g(z) \\ \text{subject to} & x = z \end{array}$$

- DR splitting algorithm: starting from any z^0, λ^0, for $k = 0, 1, \ldots,$

$$\begin{aligned} x^{k+1} &:= \mathbf{prox}_f(z^k + \lambda^k) \\ z^{k+1} &:= \mathbf{prox}_g(x^{k+1} - \lambda^k) \\ \lambda^{k+1} &:= \lambda^k + x^{k+1} - z^{k+1} \end{aligned}$$

- λ is (scaled) dual variable associated with consensus constraint
- λ^k is running summing of errors $x^k - z^k$ (integral control)
- converges to solution, if one exists

Operator splitting for control (OSC)

- consensus form optimal control problem:

$$\begin{array}{ll} \text{minimize} & (I_{\mathcal{D}}(x,u) + \phi(x,u)) + \psi(\tilde{x},\tilde{u}) \\ \text{subject to} & (x,u) = (\tilde{x},\tilde{u}) \end{array}$$

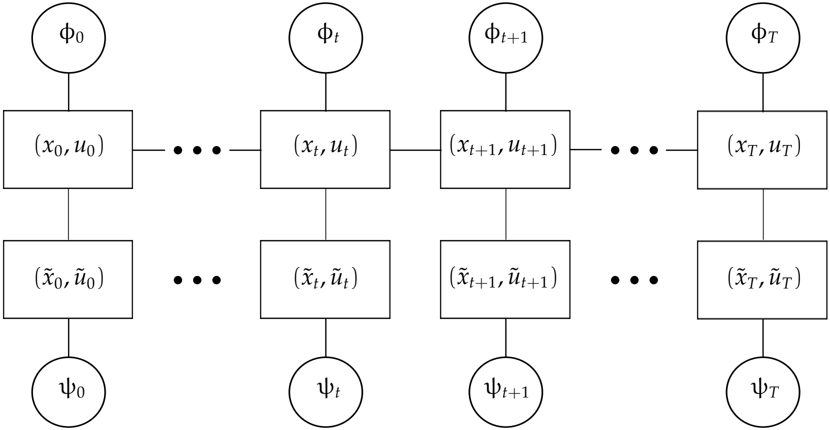
- OSC: starting from any $(\tilde{x}^0, \tilde{u}^0)$, (z^0, y^0), for $k = 0, 1, \ldots,$

$$\begin{aligned} (x^{k+1}, u^{k+1}) &:= \mathbf{prox}_{I_{\mathcal{D}}+\phi}(\tilde{x}^k + z^k, \tilde{u}^k + y^k) \\ (\tilde{x}^{k+1}, \tilde{u}^{k+1}) &:= \mathbf{prox}_{\psi}(x^{k+1} - z^k, u^{k+1} - y^k) \\ (z^{k+1}, y^{k+1}) &:= (z^k, y^k) + (\tilde{x}^{k+1} - x^{k+1}, \tilde{u}^{k+1} - u^{k+1}) \end{aligned}$$

