Ed Forum

• 2nd Midterm: April 18
  Time: 11:00am — 12:20pm
  Students with extensions please reach out to me
  Location: Same room as lecture
  Topics: linear optimization
  Exercises to prepare: past midterm + extra exercises on canvas

• Questions

  • How are tau, sigma, and mu related?

  • I was still a little confused by SY1. Why do we need to include it in the matrix?
Recap
(Sparse) Cholesky factorization

Every positive definite matrix $A$ can be factored as

$$A = PLL^T P^T \quad \rightarrow \quad P^T AP = LL^T$$

$P$ permutation,  $L$ lower triangular
(Sparse) Cholesky factorization

Every positive definite matrix $A$ can be factored as

$$A = PLL^TP^T \quad \rightarrow \quad P^TAP = LL^T$$

$P$ permutation, $L$ lower triangular

**Permutations**

- Reorder rows/cols of $A$ with $P$ to (heuristically) get sparser $L$
- $P$ depends only on sparsity pattern of $A$ (unlike $LU$ factorization)
- If $A$ is dense, we can set $P = I$
(Sparse) Cholesky factorization

Every positive definite matrix $A$ can be factored as

$$A = PLL^T P^T \quad \rightarrow \quad P^T AP = LL^T$$

$P$ permutation, $L$ lower triangular

**Permutations**

- Reorder rows/cols of $A$ with $P$ to (heuristically) get **sparser** $L$
- $P$ depends only on sparsity pattern of $A$ (unlike $LU$ factorization)
- If $A$ is dense, we can set $P = I$

**Cost**

- If $A$ dense, typically $O(n^3)$ but usually much less
- It depends on the number of nonzeros in $A$, sparsity pattern, etc.
- Typically 50% faster than $LU$ (need to find only one matrix)
Linear optimization as a root finding problem

Optimality conditions

minimize \[ c^T x \]
subject to \[ Ax \leq b \]
Linear optimization as a root finding problem

Optimality conditions

Primal

\[
\begin{align*}
\text{minimize} & \quad c^T x \\
\text{subject to} & \quad Ax \leq b
\end{align*}
\]

Dual

\[
\begin{align*}
\text{maximize} & \quad -b^T y \\
\text{subject to} & \quad A^T y + c = 0 \\
& \quad y \geq 0
\end{align*}
\]
**Linear optimization as a root finding problem**

**Optimality conditions**

minimize \( c^T x \)  
subject to \( Ax \leq b \)

\[ \text{Primal} \]
\[ \begin{align*}
    \text{minimize} & \quad c^T x \\
    \text{subject to} & \quad Ax + s = b \\
    & \quad s \geq 0
\end{align*} \]

\[ \text{Dual} \]
\[ \begin{align*}
    \text{maximize} & \quad -b^T y \\
    \text{subject to} & \quad A^T y + c = 0 \\
    & \quad y \geq 0
\end{align*} \]

**KKT conditions**

1. **Primal feasibility**
\[ Ax + s - b = 0 \]

2. **Dual feasibility**
\[ A^T y + c = 0 \]

3. **Complementary slackness**
\[ s_i y_i = 0, \quad i = 1, \ldots, m \]

\[ s \cdot y \geq 0 \]
Linear optimization as a root finding problem

\[ Ax + s - b = 0 \]
\[ A^T y + c = 0 \]
\[ s_i y_i = 0, \quad i = 1, \ldots, m \]
\[ s, y \geq 0 \]
Linear optimization as a root finding problem

\[ Ax + s - b = 0 \]
\[ A^T y + c = 0 \]
\[ s_i y_i = 0, \quad i = 1, \ldots, m \]
\[ s, y \geq 0 \]

Diagonalize complementary slackness

\[ S = \text{diag}(s) = \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_m \end{bmatrix} \]
\[ Y = \text{diag}(y) = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \]
\[ SY_1 = \text{diag}(s)\text{diag}(y)1 = \begin{bmatrix} s_1 y_1 \\ s_2 y_2 \\ \vdots \\ s_m y_m \end{bmatrix}1 = \begin{bmatrix} s_1 y_1 \\ s_2 y_2 \\ \vdots \\ s_m y_m \end{bmatrix} \]
Linear optimization as a root finding problem

\[
Ax + s - b = 0 \\
A^Ty + c = 0 \\
s_iy_i = 0, \quad i = 1, \ldots, m \\
s, y \geq 0
\]

