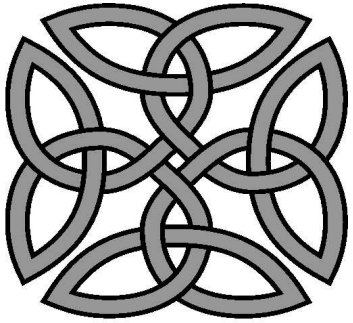


School of Mathematics

Parallel Solution Techniques in Very Large Scale Financial Planning Problems

Jacek Gondzio
In collaboration with **Andreas Grothey**

Coimbra, 26 October, 2007

Interior Point Methods

Theory: convergence in $\mathcal{O}(\sqrt{n})$ or $\mathcal{O}(n)$ iterations
Practice: convergence in $\mathcal{O}(\log n)$ iterations

Expected number of IPM iterations:

Problem Dimension	LP	QP
1,000	$\approx$ 20	$\approx$ 10
10,000	20 - 30	10 - 15
100,000	30 - 40	15 - 20
1,000,000	40 - 50	20 - 25
10,000,000	50 - 60	20 - 25
100,000,000	60 - 70	25 - 30
1000,000,000	70 - 80	25 - 30

Motivation

Truly large-scale optimization problems are usually

- sparse
- block-structured
- often have nested structure

(due to e.g. dynamics, uncertainty, spatial distribution etc.)

Exploiting structure and inherent parallelism is key to efficient implementation of IPMs:

- Faster linear algebra
- Reduced memory use
- Parallel machines are often cheaper to install than comparable serial machines: e.g. a cluster of PCs running Linux.

Outline

- Interior Point Methods for non-convex nonlinear programs
- Exploiting structure in very large scale optimization
- Object-Oriented Parallel Solver **OOPS**
- Asset and Liability Management
  - multi-period structure
  - using semi-variance and higher-order moments
- Computational results:
  - small problems ($\leq$ 1,000,000 variables): OOPS vs Cplex
  - large problems ($\geq$ 1,000,000 variables): parallel OOPS
- Conclusions

Interior Point Methods

Marsten, Subramanian, Saltzman, Lustig and Shanno:
"Interior point methods for linear programming:
Just call Newton, Lagrange, and Fiacco and McCormick!",
Interfaces 20 (1990) No 4, pp. 105–116.

- **Fiacco & McCormick (1968)**
  inequality constraints $\longrightarrow$ logarithmic barrier;
  a sequence of unconstrained minimizations
- **Lagrange (1788)**
  equality constraints $\longrightarrow$ multipliers;
- **Newton (1687)**
  solve unconstrained minimization problems;

IPM for LP

LP

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax = b,\ x \geq 0. \end{array}$$

$\longrightarrow$

Barrier LP

$$\begin{array}{ll} \min & c^T x - \mu \sum_{j=1}^{n} \ln x_j \\ \text{s.t.} & Ax = b. \end{array}$$

Write the **Lagrangian**

$$L(x, y, \mu) = c^T x - y^T (Ax - b) - \mu \sum_{j=1}^{n} \ln x_j,$$

and the conditions for a stationary point

$$\begin{array}{rcl} \nabla_x L(x, y, \mu) & = & c - A^T y - \mu X^{-1} e = 0 \\ \nabla_y L(x, y, \mu) & = & Ax - b = 0, \end{array}$$

where $X^{-1} = diag\{x_1^{-1}, x_2^{-1}, \cdots, x_n^{-1}\}$.

IPM for LP (cont'd)

Denote

$$z = \mu X^{-1} e, \quad \text{i.e.} \quad XZe = \mu e.$$

The **First Order Optimality Conditions** are:

$$\begin{aligned} Ax &= b, \\ A^T y + z &= c, \\ XZe &= \mu e. \end{aligned}$$

Use **Newton Method**: For a given point (x, y, z) find the Newton direction $(\Delta x, \Delta y, \Delta z)$ by solving the system of linear equations:

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ Z & 0 & X \end{bmatrix} \cdot \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} b - Ax \\ c - A^T y - z \\ \mu e - XZe \end{bmatrix}.$$

Solving LP by IPM

$$\min c^\top x \;\; \text{s.t.} \;\; Ax = b, \;\; x \geq 0 \qquad \text{(LP)}$$

Optimality conditions:

$$\begin{aligned} c - A^\top y - z &= 0 \\ Ax &= b \\ XZe &= 0 \quad (\mu e) \\ x, z &\geq 0 \end{aligned}$$

$\Rightarrow$ Newton Step:

$$\begin{bmatrix} -\Theta & A^\top \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} c - A^\top y - \mu X^{-1} e \\ b - Ax \end{bmatrix} \qquad \text{(NS-LP)}$$

where

$$\Theta = X^{-1} Z, \quad X = \text{diag}(x), \quad Z = \text{diag}(z)$$

Solving QP by IPM

$$\min c^\top x + \tfrac{1}{2} x^\top Q x \;\; \text{s.t.} \;\; Ax = b, \; x \geq 0 \qquad \text{(QP)}$$

Optimality conditions:

$$\begin{aligned} c + Qx - A^\top y - z &= 0 \\ Ax &= b \\ XZe &= 0 \quad (\mu e) \\ x, z &\geq 0 \end{aligned}$$

$\Rightarrow$ Newton Step:

$$\begin{bmatrix} -Q - \Theta & A^\top \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} Qx + c - A^\top y - \mu X^{-1} e \\ b - Ax \end{bmatrix} \qquad \text{(NS-QP)}$$

where

$$\Theta = X^{-1} Z, \quad X = \text{diag}(x), \quad Z = \text{diag}(z)$$

Solving NLP by IPM

Add slacks to nonlinear inequalities:

$$\min f(x) \;\text{ s.t. }\; g(x) + z = 0,\; z \geq 0 \quad (\text{i.e. } g(x) \leq 0) \qquad \text{(NLP)}$$

Optimality conditions:

$$\begin{aligned} \nabla f(x) + \nabla g(x)^\top y &= 0 \\ g(x) + z &= 0 \\ YZe &= 0 \quad (\mu e) \\ x, z &\geq 0 \end{aligned}$$

$\Rightarrow$ Newton Step:

$$\begin{bmatrix} Q(x,y) & A(x)^\top \\ A(x) & -\Theta \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} -\nabla f(x) - A(x)^\top y \\ -g(x) - \mu Y^{-1} e \end{bmatrix} \qquad \text{(NS-NLP)}$$

where

$$Q(x,y) = \nabla^2_{xx}(f(x) + y^\top g(x)), \quad A(x) = \nabla g(x)$$

$$\Theta = ZY^{-1}, \quad Y = \text{diag}(y), \quad Z = \text{diag}(z)$$

KKT systems in IPMs for LP, QP and NLP

LP
$$\begin{bmatrix} \Theta^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} f \\ d \end{bmatrix}$$

QP
$$\begin{bmatrix} Q + \Theta^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} f \\ d \end{bmatrix}$$

NLP
$$\begin{bmatrix} Q(x,y) + \Theta_P^{-1} & A(x)^T \\ A(x) & -\Theta_D \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} f \\ d \end{bmatrix}$$

Linear Algebra of IPMs

$\longrightarrow$ solve symmetric (indefinite) system of equations.