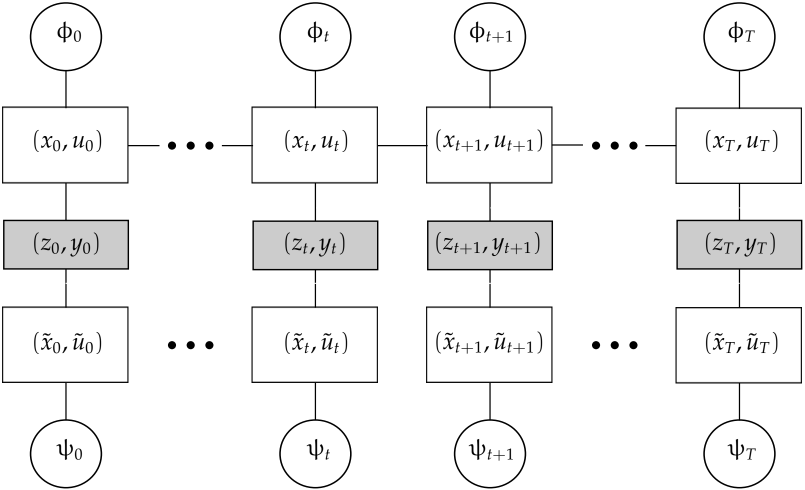
Stopping criterion

- primal residual $r^k = (x^k, u^k) - (\tilde{x}^k, \tilde{u}^k)$
- dual residual $s^k = \rho((\tilde{x}^k, \tilde{u}^k) - (\tilde{x}^{k-1}, \tilde{u}^{k-1}))$
- both converge to zero
- stopping criterion:
$$\|r^k\|_2 \leqslant \epsilon^{\text{pri}}, \quad \|s^k\|_2 \leqslant \epsilon^{\text{dual}}$$
  with tolerances $\epsilon^{\text{pri}} > 0$ and $\epsilon^{\text{dual}} > 0$

