School of Mathematics

# Interior Point Methods for Convex Quadratic Programming

**Jacek Gondzio**
Email: J.Gondzio@ed.ac.uk
URL: http://www.maths.ed.ac.uk/~gondzio



J. Gondzio IPMs for QP

## Outline

- **Part 1: IPM for QP**
  - quadratic forms
  - duality in QP
  - first order optimality conditions
  - primal-dual framework
- **Part 2: Linear Algebra in IPM**
  - LP case
  - QP case
  - Cholesky factorization
  - exploiting sparsity
- **Part 3: Huge Problems: Block-Sparsity**
- **Final Comments**



J. Gondzio IPMs for QP

**Part 1:**

## IPM for QP



J. Gondzio IPMs for QP

## Convex Quadratic Programs

The quadratic function

$$f(x) = x^T Q\, x$$

is convex if and only if the matrix $Q$ is positive definite.
In such case the quadratic programming problem

$$\begin{array}{ll} \min & c^T x + \frac{1}{2} x^T Q\, x \\ \text{s.t.} & Ax = b, \\ & x \geq 0, \end{array}$$

is well defined.

If there exists a *feasible* solution to it,
then there exists an *optimal* solution.

## QP Background:

**Def.** A matrix $Q \in \mathcal{R}^{n\times n}$ is positive semidefinite if $x^TQx \geq 0$ for any $x \neq 0$. We write $Q \succeq 0$.

**Def.** A matrix $Q \in \mathcal{R}^{n\times n}$ is positive definite if $x^TQx > 0$ for any $x \neq 0$. We write $Q \succ 0$.

## Example:

Consider quadratic functions $f(x) = x^T Q\, x$ with the following matrices:

$$Q_1 = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}, \quad Q_2 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad Q_3 = \begin{bmatrix} 5 & 4 \\ 4 & 3 \end{bmatrix}, \quad Q_4 = \begin{bmatrix} 5 & -2 \\ -2 & 3 \end{bmatrix}.$$

$Q_1$ and $Q_4$ are positive definite (hence $f_1, f_4$ are convex).
$Q_2$ and $Q_3$ are indefinite ($f_2, f_3$ are not convex).





The following 2 slides remind key facts from the duality theory applied to quadratic programming.





## Dual Quadratic Program

Consider a quadratic program

$$\begin{array}{ll} \min & c^T x + \frac{1}{2} x^T Q\, x \\ \text{s.t.} & Ax = b, \\ & x \geq 0, \end{array}$$

where $c, x \in \mathcal{R}^n, b \in \mathcal{R}^m$, $A \in \mathcal{R}^{m\times n}$, $Q \in \mathcal{R}^{n\times n}$.

We associate Lagrange multipliers $y \in \mathcal{R}^m$ and $s \in \mathcal{R}^n$ $(s \geq 0)$ with the constraints $Ax = b$ and $x \geq 0$, and write the **Lagrangian**

$$L(x, y, s) = c^T x + \frac{1}{2} x^T Q\, x - y^T (Ax - b) - s^T x.$$





## Dual QP (cont'd)

To determine the *Lagrangian dual*

$$L_D(y, s) = \min_{x \in X} L(x, y, s)$$

we need stationarity with respect to $x$:

$$\nabla_x L(x, y, s) = c + Qx - A^T y - s = 0.$$

Hence

$$\begin{aligned} L_D(y, s) &= c^T x + \tfrac{1}{2} x^T Q\, x - y^T (Ax - b) - s^T x \\ &= b^T y + x^T (c + Qx - A^T y - s) - \tfrac{1}{2} x^T Q\, x \\ &= b^T y - \tfrac{1}{2} x^T Q\, x, \end{aligned}$$

and the **dual** problem has the form:

$$\begin{array}{ll} \max & b^T y - \frac{1}{2} x^T Q\, x \\ \text{s.t.} & A^T y + s - Qx = c, \\ & x, s \geq 0, \end{array}$$

where $y \in \mathcal{R}^m$ and $x, s \in \mathcal{R}^n$.

## QP with IPMs

Consider the *convex* quadratic programming problem.
The **primal**

$$\begin{array}{ll} \min & c^T x + \frac{1}{2} x^T Q\, x \\ \text{s.t.} & Ax = b, \\ & x \geq 0, \end{array}$$

and the **dual**

$$\begin{array}{ll} \max & b^T y - \frac{1}{2} x^T Q\, x \\ \text{s.t.} & A^T y + s - Qx = c, \\ & x, s \geq 0. \end{array}$$

Apply the *usual* procedure:

- replace inequalities with log barriers;
- form the Lagrangian;
- write the first order optimality conditions;
- apply Newton method to them.