Primal Block-Angular Structure:

$$Q = \left[\begin{array}{cc} \blacksquare & \\ & \blacksquare \end{array}\right], \quad A = \left[\begin{array}{cc} \blacksquare & \\ & \blacksquare \\ \blacksquare & \blacksquare \end{array}\right] \quad and \quad A^T = \left[\begin{array}{ccc} \blacksquare & & \\ & \blacksquare & \\ & \blacksquare & \blacksquare \end{array}\right]$$

Reorder blocks: $\{1, 3;\ 2, 4;\ 5\}$.

$$H = \left[\begin{array}{cc|cc|c} \blacksquare & & \blacksquare & & \blacksquare \\ & \blacksquare & & \blacksquare & \blacksquare \\ \hline \blacksquare & & & & \\ & \blacksquare & & & \\ \blacksquare & \blacksquare & & & \end{array}\right], \qquad PHP^T = \left[\begin{array}{cc|cc|c} \blacksquare & \blacksquare & & & \blacksquare \\ \blacksquare & & & & \\ \hline & & \blacksquare & \blacksquare & \blacksquare \\ & & \blacksquare & & \\ \hline \blacksquare & & \blacksquare & & \end{array}\right]$$

Dual Block-Angular Structure:

$$Q = \left[\begin{array}{ccc} \blacksquare & & \\ & \blacksquare & \\ & & \blacksquare \end{array}\right], \quad A = \left[\begin{array}{ccc} \blacksquare & & \blacksquare \\ & \blacksquare & \blacksquare \end{array}\right] \quad and \quad A^T = \left[\begin{array}{cc} \blacksquare & \\ & \blacksquare \\ \blacksquare & \blacksquare \end{array}\right]$$

Reorder blocks: $\{1, 4;\ 2, 5;\ 3\}$.

$$H = \left[\begin{array}{ccc|cc} \blacksquare & & & \blacksquare & \\ & \blacksquare & & & \blacksquare \\ & & \blacksquare & \blacksquare & \blacksquare \\ \hline \blacksquare & & \blacksquare & & \\ & \blacksquare & \blacksquare & & \end{array}\right], \qquad PHP^T = \left[\begin{array}{ccc|cc} \blacksquare & \blacksquare & & & \\ \blacksquare & & & & \blacksquare \\ \hline & & \blacksquare & \blacksquare & \\ & & \blacksquare & & \blacksquare \\ \hline & \blacksquare & & \blacksquare & \blacksquare \end{array}\right]$$

Row & Column Bordered Block-Diag Structure:

$$Q = \left[\begin{array}{ccc} \blacksquare & & \\ & \blacksquare & \\ & & \blacksquare \end{array}\right], \quad A = \left[\begin{array}{ccc} \blacksquare & & \blacksquare \\ & \blacksquare & \blacksquare \\ \blacksquare & \blacksquare & \blacksquare \end{array}\right] \quad and \quad A^T = \left[\begin{array}{ccc} \blacksquare & & \blacksquare \\ & \blacksquare & \blacksquare \\ \blacksquare & \blacksquare & \blacksquare \end{array}\right]$$

Reorder blocks: $\{1,4;\ 2,5;\ 3,6\}$.

$$H = \left[\begin{array}{ccc|ccc} \blacksquare & & & \blacksquare & & \blacksquare \\ & \blacksquare & & & \blacksquare & \blacksquare \\ & & \blacksquare & \blacksquare & \blacksquare & \blacksquare \\ \hline \blacksquare & & \blacksquare & & & \\ & \blacksquare & \blacksquare & & & \\ \blacksquare & \blacksquare & \blacksquare & & & \end{array}\right], \qquad PHP^T = \left[\begin{array}{ccc|ccc} \blacksquare & \blacksquare & & & & \blacksquare \\ \blacksquare & & & \blacksquare & \blacksquare & \\ \hline & & \blacksquare & & \blacksquare & \blacksquare \\ & \blacksquare & & \blacksquare & & \\ \hline & & \blacksquare & & \blacksquare & \blacksquare \\ \blacksquare & & \blacksquare & & & \blacksquare \end{array}\right]$$

Example: Bordered Block-Diagonal Structure

$$\underbrace{\begin{pmatrix} \Phi_1 & & & B_1^\top \\ & \ddots & & \vdots \\ & & \Phi_n & B_n^\top \\ B_1 & \dots & B_n & \Phi_0 \end{pmatrix}}_{\Phi} =$$

$$= \underbrace{\begin{pmatrix} L_1 & & & \\ & \ddots & & \\ & & L_n & \\ L_{1,0} & \dots & L_{n,0} & L_0 \end{pmatrix}}_{L} \underbrace{\begin{pmatrix} D_1 & & & \\ & \ddots & & \\ & & D_n & \\ & & & D_0 \end{pmatrix}}_{D} \underbrace{\begin{pmatrix} L_1^\top & & & L_{1,0}^\top \\ & \ddots & & \vdots \\ & & L_n^\top & L_{n,0}^\top \\ & & & L_0^\top \end{pmatrix}}_{L^\top}$$

The blocks $\Phi_i,\ i = 0, 1, ..., n$ are KKT systems.

Example: Bordered Block-Diagonal Structure

- Cholesky-like factors obtained by Schur-complement:

$$\begin{aligned} \Phi_i &= L_i D_i L_i^\top \\ L_{i,0} &= B_i L_i^{-\top} D_i^{-1}, \quad i = 1..n \\ C &= \Phi_0 - \sum_{i=1}^n L_{i,0} D_i L_{i,0}^\top \;=\; L_0 D_0 L_0^\top \end{aligned}$$

- And the system $\Phi x = b$ is solved by

$$\begin{aligned} z_i &= L_i^{-1} b_i \\ z_0 &= L_0^{-1}(b_0 - \sum L_{i,0} z_i) \\ y_i &= D_i^{-1} z_i \\ x_0 &= L_0^{-\top} y_0 \\ x_i &= L_i^{-\top}(y_i - L_{i,0}^\top x_0) \end{aligned}$$

- Operations (Cholesky, Solve, Product) performed on sub-blocks

Abstract Linear Algebra for IPMs

Execute the operation

"solve (reduced) KKT system"

in IPMs for LP, QP and NLP.

It works like the "backslash" operator in MATLAB.

Assumptions:

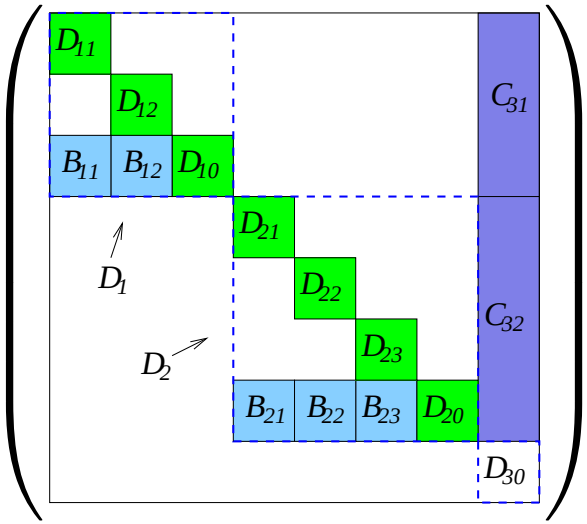
Q and A are block-structured

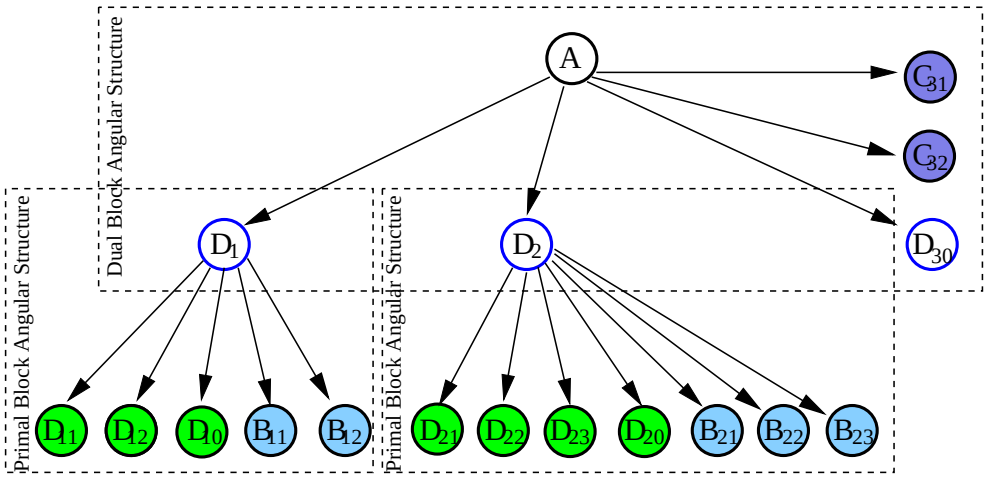
Linear Algebra of IPMs

At each iteration solve this system for several right-hand-sides

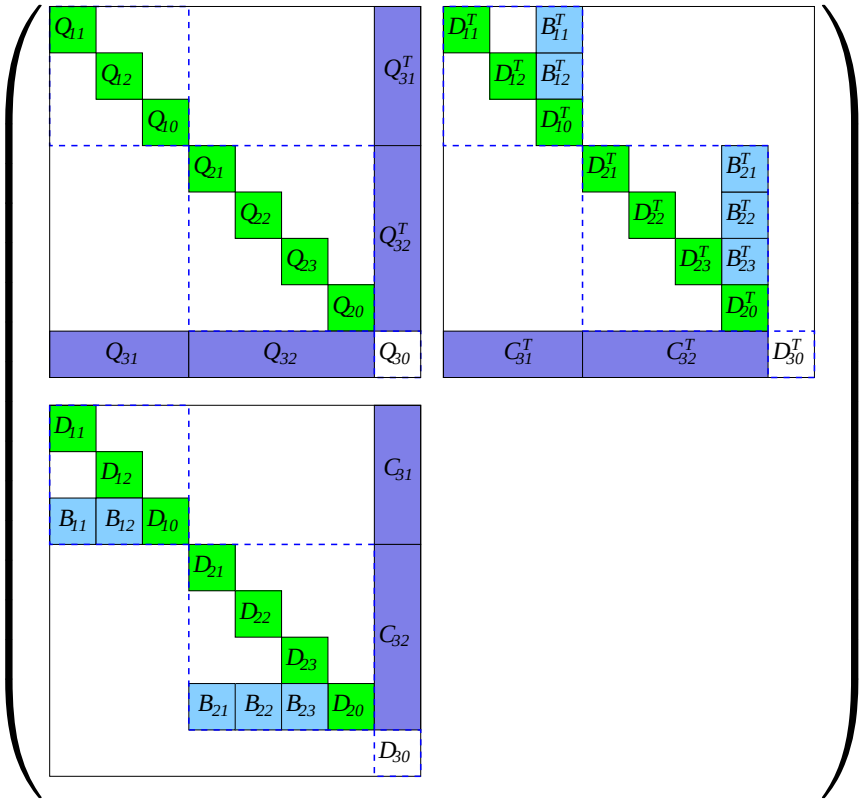
$$\underbrace{\begin{bmatrix} -Q - \Theta_P^{-1} & A^\top \\ A & \Theta_D \end{bmatrix}}_{\Phi\,(NLP)} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} f \\ d \end{bmatrix}$$

Tree representation of matrix A:

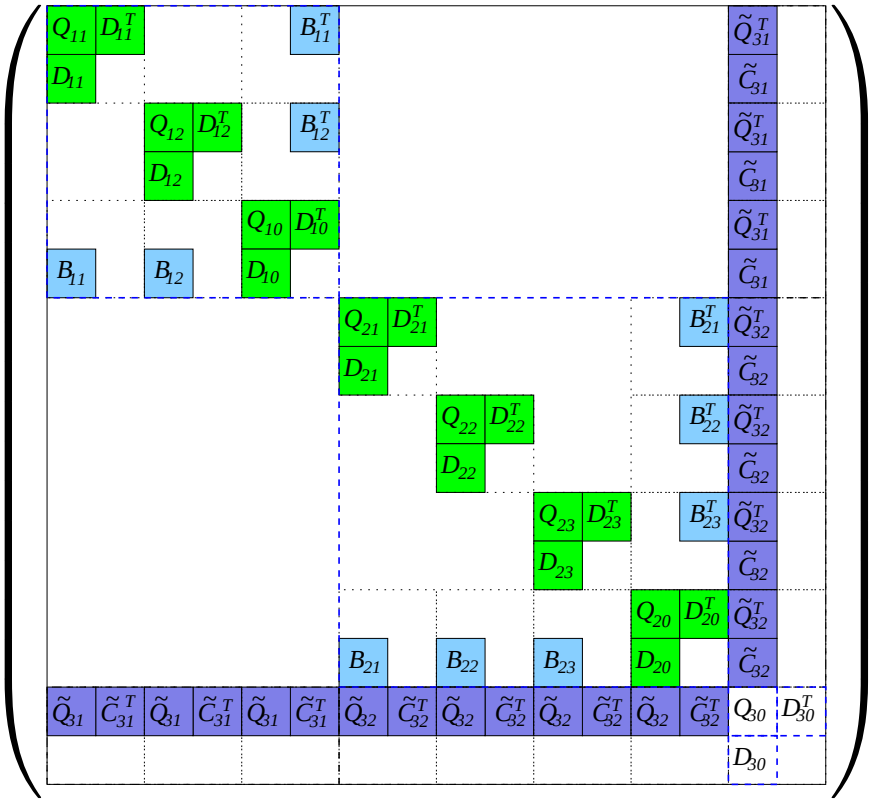




Structures of A and Q imply structure of Φ:



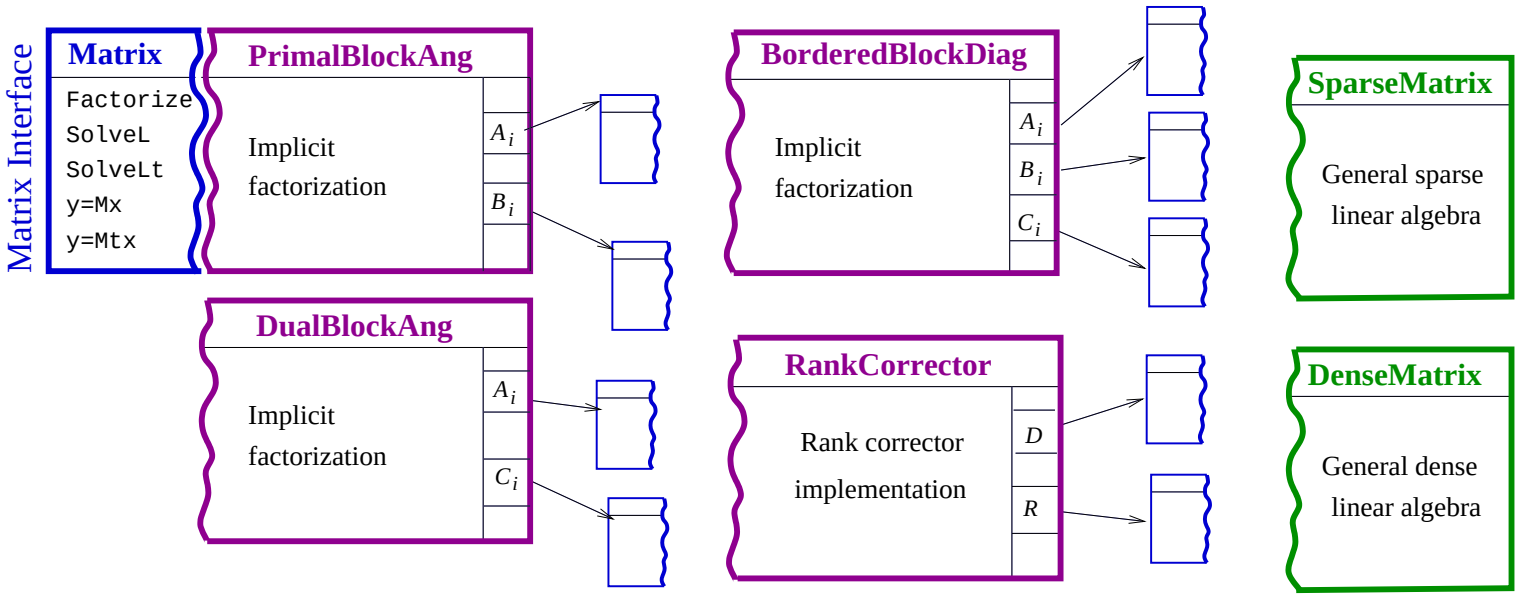
$$\begin{pmatrix} Q & A^\top \\ A & 0 \end{pmatrix}$$



$$P \begin{pmatrix} Q & A^\top \\ A & 0 \end{pmatrix} P^{-1}$$

OOPS: Object-oriented linear algebra for IPM

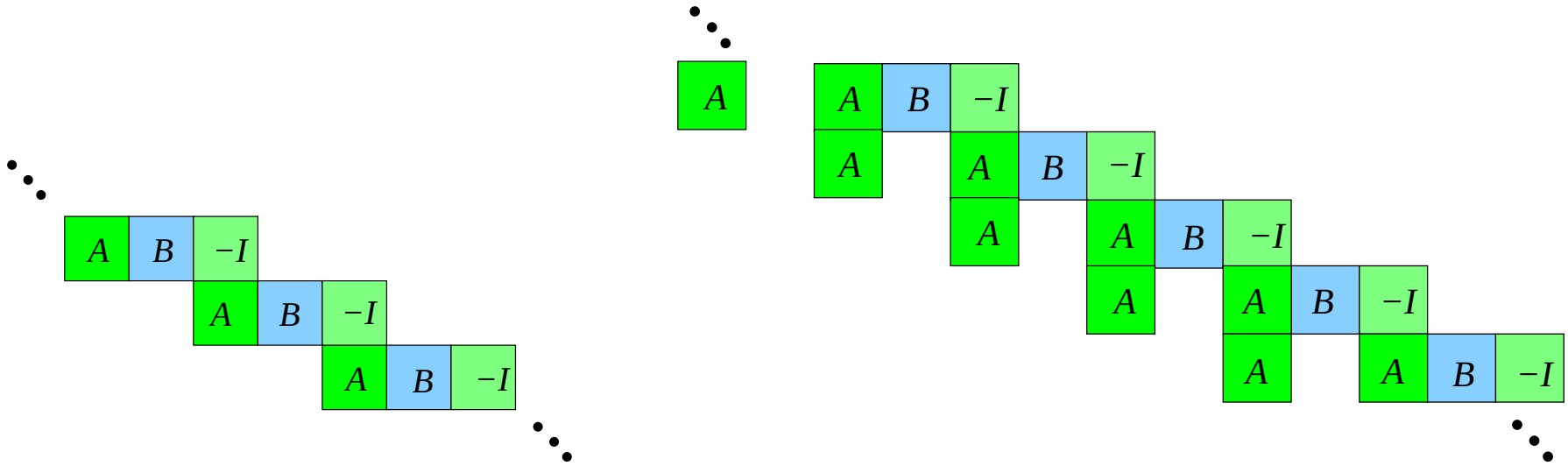
- Every node in the *block elimination tree* has its own linear algebra implementation (depending on its type)
- Each implementation is a realisation of an abstract linear algebra interface.
- Different implementations are available for different structures



$\Rightarrow$ Rebuild *block elimination tree* with matrix interface structures

Sources of Structure

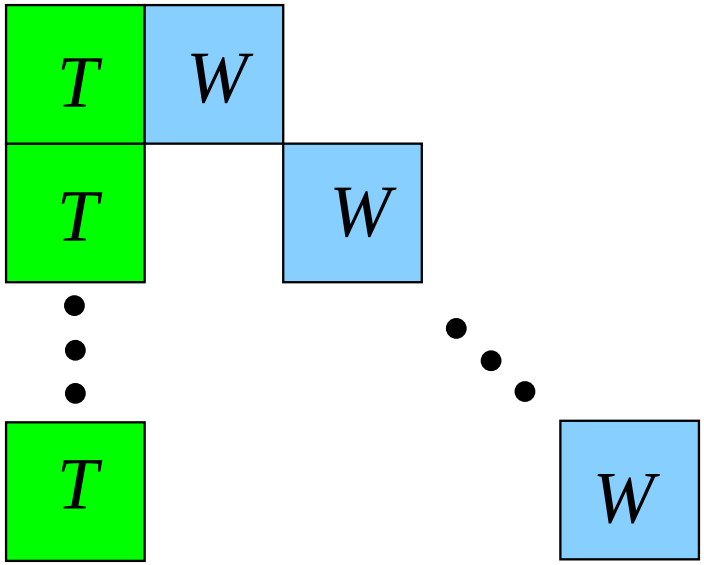
Dynamics → Staircase structure



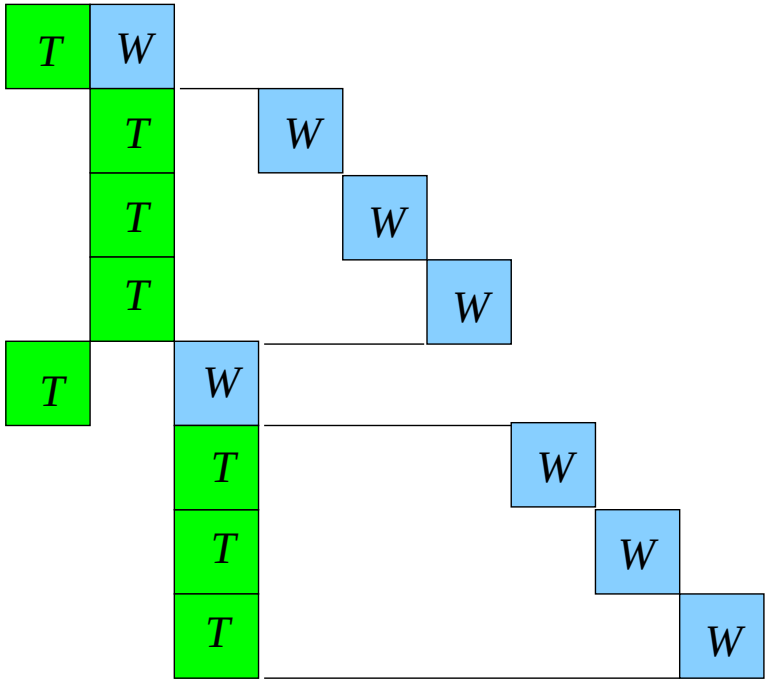
$$x_{t+1} = A_t x_t + B_t u_t \qquad x_{t+1} = A_t^{t+1} x_t + \ldots + A_{t-p}^{t+1} x_{t-p} + B_t u_t$$