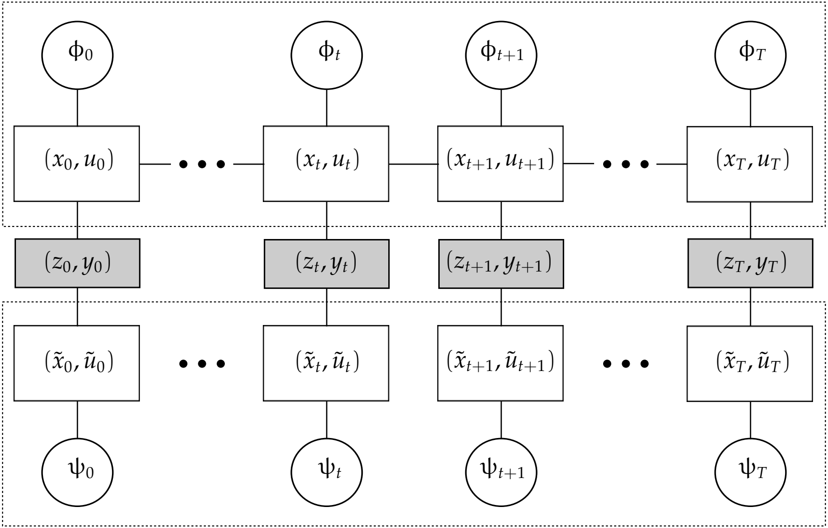
Consensus form graph



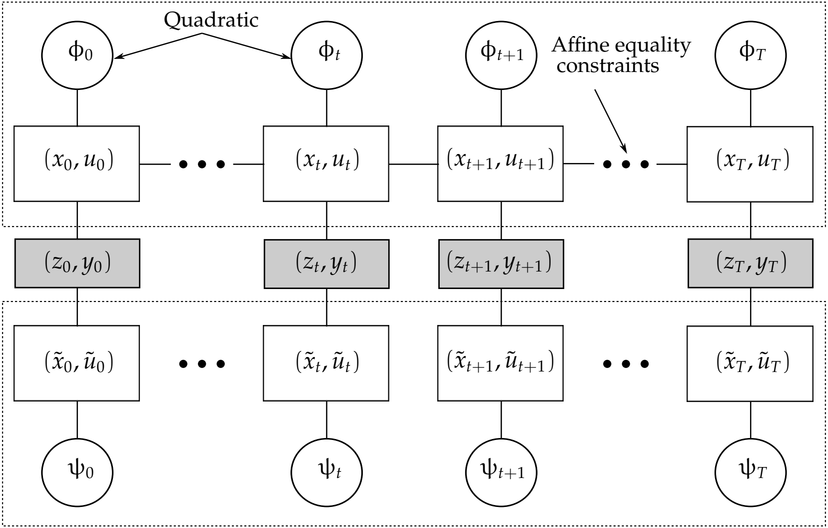
With dual variables



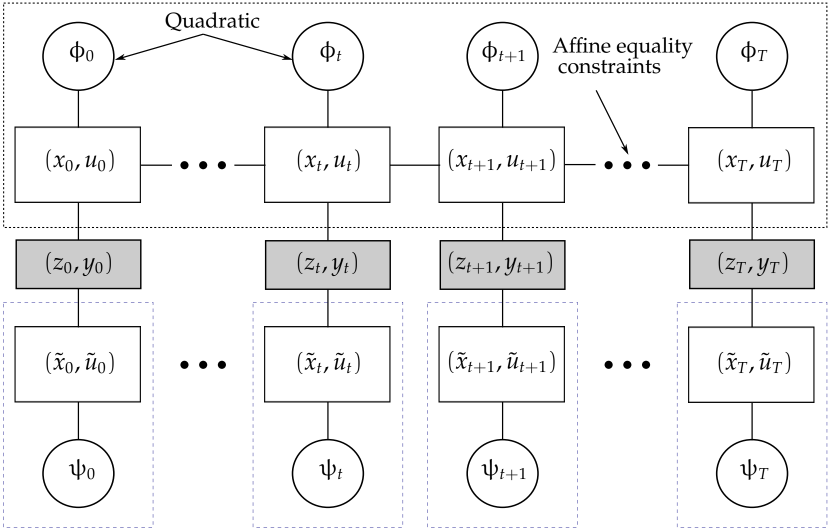
Douglas-Rachford Splitting



Sub-problems



Sub-problems



Linear quadratic step

- OSC first step is solving a linearly constrained quadratic problem

$$\begin{array}{ll} \text{minimize} & (1/2)w^T E w + f^T w \\ \text{subject to} & Gw = h \end{array}$$

  over variable $w \in \mathbf{R}^{(T+1)(n+m)}$

- E has block structure
- optimality conditions: KKT system

$$\left[\begin{array}{cc} E & G^T \\ G & 0 \end{array}\right] \left[\begin{array}{c} w \\ \lambda \end{array}\right] = \left[\begin{array}{c} -f \\ h \end{array}\right]$$

  $\lambda \in \mathbf{R}^{(T+1)n}$ dual variable associated with $Gw = h$

- in each iteration of OSC we solve KKT system *with same KKT matrix*

Sparse LDL^T decomposition

- factor KKT matrix as

$$\begin{bmatrix} E & G^T \\ G & 0 \end{bmatrix} = PLDL^T P^T$$

  - P is a permutation matrix
  - L is unit lower triangular
  - D is diagonal
- P chosen to yield a factor L with few nonzeros
- can choose P such that L is block banded
- factorize, then cache P, L, D^{-1}

Solve step

- solve KKT system using

$$\begin{bmatrix} w \\ \lambda \end{bmatrix} = P\left(L^{-T}\left(D^{-1}\left(L^{-1}\left(P^T\begin{bmatrix} -f \\ h \end{bmatrix}\right)\right)\right)\right)$$

  - multiplication by L^{-1} is forward substitution
  - multiplication by L^{-T} is backward substitution
- **these operations do not require division**
- factor cost: $\mathcal{O}(T(m+n)^3)$, solve cost: $\mathcal{O}(T(m+n)^2)$
- same as Riccati recursion, but (much) more general
- (we also use regularization and iterative refinement)

Non-quadratic prox step

- OSC second step separable across time
- solve for each t:

$$\text{minimize} \quad \psi_t(\tilde{x}_t, \tilde{u}_t) + (\rho/2)\|(\tilde{x}_t, \tilde{u}_t) - (v_t, w_t)\|_2^2$$

  over $\tilde{x}_t \in \mathbf{R}^n$ and $\tilde{u}_t \in \mathbf{R}^m$
- in many cases we have analytic or semi-analytic solutions
- **can be solved in parallel**

OSC summary

in each step:

1. solve linear-quadratic regulator problem
2. $T + 1$ parallel prox steps
3. dual update

Usage scenarios

- cold start
  - solve optimal control problem once
- warm start
  - solve many times with similar data
  - initialize algorithm using previous solution
- constant quadratic
  - solve many times, where Q_t, R_t, S_t, A_t, B_t do not change
  - perform LDL^T factorization once, offline
  - can yield **division free** algorithm
- warm start constant quadratic
  - computational savings stack