## QP with IPMs: Log Barriers

Replace the **primal** QP

$$\begin{array}{ll} \min & c^T x + \frac{1}{2} x^T Q\, x \\ \text{s.t.} & Ax = b, \\ & x \geq 0, \end{array}$$

with the **primal barrier QP**

$$\begin{array}{ll} \min & c^T x + \frac{1}{2} x^T Q\, x - \sum_{j=1}^{n} \ln x_j \\ \text{s.t.} & Ax = b. \end{array}$$





## QP with IPMs: Log Barriers

Replace the **dual** QP

$$\begin{array}{ll} \max & b^T y - \frac{1}{2} x^T Q\, x \\ \text{s.t.} & A^T y + s - Qx = c, \\ & y \text{ free}, \quad s \geq 0, \end{array}$$

with the **dual barrier QP**

$$\begin{array}{ll} \max & b^T y - \frac{1}{2} x^T Q\, x + \sum_{j=1}^{n} \ln s_j \\ \text{s.t.} & A^T y + s - Qx = c. \end{array}$$





## First Order Optimality Conditions

Consider the **primal barrier quadratic program**

$$\begin{array}{ll} \min & c^T x + \frac{1}{2} x^T Q\, x - \mu \sum_{j=1}^{n} \ln x_j \\ \text{s.t.} & Ax = b, \end{array}$$

where $\mu \geq 0$ is a barrier parameter.

Write out the **Lagrangian**

$$L(x, y, \mu) = c^T x + \frac{1}{2} x^T Q\, x - y^T (Ax - b) - \mu \sum_{j=1}^{n} \ln x_j,$$

## First Order Optimality Conditions (cont'd)

The conditions for a stationary point of the Lagrangian:

$$L(x,y,\mu) = c^T x + \frac{1}{2} x^T Q\, x - y^T (Ax - b) - \mu \sum_{j=1}^{n} \ln x_j,$$

are

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

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

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

The **First Order Optimality Conditions** are:

$$\begin{aligned} Ax \quad\quad\quad &= b, \\ A^T y + s - Qx &= c, \\ XSe \quad\quad\quad &= \mu e. \end{aligned}$$





## Apply Newton Method to the FOC

The first order optimality conditions for the barrier problem form a large system of nonlinear equations

$$F(x,y,s) = 0,$$

where $F : \mathcal{R}^{2n+m} \mapsto \mathcal{R}^{2n+m}$ is an application defined as follows:

$$F(x,y,s) = \begin{bmatrix} Ax - b \\ A^T y + s - Qx - c \\ XSe - \mu e \end{bmatrix}.$$

Actually, the first two terms of it are *linear*; only the last one, corresponding to the complementarity condition, is *nonlinear*.
Note that

$$\nabla F(x,y,s) = \begin{bmatrix} A & 0 & 0 \\ -Q & A^T & I \\ S & 0 & X \end{bmatrix}.$$





## Newton Method for the FOC (cont'd)

Thus, for a given point $(x,y,s)$
we find the Newton direction $(\Delta x, \Delta y, \Delta s)$
by solving the system of linear equations:

$$\begin{bmatrix} A & 0 & 0 \\ -Q & A^T & I \\ S & 0 & X \end{bmatrix} \cdot \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} b - Ax \\ c - A^T y - s + Qx \\ \mu e - XSe \end{bmatrix}.$$





## Interior-Point QP Algorithm

*Initialize*

$$k = 0, \quad (x^0, y^0, s^0) \in \mathcal{F}^0, \quad \mu_0 = \tfrac{1}{n} \cdot (x^0)^T s^0, \quad \alpha_0 = 0.9995$$

*Repeat until optimality*

$k = k + 1$
$\mu_k = \sigma \mu_{k-1}$, where $\sigma \in (0,1)$
$\Delta =$ Newton direction towards $\mu$-center

*Ratio test:*

$$\begin{aligned} \alpha_P &:= \max\ \{\alpha > 0 :\ x + \alpha \Delta x \geq 0\}, \\ \alpha_D &:= \max\ \{\alpha > 0 :\ s + \alpha \Delta s \geq 0\}. \end{aligned}$$

*Make step:*

$$\begin{aligned} x^{k+1} &= x^k + \alpha_0 \alpha_P \Delta x, \\ y^{k+1} &= y^k + \alpha_0 \alpha_D \Delta y, \\ s^{k+1} &= s^k + \alpha_0 \alpha_D \Delta s. \end{aligned}$$

## From LP to QP

QP problem

$$\begin{array}{ll} \min & c^T x + \frac{1}{2} x^T Q x \\ \text{s.t.} & Ax = b, \\ & x \geq 0. \end{array}$$