Diagonalize complementary slackness

\[
S = \text{diag}(s) = \begin{bmatrix} s_1 & \cdots & s_m \end{bmatrix} \\
Y = \text{diag}(y) = \begin{bmatrix} y_1 & \cdots & y_m \end{bmatrix}
\]

\[
SY_1 = \text{diag}(s)\text{diag}(y)1 = \begin{bmatrix} s_1y_1 & \cdots & s_my_m \end{bmatrix}1 = \begin{bmatrix} s_1y_1 \\ s_2y_2 \\ \vdots \\ s_my_m \end{bmatrix}
\]

\[
s_iy_i = 0, \quad i = 1, \ldots, m \iff SY_1 = 0
\]
Main idea

Optimality conditions

\[
\begin{align*}
    h(y, x, s) &= \begin{bmatrix} Ax + s - b \\ A^T y + c \\ S \cdot Y \cdot 1 \end{bmatrix} = \begin{bmatrix} r_p \\ r_d \\ S \cdot Y \cdot 1 \end{bmatrix} = 0 \\
    S &= \text{diag}(s) \\
    Y &= \text{diag}(y) \\
    s, y &\geq 0
\end{align*}
\]

- Apply variants of Newton’s method to solve \( h(x, s, y) = 0 \)
- Enforce \( s, y > 0 \) (strictly) at every iteration
- **Motivation** avoid getting stuck in “corners”
Smoothed optimality conditions

Optimality conditions

\[ Ax + s - b = 0 \]
\[ A^T y + c = 0 \]
\[ s_i y_i = \tau \]
\[ s, y \geq 0 \]

Same optimality conditions for a “smoothed” version of our problem
Smoothed optimality conditions

Optimality conditions

\[
Ax + s - b = 0 \\
A^T y + c = 0 \\
s_i y_i = \tau \\
s, y \geq 0
\]

Same optimality conditions for a “smoothed” version of our problem

Duality gap

\[
\ln \gamma = s^T y = (b - Ax)^T y = b^T x - x^T A^T y = b^T y + c^T x
\]
Central path

minimize \quad c^T x - \tau \sum_{i=1}^{m} \log(s_i)

subject to \quad Ax + s = b

Set of points \((x^*(\tau), s^*(\tau), y^*(\tau))\)

with \(\tau > 0\) such that

\[ Ax + s - b = 0 \]
\[ A^T y + c = 0 \]
\[ s_i y_i = \tau \]
\[ s, y \geq 0 \]
Central path

minimize \( c^T x - \tau \sum_{i=1}^{m} \log(s_i) \)
subject to \( Ax + s = b \)

Set of points \((x^*(\tau), s^*(\tau), y^*(\tau))\)
with \( \tau > 0 \) such that
\[
Ax + s - b = 0 \\
A^T y + c = 0 \\
s_i y_i = \tau \\
s, y \geq 0
\]

Main idea
Follow central path as \( \tau \to 0 \)
The path parameter

Duality measure

$$\mu = \frac{s^T y}{m}$$ (average value of the pairs $s_i y_i$)
The path parameter

Duality measure

\[ \mu = \frac{s^T y}{m} \text{ (average value of the pairs } s_i y_i) \]

Linear system

\[
\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y
\end{bmatrix}
\begin{bmatrix}
\Delta y \\
\Delta x \\
\Delta s
\end{bmatrix}
= 
\begin{bmatrix}
-r_p \\
-r_d \\
-SY1 + \sigma \mu 1
\end{bmatrix}
\]
The path parameter

Duality measure
\[ \mu = \frac{s^T y}{m} \] (average value of the pairs \( s_i y_i \))

Centering parameter
\[ \sigma \in [0, 1] \]