Sources of Structure

Uncertainty → Block-angular structure



$$T_i x^1 + W_i\, y_i = b_i$$



$$T_{l_t} x_{a(l_t)} + W_{l_t}\, x_{l_t} = b_{l_t}$$

Financial Planning Problems

- A set of assets $\mathcal{J} = \{1, ..., J\}$ given (bonds, stock, real estate)
- At every stage $t = 0, ..., T-1$ we can buy or sell different assets
- The return of asset j at stage t is *uncertain*

Investment decisions: **what to buy or sell, at which time stage**

Objectives:

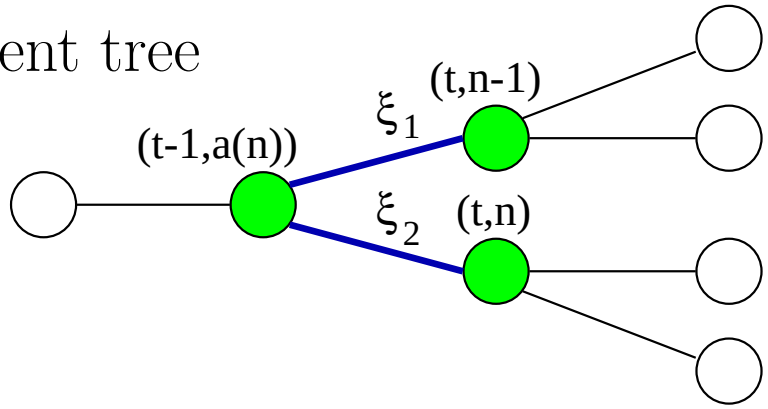
- maximize the final wealth
- minimize the associated risk

$\Rightarrow$ Mean Variance formulation:
$\max \mathbb{E}(X) - \rho \mathrm{Var}(X)$

$\Rightarrow$ Stochastic Program: $\Rightarrow$ formulate deterministic equivalent

- standard QP, but huge
- extentions: **nonlinear risk measures** (log utility, skewness)

Modelling: use event tree



and decision variables associated with its nodes (t, n).
Let $a(t, n)$ denote the **ancestor** of node (t, n).
For asset $j \in \mathcal{J}$ at node (t, n)

$$x_{j,t,n},\ x^b_{j,t,n},\ x^s_{j,t,n}$$

denote the amount of asset **held, bought, sold** at node (t, n).
The **inventory equation** for asset j at node (t, n) (for $1 \leq t \leq T$):

$$x_{j,t,n} = (1 + r_{j,t,n}) \cdot x_{j,t-1,a(t,n)} + x^b_{j,t,n} - x^s_{j,t,n},$$

where $r_{j,t,n}$ is a return of asset j corresponding to moving from node $(t-1, a(t, n))$ to node (t, n) in the event tree.

Asset/Liability Management Problem (ALM)

Final wealth y is the expected value of the final portfolio converted into cash

$$y = I\!E((1-c_t)\sum_{j=1}^{J} v_j x^h_{T,j}) = (1-c_t)\sum_{i\in L_T} p_i \sum_{j=1}^{J} v_j x^h_{i,j},$$

while risk is expressed as its variance $[\mathrm{Var}(X) = I\!E(X^2) - (I\!E(X))^2]$:

$$\mathrm{Var}((1-c_t)\sum_{j=1}^{J} v_j x^h_{T,j}) = \sum_{i\in L_T} p_i (1-c_t)^2 [\sum_j v_j x^h_{i,j}]^2 - y^2.$$

ALM formulated as an optimization problem

The ALM problem can be expressed as

$$\max_{x,y\geq 0} \quad y - \rho[\sum_{i\in L_T} p_i[(1-c_t)\sum_j v_j x^h_{i,j}]^2 - y^2]$$

$$\begin{aligned}
\text{s.t.} \quad (1-c_t)\textstyle\sum_{i\in L_T} p_i \sum_j v_j x^h_{i,j} &= y \\
(1+r_{i,j})x^h_{a(i),j} &= x^h_{i,j} - x^b_{i,j} + x^s_{i,j}, \ \forall i \neq 0, j \\
\textstyle\sum_j (1+c_t) v_j x^b_{i,j} &= \textstyle\sum_j (1-c_t) v_j x^s_{i,j}, \ \forall i \neq 0 \\
\textstyle\sum_j (1+c_t) v_j x^b_{0,j} &= b.
\end{aligned}$$

see, e.g., Steinbach, *Markowitz revisited ...*, SIAM Review 43 (2001).

ALM (cont'd)

Denote $x_i = (x^s_{i,1}, x^b_{i,1}, x^h_{i,1}, \ldots, x^s_{i,J}, x^b_{i,J}, x^h_{i,J})$, and define matrices

$$A = \begin{pmatrix} 1 & -1 & 1 & & & & \\ & & & \ddots & & & \\ & & & & 1 & -1 & 1 \\ -c^s_1 & c^b_1 & 0 & \cdots & -c^s_J & c^b_J & 0 \end{pmatrix}, \quad B_i = \begin{pmatrix} 0 & 0 & 1+r_{i,1} & & & & \\ & & & \ddots & & & \\ & & & & 0 & 0 & 1+r_{i,J} \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \end{pmatrix}$$

$$Q_i \in I\!R^{3J\times 3J} \; : \; \begin{cases} (Q_i)_{3j,3k} = p_i(1-c_t)^2 v_j v_k, \; j,k = 1,\ldots,J, & i \in L_T \\ Q_i = 0, & i \notin L_T \end{cases}$$

$$d_i \in R^{1\times 3J} \; : \; (d_i)_{3j} = (1-c_t) p_i v_j,$$

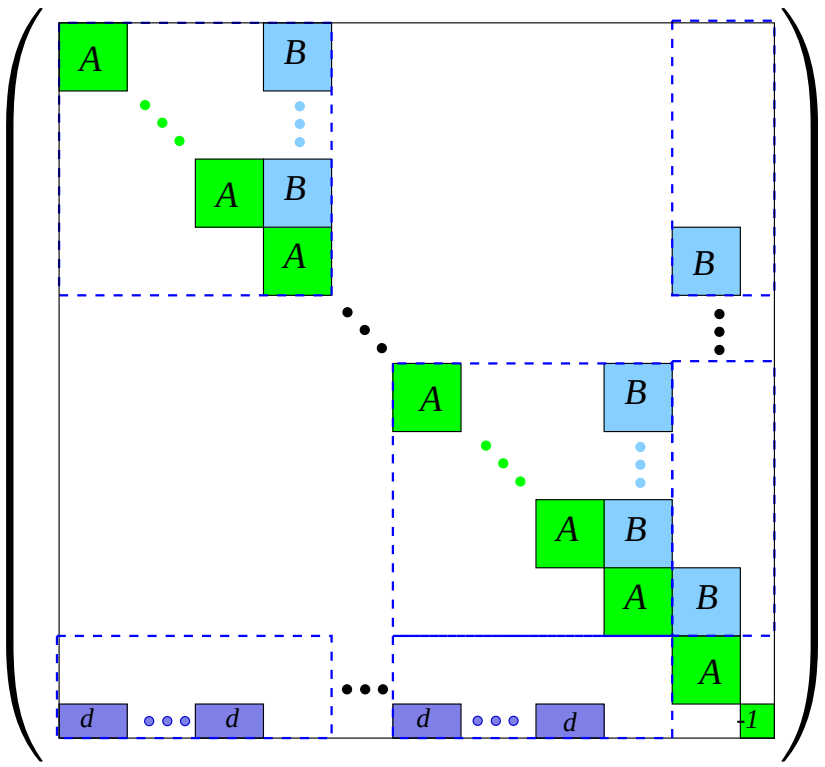
Rewrite ALM:

$$\max_{x,y\geq 0} \; y - \rho[\sum_{i\in L_T} x_i^T Q_i x_i - y^2] \quad \text{s.t.} \; \begin{aligned} \textstyle\sum_{i\in L_T} d_i^T x_i &= y \\ B_{a(i)} x_{a(i)} &= A x_i \quad \forall i \neq 0 \\ A x_0 &= b e_{J+1}. \end{aligned}$$