Outline

Examples

- three examples, three instances of each
- timing results for
  - cold start: initialize variables to zero
  - warm start: perturb x_{init}
- at termination no instance was more than 1% suboptimal
- implemented in C
- Tim Davis' AMD and LDL packages for factorization and solve steps
- run on 4 core Intel Xeon processor (3.4GHz, 16Gb of RAM)
- and, for fun, Rasberry Pi

Box-constrained quadratic optimal control

- box-constrained problem:

$$\begin{array}{ll} \text{minimize} & (1/2)\sum_{t=0}^{T}(x_t^T Q x_t + u_t^T R u_t) \\ \text{subject to} & x_{t+1} = Ax_t + Bu_t, \quad t = 0,\dots,T-1 \\ & x_0 = x_{\text{init}} \\ & \|u_t\|_\infty \leqslant 1 \end{array}$$

  $Q \succeq 0$ and $R \succ 0$

- data randomly generated; A scaled so that $\rho(A) = 1$
- x_{init} scaled so inputs saturated for at least 2/3 of horizon
- $\psi_t(x_t, u_t) = I_{\|u_t\|_\infty \leqslant 1}$, so

$$\text{prox}_{\psi_t}(v, w) = (v, \underset{[-1,1]}{\textbf{sat}}(w))$$

Results

(all times in milliseconds)

	small	**medium**	**large**
state dimension n	5	20	50
input dimension m	2	5	20
horizon length T	10	20	30
total variables	77	525	2170
CVX solve time	400	500	3400
fast MPC solve time	1.5	14.2	2710

Results

(all times in milliseconds)

	small	**medium**	**large**
state dimension n	5	20	50
input dimension m	2	5	20
horizon length T	10	20	30
total variables	77	525	2170
CVX solve time	400	500	3400
fast MPC solve time	1.5	14.2	2710
factorization time	0.1	1.3	16.8
KKT solve time	0.0	0.1	0.9
OSC iterations	92	46	68
OSC solve time	0.4	4.4	60.5
warm start OSC iterations	72.6	35.1	39.5
warm start OSC solve time	0.3	3.4	35.2

Multi-period portfolio optimization

- manage a portfolio of n assets over $t = 0, \dots, T$
- $x_t \in \mathbf{R}^n$ vector of portfolio positions at time t (in dollars)
  - $(x_t)_i < 0$: short position in asset i in period t
- $u_t \in \mathbf{R}^n$ vector of trades at time t (in dollars)
  - $(u_t)_i < 0$: asset i is sold in period t
- dynamics

$$x_{t+1} = \mathbf{diag}(r_t)(x_t + u_t), \quad t = 0, \dots, T-1$$

- $r_t > 0$ (estimated) returns in period t

Stage cost

$$\overbrace{\mathbf{1}^T u_t}^{\text{gross cash in}} + \overbrace{u_t^T \,\mathbf{diag}(s) u_t}^{\text{price-impact}} + \overbrace{(x_t + u_t)^T \Sigma (x_t + u_t)}^{\text{quadratic risk}} + \overbrace{\kappa^T |u_t|}^{\text{bid-ask spread}} + \overbrace{I_{\mathcal{C}_t}(x_t, u_t)}^{\text{trading constraint}}$$

- $\kappa \geqslant 0$, $s \geqslant 0$, and $\Sigma \succeq 0$ are data
- negative stage cost means (risk-adjusted) revenue extracted
- trading constraints
  - long-only: $\mathcal{C}_t = \{(x_t, u_t) \mid x_t + u_t \geqslant 0\}$, $t \neq T$
  - liquidate position: $\mathcal{C}_T = \{(x_T, u_T) \mid x_T + u_T = 0\}$