Linear system
\[
\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y
\end{bmatrix}
\begin{bmatrix}
\Delta y \\
\Delta x \\
\Delta s
\end{bmatrix}
= \begin{bmatrix}
-r_p \\
-r_d \\
-SY \mathbf{1} + \sigma \mu \mathbf{1}
\end{bmatrix}
\]
The path parameter

Duality measure

\[ \mu = \frac{s^T y}{m} \] (average value of the pairs \( s_i y_i \))

Centering parameter

\[ \sigma \in [0, 1] \]

\[ \sigma = 0 \quad \Rightarrow \quad \text{Newton step} \]

\[ \sigma = 1 \quad \Rightarrow \quad \text{Centering step towards } (y^*(\mu), x^*(\mu), s^*(\mu)) \]

Linear system

\[
\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y
\end{bmatrix}
\begin{bmatrix}
\Delta y \\
\Delta x \\
\Delta s
\end{bmatrix}
=
\begin{bmatrix}
-r_p \\
r_d \\
-S Y 1 + \sigma \mu 1
\end{bmatrix}
\]
The path parameter

**Duality measure**

\[ \mu = \frac{s^T y}{m} \] (average value of the pairs \( s_i y_i \))

**Linear system**

\[
\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y
\end{bmatrix}
\begin{bmatrix}
\Delta y \\
\Delta x \\
\Delta s
\end{bmatrix}
= \begin{bmatrix}
-r_p \\
-r_d \\
-S Y 1 + \sigma \mu 1
\end{bmatrix}
\]

**Centering parameter**

- \( \sigma = 0 \) \( \Rightarrow \) Newton step
- \( \sigma = 1 \) \( \Rightarrow \) Centering step towards \((y^*(\mu), x^*(\mu), s^*(\mu))\)

**Line search** to enforce \( s, y > 0 \)

\((y, x, s) \leftarrow (y, x, s) + \alpha (\Delta y, \Delta x, \Delta s)\)
Path-following algorithm idea

Newton step
\[ \sigma = 0 \]

Centering step
\[ \sigma = 1 \]

Combined step
Path-following algorithm idea

Centering step
It brings towards the central path and is usually biased towards $s, y > 0$.
**No progress** on duality measure $\mu$
Path-following algorithm idea

Centering step
It brings towards the central path and is usually biased towards $s, y > 0$.

No progress on duality measure $\mu$

Newton step
It brings towards the zero duality measure $\mu$. Quickly violates $s, y > 0$. 

Combined step
Path-following algorithm idea

Centering step
It brings towards the central path and is usually biased towards $s, y > 0$. **No progress** on duality measure $\mu$.

Newton step
It brings towards the zero duality measure $\mu$. Quickly violates $s, y > 0$.

Combined step
Best of both worlds with longer steps
Path-following algorithm idea

Centering step
It brings towards the central path and is usually biased towards $s, y > 0$.
No progress on duality measure $\mu$

Newton step
It brings towards the zero duality measure $\mu$. Quickly violates $s, y > 0$.

Combined step
Best of both worlds with longer steps
Primal-dual path-following algorithm

Initialization
1. Given \((x_0, s_0, y_0)\) such that \(s_0, y_0 > 0\)

Iterations
1. Choose \(\sigma \in [0, 1]\)

\[
\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y
\end{bmatrix}
\begin{bmatrix}
\Delta y \\
\Delta x \\
\Delta s
\end{bmatrix}
= \begin{bmatrix}
-r_p \\
-r_d \\
-SY1 + \sigma \mu 1
\end{bmatrix}
\]

where \(\mu = s^T y/m\)

2. Solve

3. Find maximum \(\alpha\) such that \(y + \alpha \Delta y > 0\) and \(s + \alpha \Delta s > 0\)

4. Update \((y, x, s) \leftarrow (y, x, s) + \alpha(\Delta y, \Delta x, \Delta s)\)
Today’s lecture
Interior-point methods II

- Mehrotra predictor-corrector algorithm
- Implementation and linear algebra
- Interior-point vs simplex
Predictor-corrector algorithm
Main idea
Predict and select centering parameter

Predict
Compute Newton direction

Newton step
$\sigma = 0$

Centering step
$\sigma = 1$

Combined step
$x^*$
**Main idea**

Predict and select centering parameter

---

**Predict**

Compute Newton direction

**Estimate**

**How good** is the Newton step?
(how much can $\mu$ decrease?)
Main idea
Predict and select centering parameter

Predict
Compute Newton direction

Estimate
How good is the Newton step?
(how much can $\mu$ decrease?)