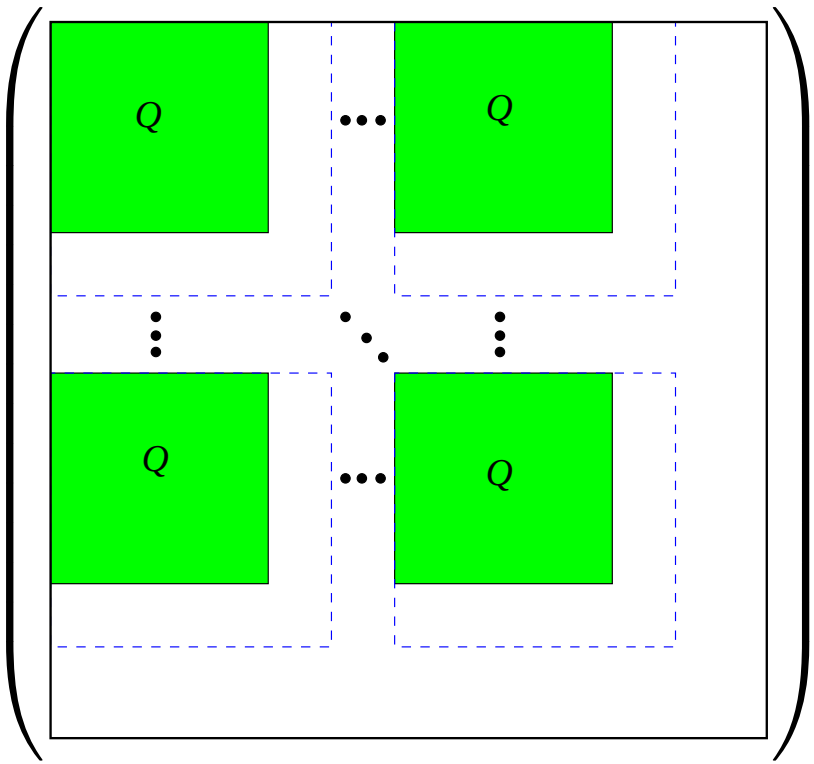
ALM (cont'd)

Assemble vectors $x_i, i \in L$ and y into $x = (x_{\sigma(0)}, x_{\sigma(1)}, \ldots, x_{\sigma(|L|-1)}, y)$, where σ is a reverse depth-first permutation of the nodes $0, \ldots, |L|-1$.
Nested block-diagonal matrix $Q = \mathrm{diag}(Q_{\sigma(0)}, Q_{\sigma(1)}, \ldots, Q_{\sigma(|L|-1)}, -1)$.
Constraint matrix A has the form

$$\left(\begin{array}{ccc|ccc|c}
A & & B_i & & & & 0 \\
& \ddots & \vdots & & & & \vdots \\
& A & B_i & & & & 0 \\
& & A & & & & B_i \\
\hline
& & & \ddots & & & \vdots \\
\hline
& & & A & & B_i & 0 \\
& & & & \ddots & \vdots & \vdots \\
& & & & A & B_i & 0 \\
& & & & & A & B_i \\
\hline
& & & & & & A \\
d_i \cdots d_i & 0 & \cdots & d_i \cdots d_i & 0 & & -1
\end{array}\right)$$