First order conditions (for barrier problem)

$$\begin{aligned} Ax &= b, \\ A^T y + s - Qx &= c, \\ XSe &= \mu e. \end{aligned}$$





**Part 2:**

## Linear Algebra in IPM





## Linear Algebra of IPM: LP Case

### FOC

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

### Newton direction

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} \xi_p \\ \xi_d \\ \xi_\mu \end{bmatrix},$$

where

$$\begin{bmatrix} \xi_p \\ \xi_d \\ \xi_\mu \end{bmatrix} = \begin{bmatrix} b - Ax \\ c - A^T y - s \\ \mu e - XSe \end{bmatrix}.$$





## Linear Algebra, LP Case (cont'd)

In **Newton direction**

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} \xi_p \\ \xi_d \\ \xi_\mu \end{bmatrix},$$

use the third equation to eliminate

$$\Delta s = X^{-1}(\xi_\mu - S\Delta x) = -X^{-1}S\Delta x + X^{-1}\xi_\mu,$$

from the second equation and get

$$\begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} \xi_d - X^{-1}\xi_\mu \\ \xi_p \end{bmatrix}.$$

where $\Theta = XS^{-1}$ is a diagonal scaling matrix.

# Linear Algebra of IPM: QP Case

**FOC**

$$\begin{aligned} Ax &= b, \\ A^T y + s - Qx &= c, \\ XSe &= \mu e. \end{aligned}$$

**Newton direction**

$$\begin{bmatrix} A & 0 & 0 \\ -Q & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} \xi_p \\ \xi_d \\ \xi_\mu \end{bmatrix},$$

where

$$\begin{aligned} \xi_p &= b - Ax, \\ \xi_d &= c - A^T y - s + Qx, \\ \xi_\mu &= \mu e - XSe. \end{aligned}$$





# Linear Algebra, QP Case (cont'd)

In **Newton direction**

$$\begin{bmatrix} A & 0 & 0 \\ -Q & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} \xi_p \\ \xi_d \\ \xi_\mu \end{bmatrix},$$

use the third equation to eliminate

$$\Delta s = X^{-1}(\xi_\mu - S\Delta x) = -X^{-1}S\Delta x + X^{-1}\xi_\mu,$$

from the second equation and get

$$\begin{bmatrix} -Q - \Theta^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} \xi_d - X^{-1}\xi_\mu \\ \xi_p \end{bmatrix}.$$

where $\Theta = XS^{-1}$ is a diagonal scaling matrix.





# Summary: From LP to QP

Newton direction

$$\begin{bmatrix} A & 0 & 0 \\ -Q & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} \xi_p \\ \xi_d \\ \xi_\mu \end{bmatrix},$$

where

$$\begin{aligned} \xi_p &= b - Ax, \\ \xi_d &= c - A^T y - s + Qx, \\ \xi_\mu &= \mu e - XSe. \end{aligned}$$

**Augmented system**

$$\begin{bmatrix} -Q - \Theta^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} \xi_d - X^{-1}\xi_\mu \\ \xi_p \end{bmatrix}.$$

**Conclusion:**
QP is a natural extension of LP.





# IPMs: LP vs QP

Augmented system in **LP**

$$\begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} \xi_d - X^{-1}\xi_\mu \\ \xi_p \end{bmatrix}.$$

Eliminate $\Delta x$ from the first equation and get normal equations

$$(A\Theta A^T)\Delta y = g.$$

# IPMs: LP vs QP

Augmented system in **QP**

$$\begin{bmatrix} -Q - \Theta^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} \xi_d - X^{-1}\xi_\mu \\ \xi_p \end{bmatrix}.$$

Eliminate $\Delta x$ from the first equation and get normal equations

$$(A(Q + \Theta^{-1})^{-1} A^T)\Delta y \;=\; g.$$

One can use normal equations in LP, but not in QP. Normal equations in QP may become almost completely dense even for sparse matrices $A$ and $Q$. Thus, in QP, usually the indefinite augmented system form is used.





# Normal Equations

$$(A\Theta A^T)\Delta y \;=\; g.$$

Matrix $A\Theta A^T$ has always the same sparsity structure (only $\Theta$ changes in subsequent iterations).

**Two step solution method:**

- factorization to $LDL^T$ form,
- backsolve to compute direction $\Delta y$.





# Cholesky factorization

Compute a decomposition

$$LDL^T = A\Theta A^T.$$

where:
$L$ is a lower triangular matrix; and
$D$ is a diagonal matrix.

Cholesky factorization is simply the **Gaussian Elimination** process that exploits two properties of the matrix:

- symmetry;
- positive definiteness.