Select centering parameter
Very roughly:
Pick $\sigma \approx 0$ if Newton step is good
Pick $\sigma \approx 1$ if Newton step is bad
How good is the Newton step?

**Newton step**

\[(\Delta x_a, \Delta s_a, \Delta y_a)\]

**Maximum step-size**

\[\alpha_p = \max\{\alpha \in [0, 1] \mid s + \alpha \Delta s_a \geq 0\}\]

\[\alpha_d = \max\{\alpha \in [0, 1] \mid y + \alpha \Delta y_a \geq 0\}\]
How good is the Newton step?

Newton step

\((\Delta x_a, \Delta s_a, \Delta y_a)\)

Maximum step-size

\[\alpha_p = \max\{ \alpha \in [0, 1] \mid s + \alpha \Delta s_a \geq 0\}\]
\[\alpha_d = \max\{ \alpha \in [0, 1] \mid y + \alpha \Delta y_a \geq 0\}\]

Two issues

• The new points will not produce much improvement:
\((s + \alpha_p \Delta s_a)_i (y + \alpha_d \Delta y_a)_i\) much larger than 0

• The complementarity error depends on step lengths \(\alpha_p\) and \(\alpha_d\)
Choosing a centering parameter to make good improvement

**Newton step**

\[(\Delta x_a, \Delta s_a, \Delta y_a)\]

**Maximum step-size**

\[\alpha_p = \max\{\alpha \in [0, 1] \mid s + \alpha \Delta s_a \geq 0\}\]

\[\alpha_d = \max\{\alpha \in [0, 1] \mid y + \alpha \Delta y_a \geq 0\}\]
Choosing a centering parameter to make good improvement

Newton step

\((\Delta x_a, \Delta s_a, \Delta y_a)\)

Maximum step-size

\[\alpha_p = \max\{\alpha \in [0, 1] \mid s + \alpha \Delta s_a \geq 0\}\]
\[\alpha_d = \max\{\alpha \in [0, 1] \mid y + \alpha \Delta y_a \geq 0\}\]

Duality measure candidate
(after Newton step)

\[\mu_a = \frac{(s + \alpha_p \Delta s_a)^T (y + \alpha_d \Delta y_a)}{m}\]
Choosing a centering parameter to make good improvement

**Newton step**

\[
(\Delta x_a, \Delta s_a, \Delta y_a)
\]

**Maximum step-size**

\[
\alpha_p = \max\{\alpha \in [0, 1] \mid s + \alpha \Delta s_a \geq 0\}
\]

\[
\alpha_d = \max\{\alpha \in [0, 1] \mid y + \alpha \Delta y_a \geq 0\}
\]

**Duality measure candidate**

(after Newton step)

\[
\mu_a = \frac{(s + \alpha_p \Delta s_a)^T (y + \alpha_d \Delta y_a)}{m}
\]

**Centering parameter heuristic** \(\sigma\)

\[
\sigma = \left(\frac{\mu_a}{\mu}\right)^3
\]
Correcting for complementary error

Newton step

\[
\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y \\
\end{bmatrix}
\begin{bmatrix}
\Delta y_a \\
\Delta x_a \\
\Delta s_a \\
\end{bmatrix}
= 
\begin{bmatrix}
-r_p \\
-r_d \\
-SY1 \\
\end{bmatrix}
\]
Correcting for complementary error

Newton step

\[
\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y
\end{bmatrix}
\begin{bmatrix}
\Delta y_a \\
\Delta x_a \\
\Delta s_a
\end{bmatrix} =
\begin{bmatrix}
-r_p \\
-r_d \\
-SY1
\end{bmatrix}
\]

\[s_i(\Delta y_a)_i + y_i(\Delta s_a)_i + s_iy_i = 0\]
Correcting for complementary error