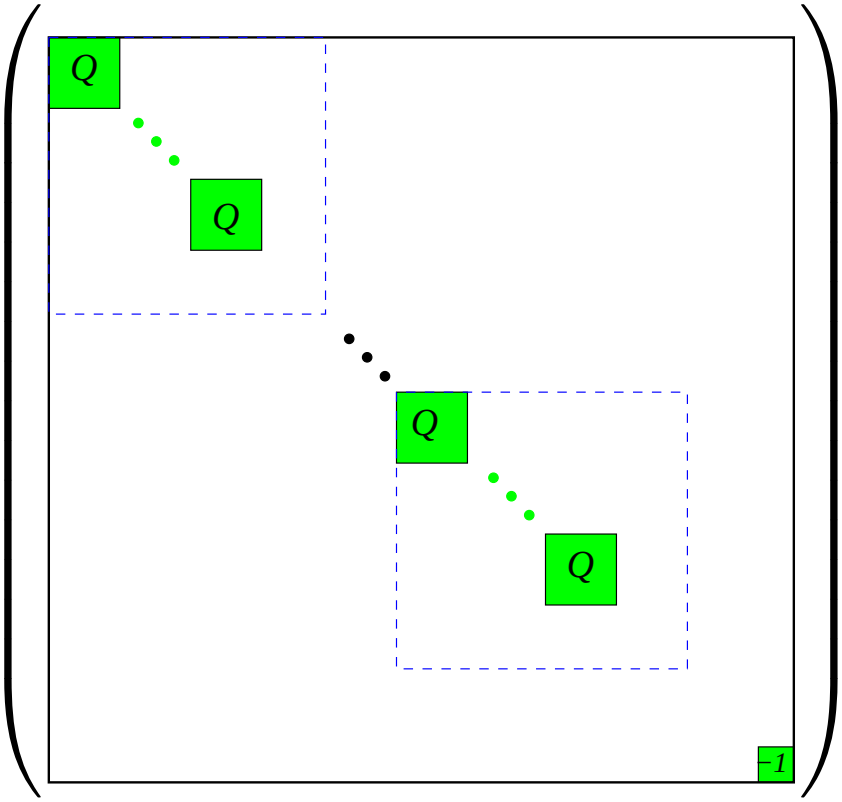


Variance representation



$$\mathrm{Var}(X) = I\!E(X - I\!E(X))^2$$

dense, convex QP

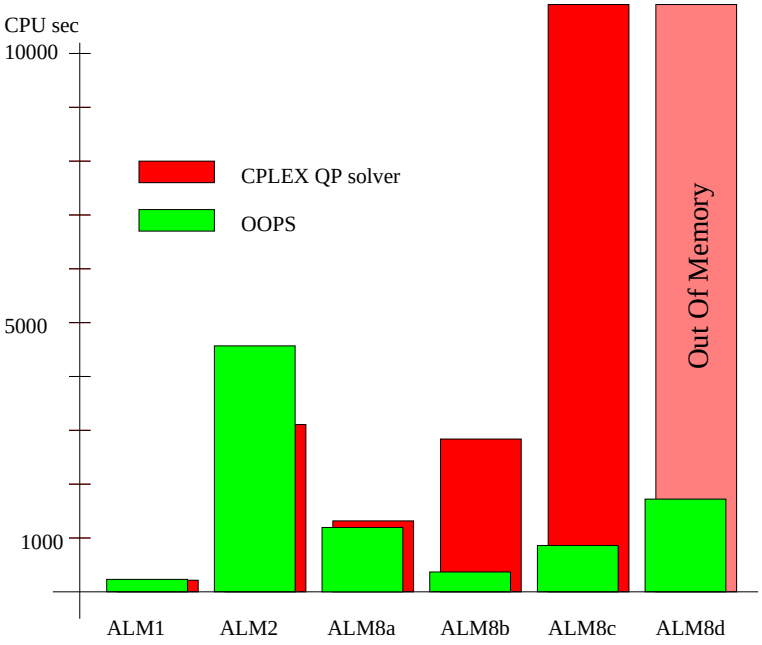


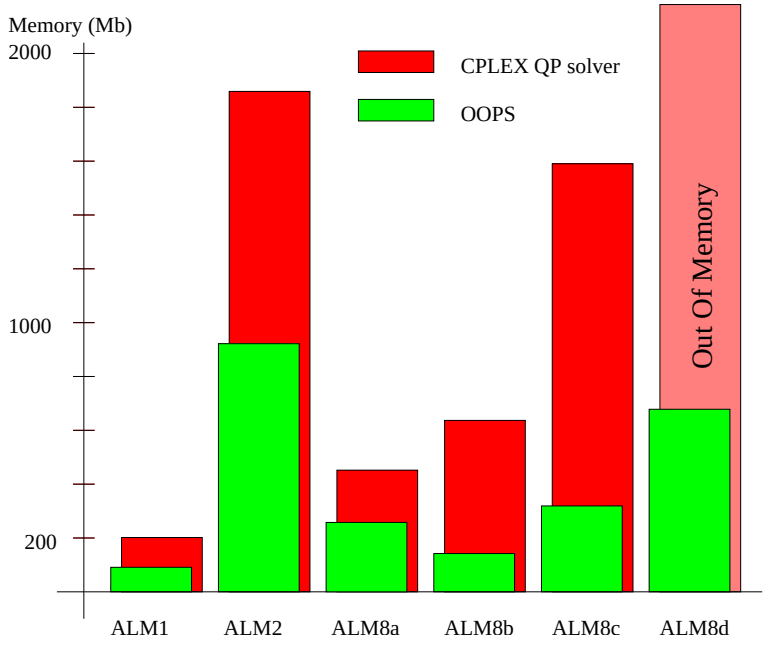
$$\mathrm{Var}(X) = I\!E(X^2) - (I\!E(X))^2$$

sparse, nonconvex QP

OOPS vs. CPLEX 7.0 (convexified QPs)

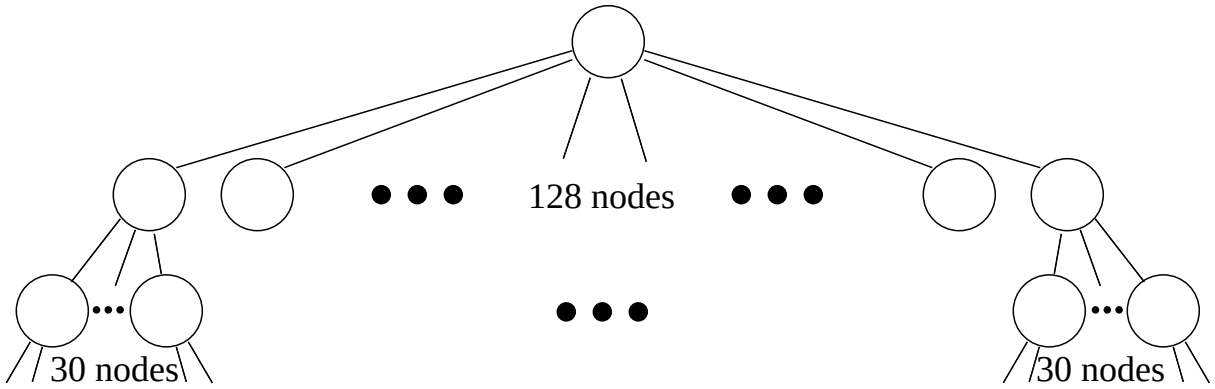
Pb	Stochastic Data			Dimensions		CPLEX 7.0			OOPS		
	Stgs	A	Nodes	Rows	Cols	time	iter	Mem	time	iter	Mem
2	6	5	111111	667K	1667K	3107	51	1859	4570	26	922
8a	4	50	1111	57K	167K	1317	29	452	1196	14	258
8b	3	50	1123	57K	168K	2838	31	637	368	16	142
8c	3	50	2552	130K	383K	10910	29	1590	860	16	319
8d	3	50	4971	254K	746K	51000*	30*	OoM	1723	17	678



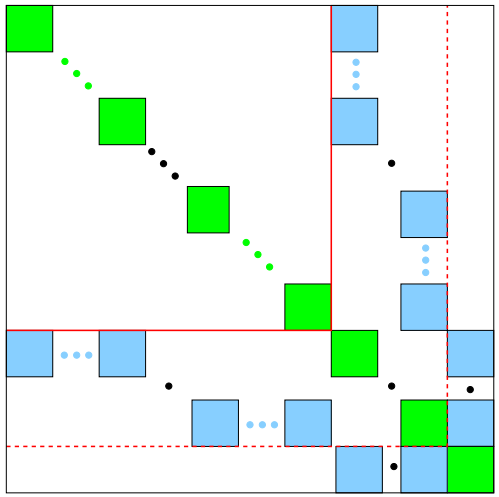
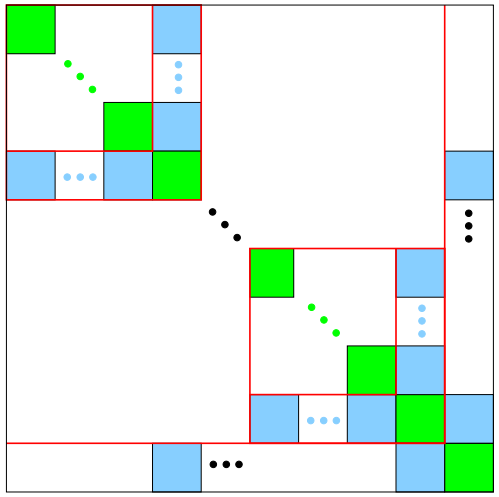


ALM: Largest Problem Attempted

- Optimization of 21 assets (stock market indices) 7 time stages.
- Using multistage stochastic programming
  Scenario tree geometry: 128-30-16-10-5-4 $\Rightarrow$ 16M scenarios.
- 3840 second level nodes with 350.000 variables each.
- Scenario Tree generated using geometric Brownian motion.
- $\Rightarrow$ 1.01 billion variables, 353 million constraints



Sparsity of Linear Algebra

-  $\Rightarrow$
  - $63 + 128 \times 63 = 8127$ columns for Schur-complement
  - Prohibitively expensive
-  $\Rightarrow$
  - Need facility to exploit nested structure
  - Need to be careful that Schur-complement calculations stay sparse on second level

Results (ALM: Mean-Variance QP formulation):

Prob	Stgs	Asts	Scen	Rows	Cols	iter	time	procs	machine
ALM8	7	6	13M	64M	154M	42	3923	512	BlueGene
ALM9	7	14	6M	96M	269M	39	4692	512	BlueGene
ALM10	7	13	12M	180M	500M	45	6089	1024	BlueGene
ALM11	7	21	16M	353M	1.011M	53	3020	1280	HPCx

The problem with

- **353 million of constraints**
- **1 billion of variables**

was solved in 50 minutes using 1280 procs.