# Use of Cholesky factorization

Replace the **difficult** equation

$$(A\Theta A^T) \cdot \Delta y = g,$$

with a sequence of **easy** equations:

$$\begin{aligned} L \cdot u &= g, \\ D \cdot v &= u, \\ L^T \cdot \Delta y &= v. \end{aligned}$$

Note that

$$\begin{aligned} g &= Lu \\ &= L(Dv) \\ &= LD(L^T \Delta y) \\ &= (LDL^T)\Delta y \\ &= (A\Theta A^T)\Delta y. \end{aligned}$$

## Symmetric Gaussian Elimination

Let $H \in \mathcal{R}^{m \times m}$ be a symmetric positive definite matrix

$$H = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1m} \\ h_{21} & h_{22} & \cdots & h_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ h_{m1} & h_{m2} & \cdots & h_{mm} \end{bmatrix}.$$

By applying Gaussian Elimination to it, we can represent it in the following form:

$$\begin{bmatrix} 1 & 0 & \cdots & 0 \\ l_{21} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{m1} & l_{m2} & \cdots & 1 \end{bmatrix} \begin{bmatrix} d_{11} & 0 & \cdots & 0 \\ 0 & d_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_{mm} \end{bmatrix} \begin{bmatrix} 1 & l_{21} & \cdots & l_{m1} \\ 0 & 1 & \cdots & l_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}.$$





## Symmetric Gaussian Elimination

**Example 1:**

$$\begin{bmatrix} 1 & -1 & 2 \\ -1 & 3 & 0 \\ 2 & 0 & 9 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 2 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 1 & -1 & 2 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

**Example 2:**

$$\begin{bmatrix} 1 & 1 & -1 \\ 1 & 5 & 7 \\ -1 & 7 & 22 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -1 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 5 \end{bmatrix} \begin{bmatrix} 1 & 1 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}.$$





## Existence of $LDL^T$ factorization

**Lemma 2**: The decomposition $H = LDL^T$ with $d_{ii} > 0, \forall i$ exists iff $H$ is positive definite (PD).

**Proof:**
Part 1 ( $\Rightarrow$ )
Let $H = LDL^T$ with $d_{ii} > 0$. Take any $x \neq 0$ and let $u = L^T x$. Since $L$ is a unit lower triangular matrix it is nonsingular so $u \neq 0$ and

$$x^T H x = x^T L D L^T x = u^T D u = \sum_{i=1}^{m} d_{ii} u_i^2 > 0.$$





**Proof (cont'd):**
Part 2 ( $\Leftarrow$ )
Proof by induction on dimension of $H$.
For $m = 1$. $H = h_{11} = d_{11} > 0$ iff $H$ is PD.
Assume the result is true for $m = k - 1 \geq 1$.
Let $H = \begin{bmatrix} W & a \\ a^T & q \end{bmatrix} \in \mathcal{R}^{k \times k}$ be given $k \times k$ positive definite matrix with $W \in \mathcal{R}^{(k-1)\times(k-1)}$, $a \in \mathcal{R}^{k-1}$ and $q \in \mathcal{R}$. Note first that since $H$ is PD, $W$ is also PD. Indeed for any $(x, 0) \in \mathcal{R}^k$ we have

$$[x, 0] \begin{bmatrix} W & a \\ a^T & q \end{bmatrix} \begin{bmatrix} x \\ 0 \end{bmatrix} = x^T W x > 0 \quad \forall x \in \mathcal{R}^{k-1}, x \neq 0.$$

From inductive hypothesis we know that $W = LDL^T$ with $d_{ii} > 0$. Let

$$\begin{bmatrix} W & a \\ a^T & q \end{bmatrix} = \begin{bmatrix} L & 0 \\ l^T & 1 \end{bmatrix} \begin{bmatrix} D & 0 \\ 0 & d \end{bmatrix} \begin{bmatrix} L^T & l \\ 0 & 1 \end{bmatrix},$$

where $l$ is the solution of equation $(LD)l = a$ (it is well defined since $L$ and $D$ are nonsingular) and $d$ is given by $d = q - l^T Dl$.

Hence matrix $H = \begin{bmatrix} W & a \\ a^T & q \end{bmatrix}$ has an $\bar{L}\bar{D}\bar{L}^T$ decomposition.

It remains to prove that $d > 0$. Consider the vector

$$x = \begin{bmatrix} -L^{-T}l \\ 1 \end{bmatrix}.$$

Since $H$ is positive definite, we get

$$\begin{aligned} 0 &< x^T Hx \\ &= [-l^T L^{-1}, 1] \begin{bmatrix} L & 0 \\ l^T & 1 \end{bmatrix} \begin{bmatrix} D & 0 \\ 0 & d \end{bmatrix} \begin{bmatrix} L^T & l \\ 0 & 1 \end{bmatrix} \begin{bmatrix} -L^{-T}l \\ 1 \end{bmatrix} \\ &= [0, 1] \begin{bmatrix} D & 0 \\ 0 & d \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = d, \end{aligned}$$

which completes the proof.