Newton step

\[
\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y
\end{bmatrix}
\begin{bmatrix}
\Delta y_a \\
\Delta x_a \\
\Delta s_a
\end{bmatrix}
= 
\begin{bmatrix}
-r_p \\
r_d \\
-SY1
\end{bmatrix}
\]

\[
s_i(\Delta y_a)_i + y_i(\Delta s_a)_i + s_iy_i = 0
\]

Complementarity error

\[
(s_i + (\Delta s_a)_i)(y_i + (\Delta y_a)_i) = (\Delta s_a)_i(\Delta y_a)_i \neq 0
\]

Complementarity violation
depends on step length
Correcting for complementary error

Newton step

\[
\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y \\
\end{bmatrix}
\begin{bmatrix}
\Delta y_a \\
\Delta x_a \\
\Delta s_a \\
\end{bmatrix}
= 
\begin{bmatrix}
-r_p \\
-r_d \\
-S Y 1 \\
\end{bmatrix}
\]

\[\rightarrow \quad s_i(\Delta y_a)_i + y_i(\Delta s_a)_i + s_i y_i = 0\]

Complementarity error

\[s_i + (\Delta s_a)_i)(y_i + (\Delta y_a)_i) = (\Delta s_a)_i(\Delta y_a)_i \neq 0\]

Corrected direction

\[
\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y \\
\end{bmatrix}
\begin{bmatrix}
\Delta y \\
\Delta x \\
\Delta s \\
\end{bmatrix}
= 
\begin{bmatrix}
-r_p \\
-r_d \\
-S Y 1 - \Delta S_a \Delta Y_a 1 + \sigma \mu 1 \\
\end{bmatrix}
\]

\[\Delta S_a = \text{diag}(\Delta s_a)\]
\[\Delta Y_a = \text{diag}(\Delta y_a)\]
Mehrotra predictor-corrector algorithm

Initialization

Given \((x, s, y)\) such that \(s, y > 0\)

1. Termination conditions

\[ r_p = Ax + s - b, \quad r_d = A^T y + c, \quad \mu = (s^T y)/m \]

If \(\|r_p\|, \|r_d\|, \mu\) are small, \textbf{break} Optimal solution \((x^*, s^*, y^*)\)

2. Newton step (affine scaling)

\[
\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y
\end{bmatrix}
\begin{bmatrix}
\Delta y_a \\
\Delta x_a \\
\Delta s_a
\end{bmatrix}
= \begin{bmatrix}
-r_p \\
-r_d \\
-SY 1
\end{bmatrix}
\]
Mehrotra predictor-corrector algorithm

3. Barrier parameter

$$\alpha_p = \max\{\alpha \in [0, 1] \mid s + \alpha \Delta s_a \geq 0\}$$

$$\alpha_d = \max\{\alpha \in [0, 1] \mid y + \alpha \Delta y_a \geq 0\}$$

$$\mu_a = \frac{(s + \alpha_p \Delta s_a)^T(y + \alpha_d \Delta y_a)}{m}$$

$$\sigma = \left(\frac{\mu_a}{\mu}\right)^3$$

4. Corrected direction

$$\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y
\end{bmatrix} \begin{bmatrix}
\Delta y \\
\Delta x \\
\Delta s
\end{bmatrix} = \begin{bmatrix}
-r_p \\
-r_d \\
-S Y 1 - \Delta S_a \Delta Y_a 1 + \sigma \mu 1
\end{bmatrix}$$
5. **Update iterates**

\[
\alpha_p = \max\{\alpha \geq 0 \mid s + \alpha \Delta s \geq 0\}
\]

\[
\alpha_d = \max\{\alpha \geq 0 \mid y + \alpha \Delta y \geq 0\}
\]

\[
(x, s) = (x, s) + \min\{1, \eta \alpha_p\}(\Delta x, \Delta s)
\]

\[
y = y + \min\{1, \eta \alpha_d\}\Delta y
\]

Avoid corners

\[
\eta = 1 - \epsilon \approx 0.99
\]
Implementation and linear algebra
Search equations