Splitting

$$\underbrace{\overbrace{\mathbf{1}^T u_t}^{\text{gross cash in}} + \overbrace{u_t^T \,\mathbf{diag}(s_t) u_t}^{\text{price-impact}} + \lambda \overbrace{(x_t+u_t)^T \Sigma_t (x_t+u_t)}^{\text{quadratic risk}}}_{\phi_t} + \underbrace{\overbrace{\kappa^T |u_t|}^{\text{bid-ask spread}} + \overbrace{I_{\mathcal{C}_t}(x_t, u_t)}^{\text{trading constraint}}}_{\psi_t}$$

note: ψ_t separable across assets

Proximal operator for ψ_t

- for $t < T$ prox step given by solution to

$$\begin{array}{ll} \text{minimize} & \kappa_i |u|_i + (\rho/2)\left((x_i - v_i)^2 + (u_i - w_i)^2\right) \\ \text{subject to} & x_i + u_i \geqslant 0 \end{array}$$

  with scalar variables x_i and u_i

- solution easily expressed using soft-thresholding operator $S_\gamma(z)$

$$S_\gamma(z) = \underset{y}{\operatorname{argmin}} \left(\gamma|y| + (1/2)(y-z)^2\right) = z(1 - \gamma/|z|)_+$$

Results

(all times in milliseconds)

	small	**medium**	**large**
number of assets n	10	30	50
horizon length T	30	60	100
total variables	620	3660	10100
CVX solve time	800	2100	10750

Results

(all times in milliseconds)

	small	medium	large
number of assets n	10	30	50
horizon length T	30	60	100
total variables	620	3660	10100
CVX solve time	800	2100	10750
factorization time	0.7	13.3	73.6
KKT solve time	0.1	0.7	3.2
OSC iterations	27	41	53
OSC solve time	1.5	30.8	177.7
warm start OSC iterations	5.1	5.9	4.8
warm start OSC solve time	0.3	4.4	16.1

Supply chain management

- single commodity supply chain on a directed graph
  - n nodes: warehouses or storage locations
  - m edges: shipment links between warehouses, sources, and sinks
- $x_t \in \mathbf{R}^n_+$ amount of the commodity stored in warehouses
- $u_t \in \mathbf{R}^m_+$ amount shipped across links
- dynamics

$$x_{t+1} = x_t + (B^+ - B^-)u_t$$

  - $B^+_{ij} = 1$ if edge j enters node i
  - $B^-_{ij} = 1$ if edge j leaves node i

Stage cost

$$\underbrace{\overbrace{q_t^T x + \tilde{q}_t^T x^2}^{\text{storage cost}} + \overbrace{r_t^T u_t}^{\text{transportation cost}}}_{\phi_t} + \underbrace{\overbrace{I_{0 \leqslant x_t \leqslant C}}^{\text{warehouse capacities}} + \overbrace{I_{0 \leqslant u_t \leqslant U}}^{\text{link capacities}} + \overbrace{I_{B^- u_t \leqslant x_t}}^{\text{can't ship more than on hand}}}_{\psi_t}$$

transportation costs include

- cost of acquisition
- revenue from sales

prox step of ψ_t solved via saturation and bisection

Results

(all times in milliseconds)

	small	**medium**	**large**
warehouses n	10	20	40
edges m	25	118	380
horizon length T	20	20	20
total variables	735	2898	8820
CVX solve time	500	1200	3300

Results

(all times in milliseconds)

	small	**medium**	**large**
warehouses n	10	20	40
edges m	25	118	380
horizon length T	20	20	20
total variables	735	2898	8820
CVX solve time	500	1200	3300
factorization time	0.3	1.3	4.7
KKT solve time	0.0	0.1	0.3
single-thread prox step time	0.1	0.4	1.3
multi-thread prox step time	0.0	0.1	0.4
OSC iterations	82	77	116
OSC solve time	4.6	19.1	88.1
warm start OSC iterations	21.9	31.0	24.2
warm start OSC solve time	1.2	7.5	18.5

Outline

Summary

- decompose convex optimal control problem into
  - convex linear quadratic control problem
  - time-separable nonquadratic problems
- yields fast, reliable algorithm
  - small problems solved in microseconds
  - large problems solved in milliseconds
- if dynamics matrices don't change, yields division-free method
- can be improved by diagonal scaling, computed on-line or off-line