## Large Problems are Sparse

Suppose a medium or large LP is solved: $m, n \sim 10^3 - 10^6$.
Can all variables be linked at the same time?
No, usually only a subset of them is linked.

There are usually only *several* nonzeros per row in an LP.
Large problems are always **sparse**.
Very large problems are often **block-sparse**.

Exploiting sparsity in computations leads to huge savings.
Exploiting sparsity means mainly avoiding doing useless computations: the computations for which the result is known, as for example multiplications with zero.





## Exploiting sparsity: Example

$$Ax = \begin{bmatrix} 2 & 1 & 0 & 4 & 0 & 0 \\ 0 & 2 & 0 & -1 & 5 & -1 \\ 3 & 0 & 3 & 8 & 0 & 5 \end{bmatrix} \begin{bmatrix} 2 \\ 0 \\ 5 \\ 0 \\ 0 \\ -2 \end{bmatrix}.$$

It requires computing

$$2 \cdot A_{.1} + 5 \cdot A_{.3} - 2 \cdot A_{.6}$$

and involves only five multiplications and five additions.
We say that this matrix-vector multiplication needs 5 flops.

A **flop** is a *floating point operation*:

$$x := x + a \cdot b.$$





## Exploiting Sparsity in Cholesky Factorization

### Matrix H and its Cholesky Factor

$$H = \begin{bmatrix} \mathbf{p} & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ \mathbf{x} & x & & \\ \mathbf{x} & & x & \\ \mathbf{x} & & & x \end{bmatrix} \Rightarrow L = \begin{bmatrix} x & & & \\ x & x & & \\ x & x & x & \\ x & x & x & x \end{bmatrix}$$

### Reordered Matrix H and its Cholesky Factor

$$PHP^T = \begin{bmatrix} x & & & x \\ & x & & x \\ & & x & x \\ x & x & x & x \end{bmatrix} \Rightarrow L = \begin{bmatrix} x & & & \\ & x & & \\ & & x & \\ x & x & x & x \end{bmatrix}$$

# Minimum Degree Ordering

**Sparse Matrix** **Pivot $h_{11}$** **Pivot $h_{22}$**

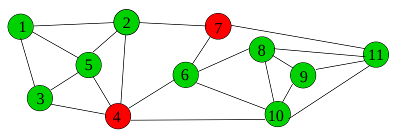
$$H = \begin{bmatrix} x & & x & x & x & \\ & x & & & x & \\ x & & x & & & x \\ x & & & x & & x \\ x & x & & & x & \\ & & x & x & & x \end{bmatrix} \quad \begin{bmatrix} \mathbf{p} & & \mathbf{x} & \mathbf{x} & \mathbf{x} & \\ & x & & & x & \\ \mathbf{x} & & x & \mathbf{f} & \mathbf{f} & x \\ \mathbf{x} & & \mathbf{f} & x & \mathbf{f} & x \\ \mathbf{x} & x & \mathbf{f} & \mathbf{f} & x & \\ & & x & x & & x \end{bmatrix} \quad \begin{bmatrix} x & & x & x & x & \\ & \mathbf{p} & & & \mathbf{x} & \\ x & & x & & & x \\ x & & & x & & x \\ x & \mathbf{x} & & & x & \\ & & x & x & & x \end{bmatrix}$$

**Minimum degree ordering**:
choose a diagonal element corresponding to a row with the *minimum* number of nonzeros.
Permute rows and columns of $H$ accordingly.





# Nested Dissection:




**Original Matrix**

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | $x$ | $x$ | $x$ | | $x$ | | | | | | |
| 2 | $x$ | $x$ | | $x$ | $x$ | | $x$ | | | | |
| 3 | $x$ | | $x$ | $x$ | $x$ | | | | | | |
| 4 | | $x$ | $x$ | $x$ | $x$ | $x$ | | | | $x$ | |
| 5 | $x$ | $x$ | $x$ | $x$ | $x$ | | | | | | |
| 6 | | | | $x$ | | $x$ | $x$ | $x$ | | $x$ | |
| 7 | | $x$ | | | | $x$ | $x$ | | | | $x$ |
| 8 | | | | | | $x$ | | $x$ | $x$ | $x$ | $x$ |
| 9 | | | | | | | | $x$ | $x$ | $x$ | $x$ |
| 10 | | | | $x$ | | $x$ | | $x$ | $x$ | $x$ | $x$ |
| 11 | | | | | | | $x$ | $x$ | $x$ | $x$ | $x$ |