Step 2 (Newton) and 4 (Corrected direction) solve equations of the form

\[
\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y
\end{bmatrix}
\begin{bmatrix}
\Delta y \\
\Delta x \\
\Delta s
\end{bmatrix} =
\begin{bmatrix}
b_y \\
b_x \\
b_s
\end{bmatrix}
\]

The Newton step right hand side:

\[
\begin{bmatrix}
b_y \\
b_x \\
b_s
\end{bmatrix} =
\begin{bmatrix}
-r_p \\
r_d \\
-SY1
\end{bmatrix}
\]

The corrector step right hand side:

\[
\begin{bmatrix}
b_y \\
b_x \\
b_s
\end{bmatrix} =
\begin{bmatrix}
-r_p \\
r_d \\
-SY1 \quad -\Delta S_a \Delta Y_{a1} + \sigma \mu 1
\end{bmatrix}
\]
Solving the search equations

Our linear system is not symmetric

\[
\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y
\end{bmatrix}
\begin{bmatrix}
\Delta y \\
\Delta x \\
\Delta s
\end{bmatrix} =
\begin{bmatrix}
b_y \\
b_x \\
b_s
\end{bmatrix}
\]
Solving the search equations

Our linear system is not symmetric

\[
\begin{bmatrix}
0 & A & I \\
A^T & 0 & 0 \\
S & 0 & Y
\end{bmatrix}
\begin{bmatrix}
\Delta y \\
\Delta x \\
\Delta s
\end{bmatrix}
=
\begin{bmatrix}
b_y \\
b_x \\
b_s
\end{bmatrix}
\]

Substitute last equation, \( \Delta s = Y^{-1}(b_s - S\Delta y) \), into first

\[
\begin{bmatrix}
-Y^{-1}S & A \\
A^T & 0
\end{bmatrix}
\begin{bmatrix}
\Delta y \\
\Delta x
\end{bmatrix}
=
\begin{bmatrix}
b_y - Y^{-1}b_s \\
b_x
\end{bmatrix}
\]
Solving the search equations

Our reduced system is symmetric but not positive definite
Solving the search equations

Our reduced system is symmetric but not positive definite

\[
\begin{bmatrix}
-Y^{-1}S & A \\
AT & 0
\end{bmatrix}
\begin{bmatrix}
\Delta y \\
\Delta x
\end{bmatrix}
= \begin{bmatrix}
by - Y^{-1}b_s \\
b_x
\end{bmatrix}
\]

\[X^{-1}S \Delta y + A \Delta x = by - Y^{-1}b_s\]
Solving the search equations

Our reduced system is symmetric but not positive definite

\[
\begin{bmatrix}
-Y^{-1} S & A \\
A^T & 0
\end{bmatrix}
\begin{bmatrix}
\Delta y \\
\Delta x
\end{bmatrix}
= \begin{bmatrix}
b_y - Y^{-1} b_s \\
b_x
\end{bmatrix}
\]

Substitute first equation, \( \Delta y = S^{-1} Y (A \Delta x - b_y + Y^{-1} b_s) \), into second

\[
A^T S^{-1} Y A \Delta x = b_x + A^T S^{-1} Y b_y - A^T S^{-1} b_s
\]
Reduced linear system

Coefficient matrix

\[ B = A^T S^{-1} Y A \]

Characteristics

- \( A \) is large and sparse
- \( S^{-1} Y \) is positive and diagonal, different at each iteration
- \( B \) is positive definite if \( \text{rank}(A) = n \)
- Sparsity pattern of \( B \) is the pattern of \( A^T A \) (independent of \( S^{-1} Y \))
Reduced linear system

Coefficient matrix

\[ B = A^T S^{-1} Y A \]

Cholesky factorizations

\[ B = PLL^T P^T \]

- Reordering only once to get \( P \)
- One numerical factorization per interior-point iteration \( O(n^3) \)
- Forward/backward substitution twice per iteration \( O(n^2) \)
Reduced linear system

Coefficient matrix

\[ B = A^T S^{-1} Y A \]

Cholesky factorizations

\[ B = PLL^T P^T \]

- Reordering only once to get \( P \)
- One numerical factorization per interior-point iteration \( O(n^3) \)
- Forward/backward substitution twice per iteration \( O(n^2) \)