Equation systems of dimension **1.363 billion**
were solved with the direct (implicit) factorization.

$\longrightarrow$ One IPM iteration takes less than a minute.

ALM: Extensions

Introduce two more (nonnegative) variables per final node (scenario) $i \in L_t$ to model the positive and negative variation from the mean

$$(1 - c_t) \sum_{j=1}^{J} v_j x_{i,j}^h + s_i^+ - s_i^- = y.$$

$(s_i^+)^2, (s_i^-)^2$ cannot both be positive hence the variance becomes:

$$\mathrm{Var}(X) = \sum_{i \in L_t} p_i (s_i^+ - s_i^-)^2 = \sum_{i \in L_t} p_i ((s_i^+)^2 + (s_i^-)^2).$$

We model **downside risk** using a semi-variance $I\!E[(X - I\!EX)_-^2]$

$$I\!E[(X - I\!EX)_-^2] = \sum_{i \in L_t} p_i (s_i^+)^2.$$

Downside risk can be taken into account

- in the objective, or
- as a constraint.

ALM Extensions

$x_i = (x_{i,1}, ..., x_{i,J})$ denotes the portfolio in node i

Standard Markowitz formulation:

$$\max \; y - \sigma \sum_{i \in L_t} p_i (d_i^\top x_i - y)^2 \quad \text{s.t.} \quad \begin{array}{rll} (C1) & \sum_{i \in L_t} p_i d_i^\top x_i - y & = 0 \\ (C2) & B x_{a(i)} - A x_i & = 0, \; i \neq 0 \\ (C3) & A x_0 & = b \end{array} \qquad \text{(QP)}$$

Downside risk constrained:

$$\max \quad y \qquad \text{s.t.} \quad \begin{array}{c} \sum_{i \in L_t} p_i (s_i^+)^2 \leq \rho \\ d_i^\top x_i + s_i^+ - s_i^- - y = 0, \quad i \in L_t \\ (C1) - (C3) \end{array} \qquad \text{(NLP)}$$

ALM Extensions

$x_i = (x_{i,1}, ..., x_{i,J})$ denotes the portfolio in node i

Nonlinear utility function:

$$\max \sum_{i \in L_t} p_i \log (d_i^\top x_i) \quad \text{s.t.} \quad \begin{array}{c} \sum_{i \in L_t} p_i (s_i^+)^2 \leq \rho \\ d_i^\top x_i + s_i^+ - s_i^- - y = 0, \quad i \in L_t \\ (C1) - (C3) \end{array}$$

(NLP)

Skewness formulation:

$$\max \; y + \gamma \sum_{i \in L_t} p_i (s_i^+ - s_i^-)^3 \quad \text{s.t.} \quad \begin{array}{c} \sum_{i \in L_t} p_i (s_i^+)^2 \leq \rho \\ d_i^\top x_i + s_i^+ - s_i^- - y = 0, \quad i \in L_t \\ (C1) - (C3) \end{array}$$

(NLP)

Results (NLP formulation): textbook SQP implemented

Semi-variance constrained → quadratically constrained problem.

Problem	Stages	Blocks	Assets	Total Nodes	constraints	variables
ALM1	5	10	5	11111	76.668	186.667
ALM2	6	10	5	111111	766.668	1.866.667
ALM4	5	24	5	346201	2.408.984	5.856.569
UNS1	5	35	5	360152	2.503.994	6.088.445

Problem	1 proc		2 procs		k procs		
	iter	time (s)	time (s)	speed-up	time (s)	speed-up	k
ALM1	36	218	107	2.04	44	4.95	5
ALM2	45	3456	1737	1.98	703	4.92	5
ALM4	67	11744	5902	1.98	1973	5.95	6
UNS1	42	14705	7949	1.85	3109	4.73	5

24 750MHz UltraSparc-III processors, 48GB of shared memory

Results (NLP formulation): textbook SQP, naive warmstart

Problem	Stages	Blocks	Assets	Total Nodes	constraints	variables
NLP4	3	70	40	4971	208.713	606.322
NLP5	4	24	25	14425	388.876	1.109.525
NLP6	4	40	50	65641	3.411.693	9.974.152
NLP7	4	55	20	169456	3.724.953	10.500.112

Semi-variance constrained (one quadratic constraint):

Problem	1 proc		2 procs		4 procs		8 procs	
	iter	t (s)	t (s)	speed-up	t (s)	speed-up	t (s)	speed-up
NLP4	35	568	258	2.20	141	4.02	92	6.11
NLP5	30	1073	516	2.08	254	4.21	148	7.27
NLP6	41	17764	9028	1.97	4545	3.91	2390	7.43
NLP7	43	18799	9391	2.00	4778	3.93	2459	7.64

Results (NLP formulation): textbook SQP, naive warmstart

Log utility function (nonlin objective, quadr constraint)

Problem	1 proc		2 procs		4 procs		8 procs	
	iter	t (s)	t (s)	speed-up	t (s)	speed-up	t (s)	speed-up
NLP4	25	448	214	2.09	110	4.07	72	6.22
NLP5	31	1287	618	2.08	306	4.20	179	7.19
NLP6	56	25578	13044	1.96	6650	3.85	3566	7.17
NLP7	60	24414	12480	1.96	6275	3.89	3338	7.31

Skewness formulation (nonlin objective, quadr constraint)

Problem	1 proc		2 procs		4 procs		8 procs	
	iter	t (s)	t (s)	speed-up	t (s)	speed-up	t (s)	speed-up
NLP4	50	820	390	2.10	208	3.94	130	6.31
NLP5	43	1466	715	2.05	396	3.70	207	7.08
NLP6	65	28678	14393	1.99	7144	4.01	3845	7.46
NLP7	62	23664	11963	1.98	6131	3.86	3097	7.64

OOPS vs. CPLEX 7.0 (solving convexified QPs)

Problem	Stages	Blocks	Assets	Total Nodes	constraints	variables
QP-NLP1	4	24	12	14425	201.351	546.950
QP-NLP2	3	60	20	3661	80.483	226.862
QP-NLP3	3	80	20	6481	142.503	401.662
QP-NLP4	3	70	40	4971	208.713	606.322

Problem	CPLEX 7.0			OOPS		
	time (s)	iter	mem (Mb)	time (s)	iter	mem (Mb)
QP-NLP1	615	28	369	406	14	325
QP-NLP2	955	16	340	159	15	136
QP-NLP3	2616	16	736	283	13	238
QP-NLP4	12236	18	1731	531	17	365

400MHz UltraSparc-II processor, 2GB of memory

Observations:

Interior Point Methods

→ are well-suited to Large Scale Optimization

Direct Methods

→ are well-suited to structure exploitation

Conclusion:

IPM implemented in OOPS

→ can solve very large financial planning problems

Object-Oriented Parallel IPM Solver (OOPS):

`http://www.maths.ed.ac.uk/~gondzio/parallel/solver.html`

- **G. and Sarkissian**, *Mathematical Programming* 96 (2003).
- **G. and Grothey**, *SIAM J. on Optmization* 13 (2003).
- **G. and Grothey**, *Lect. Notes in Comp. Sci.* 3911 (2006).
- **G. and Grothey**, *Annals of Oper. Res.* 152 (2007).
- **G. and Grothey**, *European J. of Oper. Res.* 181 (2007).

Papers available from: `http://www.maths.ed.ac.uk/~gondzio/`