**Reordered Matrix**

| | 1 | 2 | 3 | 5 | 6 | 8 | 9 | 10 | 11 | 4 | 7 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | $x$ | $x$ | $x$ | $x$ | | | | | | | |
| 2 | $x$ | $x$ | | $x$ | | | | | | **x** | **x** |
| 3 | $x$ | | $x$ | $x$ | | | | | | **x** | |
| 5 | $x$ | $x$ | $x$ | $x$ | | | | | | **x** | |
| 6 | | | | | $x$ | $x$ | | $x$ | | **x** | **x** |
| 8 | | | | | $x$ | $x$ | $x$ | $x$ | $x$ | | |
| 9 | | | | | | $x$ | $x$ | $x$ | $x$ | | |
| 10 | | | | | $x$ | $x$ | $x$ | $x$ | $x$ | **x** | |
| 11 | | | | | | $x$ | $x$ | $x$ | $x$ | | **x** |
| 4 | | **x** | **x** | **x** | **x** | | | **x** | | **x** | |
| 7 | | **x** | | | **x** | | | | **x** | | **x** |





# Cholesky factorization

$$LDL^T = A\Theta A^T.$$

Involved preparation step:

- minimum degree ordering (reduces # of nonzeros of $L$);
- symbolic factorization (predicts the sparsity structure of $L$).

Computational complexity of different steps:

- minimum degree ordering $\mathcal{O}(\sum_i n_i^2)$
- numerical factorization $\mathcal{O}(\sum_i n_i^2)$
- symbolic factorization $\mathcal{O}(\sum_i n_i)$
- backsolve $\mathcal{O}(\sum_i n_i)$

where $n_i$ is # of nonzero entries in $L_{.i}$





# Linear Algebra: Simplex Method vs IPM

Suppose an LP of dimension $m \times n$ is solved.

**Iterations to reach an optimum:**

| Simplex Method | | IPM | |
|---|---|---|---|
| Theory | Practice | Theory | Practice |
| Nonpolynomial | $O(m+n)$ | $O(\sqrt{n})$ | $O(\log_{10} n)$ |

But one iteration of the simplex method is usually significantly less expensive. Simplex method solves equation with the basis matrix:

$$\begin{bmatrix} B & N \\ 0 & I_{n-m} \end{bmatrix} \begin{bmatrix} x_B \\ x_N \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix},$$

which reduces to

$$Bx_B = b.$$

IPM solves equation with the matrix $A\Theta A^T$:

$$(A\Theta A^T)\Delta y = g.$$

## Implementation of IPMs

**Andersen, Gondzio, Mészáros and Xu**
Implementation of IPMs for large scale LP,
in: *Interior Point Methods in Mathematical Programming*,
T. Terlaky (ed.), Kluwer Academic Publishers, 1996, pp. 189–252.

## Recent Survey on IPMs (easy reading)

**Gondzio**
Interior point methods 25 years later,
*European J. of Operational Research* 218 (2012) 587–601.
`http://www.maths.ed.ac.uk/~gondzio/reports/ipmXXV.html`





**Part 3:**

## Huge Problems: Block-Sparsity





## Structured Problems

## Observation:

**Truly large scale problems are not only sparse...**
**→ such problems are structured**

### Structure is displayed in:

- Jacobian matrix $A$
- Hessian matrix $Q$

### Structure can be exploited in:

- IPM Algorithm
- Linear Algebra of IPM ⟶ (focus of the rest of this lecture)