Per-iteration complexity

\( O(n^3) \)
Convergence

No convergence theory → Mehrotra’s algorithm → Examples where it diverges (rare!)
Convergence

Mehrotra’s algorithm

No convergence theory  Examples where it diverges (rare!)
Fantastic convergence in practice  Less than 30 iterations
Convergence

Mehrotra’s algorithm

No convergence theory → Examples where it **diverges** (rare!)
Fantastic convergence **in practice** → Less than 30 iterations

Theoretical iteration complexity
Alternative versions (slower than Mehrotra) converge in $O(\sqrt{n})$ iterations
Convergence

Mehrotra’s algorithm

No convergence theory — Examples where it **diverges** (rare!)

Fantastic convergence **in practice** — Less than 30 iterations

**Theoretical iteration complexity**
Alternative versions (slower than Mehrotra) converge in $O(\sqrt{n})$ iterations

**Average iteration complexity**
Average iterations complexity is $O(\log n)$
Convergence

Mehrotra’s algorithm

No convergence theory  Examples where it diverges (rare!)
Fantastic convergence in practice  Less than 30 iterations

Theoretical iteration complexity
Alternative versions (slower than Mehrotra) converge in $O(\sqrt{n})$ iterations

Average iteration complexity
Average iterations complexity is $O(\log n)$

Floating point operations
$O(n^{3.5})$

$O(n^{3 \log n})$
Warm-starting

Interior-point methods are difficult to warm-start
Warm-starting

Interior-point methods are difficult to warm-start

Previous solution
Warm-starting

Interior-point methods are difficult to warm-start

Previous solution

Badly centered initial point

Hard to make progress with long steps
Interior-point vs simplex
Example

minimize $-10x_1 - 12x_2 - 12x_3$
subject to
$x_1 + 2x_2 + 2x_3 \leq 20$
$2x_1 + x_2 + x_3 \leq 20$
$2x_1 + 2x_2 + x_3 \leq 20$
$x_1, x_2, x_3 \geq 0$

$x^* = (4, 4, 4)$
Example

minimize \(-10x_1 - 12x_2 - 12x_3\)
subject to
\(x_1 + 2x_2 + 2x_3 \leq 20\)
\(2x_1 + x_2 + x_3 \leq 20\)
\(2x_1 + 2x_2 + x_3 \leq 20\)
\(x_1, x_2, x_3 \geq 0\)

\(c = (-10, -12, -12)\)
\(A = \begin{bmatrix} 1 & 2 & 2 \\ 2 & 1 & 2 \\ 2 & 2 & 1 \end{bmatrix}\)
\(b = (20, 20, 20)\)
Example with real solver
CVXOPT (open-source)

Code

```python
import numpy as np
import cvxpy as cp

c = np.array([-10, -12, -12])
A = np.array([[1, 2, 2],
              [2, 1, 2],
              [2, 2, 1]])
b = np.array([20, 20, 20])
n = len(c)

x = cp.Variable(n)
problem = cp.Problem(cp.Minimize(c @ x),
                     [A @ x <= b, x >= 0])
problem.solve(solver=cp.CVXOPT, verbose=True)
```

Output

```
In [3]: x.value
Out[3]: array([3.99999999, 4., 4.])
```

Solution

```
pcost    dcost    gap    pres    dres    k/t
0: -1.3077e+02 -2.3692e+02  2e+01  1e-16  6e-01  1e+00
1: -1.3522e+02 -1.4089e+02  1e+00  2e-16  3e-02  4e-02
2: -1.3599e+02 -1.3605e+02  1e-02  2e-16  3e-04  4e-04
3: -1.3600e+02 -1.3600e+02  1e-04  1e-16  3e-06  4e-06
4: -1.3600e+02 -1.3600e+02  1e-06  1e-16  3e-08  4e-08
Optimal solution found.
```

[The CVXOPT linear and quadratic cone program solvers, L. Vandenberghe 2010]
Average interior-point complexity

Random LPs

minimize $c^T x$  
subject to $Ax \leq b$  
n variables  
$3n$ constraints
Average interior-point complexity