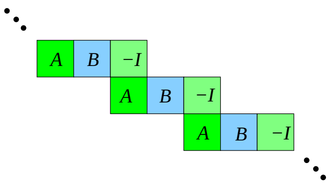


## Structured Problems

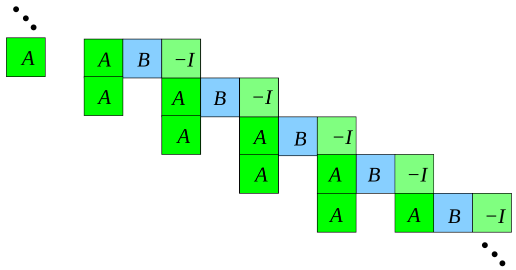
**... are present everywhere.**

# Sources of Structure

**Dynamics → Staircase structure**




$x_{t+1} = A_t x_t + B_t u_t$



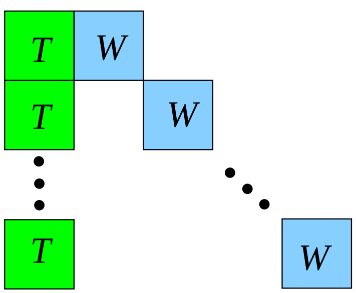

$x_{t+1} = A_t^{t+1} x_t + \ldots + A_{t-p}^{t+1} x_{t-p} + B_t u_t$



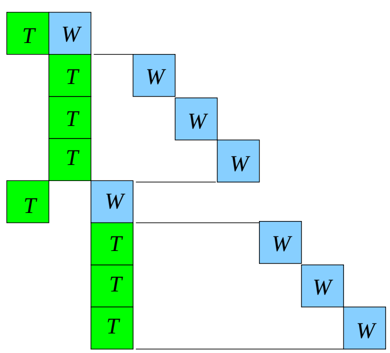


# Sources of Structure

**Uncertainty → Block-angular structure**




$T_i x^1 + W_i y_i = b_i$




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

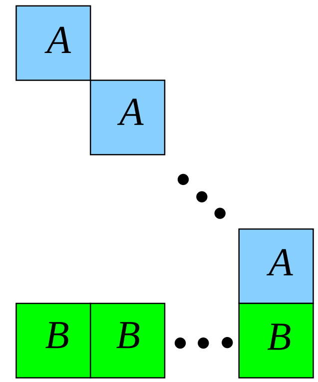




# Sources of Structure

**Common resource constraint**

$\sum_{i=1}^{k} B_i x_i = b$ → **Dantzig-Wolfe structure**



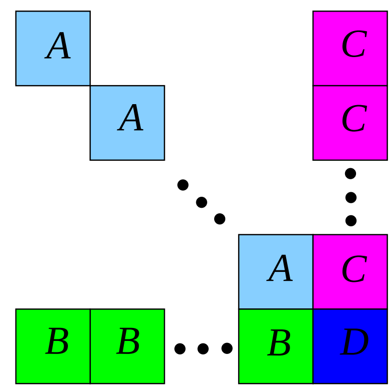





# Sources of Structure

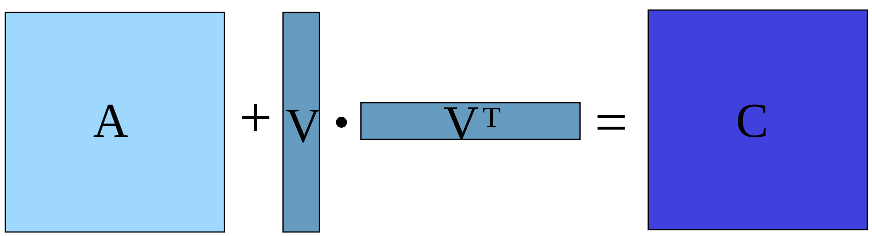
Other types of **near-separability**

→ **Row and column bordered block-diagonal structure**



## Sources of Structure

(low) **rank-corrector**
$A + VV^T = C$




**and networks, ODE- or PDE-discretizations, etc.**



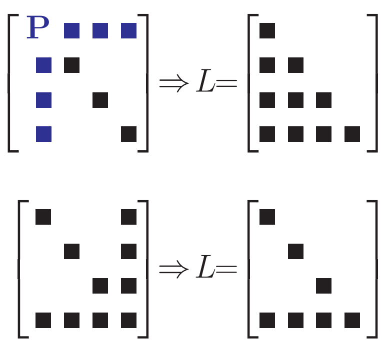


## From Sparsity to Block-Sparsity:

**Sparse Matrix** **Block-Sparse Matrix**

$$H = \begin{bmatrix} \mathbf{p} & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ \mathbf{x} & x & & \\ \mathbf{x} & & x & \\ \mathbf{x} & & & x \end{bmatrix} \Rightarrow L = \begin{bmatrix} x & & & \\ x & x & & \\ x & x & x & \\ x & x & x & x \end{bmatrix}$$

$$PHP^T = \begin{bmatrix} x & & & x \\ & x & & x \\ & & x & x \\ x & x & x & x \end{bmatrix} \Rightarrow L = \begin{bmatrix} x & & & \\ & x & & \\ & & x & \\ x & x & x & x \end{bmatrix}$$



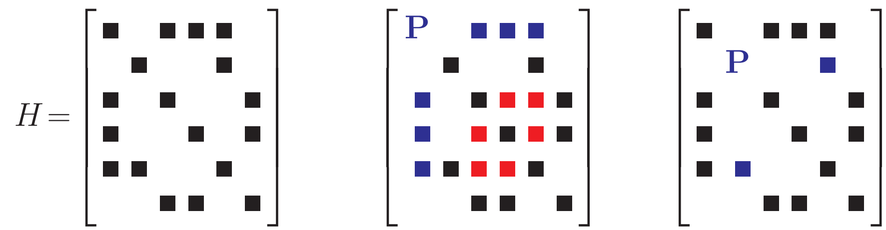





## From Sparsity to Block-Sparsity:

Apply minimum degree ordering to **(sparse) blocks**:

**Block-Sparse Matrix** **Pivot Block** $H_{11}$ **Pivot Block** $H_{22}$




Choose a diagonal block-pivot corresponding to a block-row with the *minimum* number of blocks.
Permute block-rows and block-columns of $H$ accordingly.