Random LPs

\[
\begin{align*}
\text{minimize} & \quad c^T x & \quad n \text{ variables} \\
\text{subject to} & \quad Ax \leq b & \quad 3n \text{ constraints}
\end{align*}
\]

\textbf{Iterations: } \mathcal{O}(\log n)
Average interior-point complexity

Random LPs

\[
\begin{align*}
\text{minimize} & \quad c^T x \\
\text{subject to} & \quad Ax \leq b \\
& \quad n \text{ variables} \\
& \quad 3n \text{ constraints}
\end{align*}
\]

**Iterations:** \( O(\log n) \)

**Time:** \( O(n^3 \log n) \)
Comparison between interior-point method and simplex

**Primal simplex**
- Primal feasibility
- Zero duality gap

**Dual feasibility**

**Dual simplex**
- Dual feasibility
- Zero duality gap

**Primal feasibility**

**Primal-dual interior-point**
- Interior condition
  - Primal feasibility
  - Dual feasibility
  - Zero duality gap
Comparison between interior-point method and simplex

Primal simplex
- Primal feasibility
- Zero duality gap

Dual feasibility

Dual simplex
- Dual feasibility
- Zero duality gap

Primal feasibility

Primal-dual interior-point
- Interior condition
- Primal feasibility
- Dual feasibility
- Zero duality gap

Polynomial worst-case complexity

Exponential worst-case complexity
## Comparison between interior-point method and simplex

<table>
<thead>
<tr>
<th>Primal simplex</th>
<th>Dual simplex</th>
<th>Primal-dual interior-point</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Primal feasibility</td>
<td>• Dual feasibility</td>
<td>• Interior condition</td>
</tr>
<tr>
<td>• Zero duality gap</td>
<td>• Zero duality gap</td>
<td>• Primal feasibility</td>
</tr>
<tr>
<td>Dual feasibility</td>
<td>Primal feasibility</td>
<td>• Dual feasibility</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Zero duality gap</td>
</tr>
</tbody>
</table>

- **Exponential worst-case complexity**
  - Requires feasible point

- **Polynomial worst-case complexity**
  - Allows infeasible start
Comparison between interior-point method and simplex

**Primal simplex**
- Primal feasibility
- Zero duality gap

**Dual feasibility**

**Dual simplex**
- Dual feasibility
- Zero duality gap

**Primal feasibility**

**Primal-dual interior-point**
- Interior condition
  - Primal feasibility
  - Dual feasibility
  - Zero duality gap

**Exponential worst-case complexity**
- Requires feasible point
- Can be warm-started

**Polynomial worst-case complexity**
- Allows infeasible start
- Cannot be warm-started
Which algorithm should I use?

Dual simplex
- Small-to-medium problems
- Repeated solves with varying data

Interior-point (barrier)
- Medium-to-large problems
- Sparse structured problems
Which algorithm should I use?

<table>
<thead>
<tr>
<th>Dual simplex</th>
<th>Interior-point (barrier)</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Small-to-medium problems</td>
<td>• Medium-to-large problems</td>
</tr>
<tr>
<td>• Repeated solves with varying data</td>
<td>• Sparse structured problems</td>
</tr>
</tbody>
</table>

How do solvers with multiple options decide?
Concurrent Optimization
Which algorithm should I use?

**Dual simplex**
- Small-to-medium problems
- Repeated solves with varying data

**Interior-point (barrier)**
- Medium-to-large problems
- Sparse structured problems

How do solvers with multiple options decide?
Concurrent Optimization

**Why not both?** (crossover)
Interior-point ➔ Few simplex steps
Interior-point methods implementation

Today, we learned to:

• Apply Mehrotra predictor-corrector algorithm
• Exploit linear algebra to speedup computations
• Analyze empirical complexity
• Compare interior-point and simplex methods
References

• D. Bertsimas and J. Tsitsiklis: Introduction to Linear Optimization
  • Chapter 9.4 — 9.6: Interior point methods

• R. Vanderbei: Linear Programming
  • Chapter 17: The Central Path
  • Chapter 15: A Path-Following Method
Next lecture

- Overview for linear optimization