## Abstract Linear Algebra for IPMs

**Execute the operation**

**"solve (reduced) KKT system"**

**in IPMs for LP, QP and NLP.**

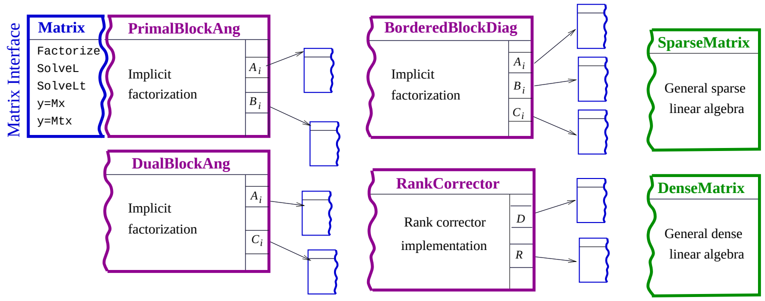
**It works like the "backslash" operator in MATLAB.**

## Assumptions:

**Q and A are block-structured**

## 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




⇒ Rebuild *block elimination tree* with matrix interface structures





## Example: Financial Planning Problems (ALM)

- 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

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

⇒ Stochastic Program: ⇒ formulate deterministic equivalent

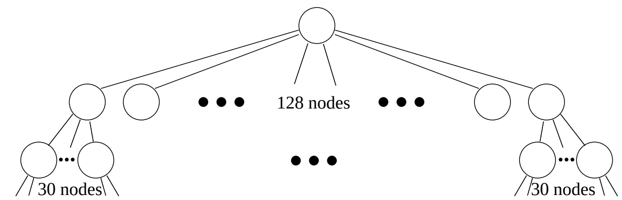
- standard QP, but huge
- extentions: *nonlinear risk measures* (log utility, skewness)





## ALM: Largest Problem Attempted

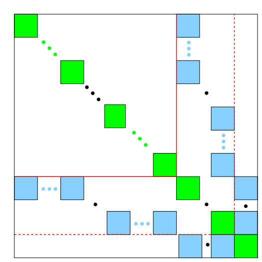
- Optimization of 21 assets (stock market indices) 7 time stages.
- Using multistage stochastic programming
  Scenario tree geometry: 128-30-16-10-5-4 ⇒ 16M scenarios.
- 3840 second level nodes with 350.000 variables each.
- Scenario Tree generated using geometric Brownian motion.
- ⇒ 1.01 billion variables,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



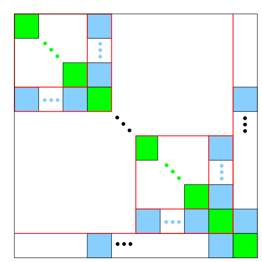





## Sparsity of Linear Algebra

- 

  ⇒
  - $63 + 128 \times 63 = 8127$ columns for Schur-complement
  - Prohibitively expensive

- 

  ⇒
  - 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 QP problem with

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

was solved in 50 minutes using 1280 procs (*May 2005*).

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

⟶ One IPM iteration takes less than a minute.





## References


- **Gondzio and Sarkissian**, Parallel interior point solver for structured linear programs, *Math Prog* 96 (2003) 561-584.
- **Gondzio and Grothey**, Parallel IPM solver for structured QPs: application to financial planning problems, *Annals of Operations Research* 152 (2007) 319-339.
- **Woodsend and Gondzio**, Exploiting separability in large scale linear support vector machine training, *Comput Optimization and Appls* 49 (2011) 241-269.
- **K. Fountoulakis, J. Gondzio and P. Zhlobich**, Matrix-free interior point method for compressed sensing problems, *Math Prog Computation* 6 (2014), pp. 1-31.


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

## OOPS: Object-Oriented Parallel Solver

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





## Interior Point Methods:

- Unified view of optimization
  → from LP via QP to NLP
- Predictable behaviour
  → small number of iterations
- Unequalled efficiency
  - competitive for small problems ($n \le 10^6$)
  - beyond competition for large problems ($n \ge 10^6$)

# Use IPMs in your research!