Complexity of the Simplex Method

Given a particular instance of a linear program, can we predict how long it will take to solve it?

- how much time does it take to perform one iteration?
- what is the overall number of iterations?

The principal work associated with an iteration of the simplex method comes from the \textit{linear algebra}.

Time for one iteration depends on the size of the problem data, i.e., the dimensions of $A$, $b$ and $c$.

The main overhead is the cost of solving the systems:

- \[
  A^T_k \lambda_k = c_k \quad \text{and} \quad A_k p_k = e_s \quad (\text{all-inequality form})
\]
- \[
  B^T \pi = c_B \quad \text{and} \quad B p_B = -a_{vs} \quad (\text{standard form})
\]

Work to solve $n$ equations in $n$ unknowns “from scratch”?

\[ W(n) = c_3 n^3 + c_2 n^2 + c_1 n + c_0 \]

calculations

where $c_3$, $c_2$, $c_1$, $c_0$ depend on the method.

In “big O” notation, the solve will take $O(n^3)$ calculations

e.g., for $n = 1000$ the work is $10^9$ calculations per system!

Fortunately, when solving these equations for the simplex method we do not have to solve them from scratch.

We can exploit the fact that the equations defining two adjacent vertices are \textit{very similar}.

Standard-Form Simplex Method

Consider solving an LP in standard form using the simplex method.

Each iteration, the main work is solving:

\[
  B^T \pi = c_B \quad \text{and} \quad B p_B = -a_{vs}
\]

If $B$ is the basis matrix at the $k$th iteration, then the matrix for the $(k+1)$-th iteration differs from $B$ by one column.

i.e., the basis matrix undergoes \textit{column replacement}.
Column replacement

Let $B$ and $\bar{B}$ be the old and new basis matrices.

$$B = \begin{pmatrix} a_{\beta_1} & \cdots & a_{\beta_t} & \cdots & a_{\beta_m} \end{pmatrix}$$

$$\bar{B} = \begin{pmatrix} a_{\beta_1} & \cdots & a_{\nu_s} & \cdots & a_{\beta_m} \end{pmatrix}$$

Equivalently,

$$\bar{B} = \begin{pmatrix} a_{\beta_1} & \cdots & a_{\beta_t} & \cdots & a_{\beta_m} \end{pmatrix} - \begin{pmatrix} 0 & \cdots & a_{\beta_t} & \cdots & 0 \end{pmatrix} + \begin{pmatrix} 0 & \cdots & a_{\nu_s} & \cdots & 0 \end{pmatrix}$$

$$\begin{pmatrix} a_{\beta_t} \epsilon_t^T & a_{\nu_s} \epsilon_t^T \end{pmatrix}$$

To replace column $t$ of $B$ by column $a_{\nu_s}$ we form

$$B - a_{\beta_t} \epsilon_t^T + a_{\nu_s} \epsilon_t^T = \begin{pmatrix} a_{\beta_1} & \cdots & a_{\nu_s} & \cdots & a_{\beta_m} \end{pmatrix}$$

We can combine these steps as

$$\bar{B} = B - a_{\beta_t} \epsilon_t^T + a_{\nu_s} \epsilon_t^T = B + (a_{\nu_s} - a_{\beta_t}) \epsilon_t^T$$

$\Rightarrow \bar{B}$ differs from $B$ by a matrix of rank one.

$\Rightarrow \bar{B}^{-1}$ can be found in $O(m^2)$ operations instead of $O(m^3)$.

Sherman-Morrison formula

First, consider the general case of a rank-one update:

$$\bar{B} = B + uv^T$$

where $u$ and $v$ are $m$-vectors. Then

$$\bar{B}^{-1} = (B + uv^T)^{-1}$$

$$= B^{-1} - \frac{1}{\tau} B^{-1} uv^T B^{-1}$$

with $\tau = 1 + v^T B^{-1} u$

$\Rightarrow \bar{B}^{-1}$ differs from $B^{-1}$ by a matrix of rank one.

If $\bar{B} = B + (a_{\nu_s} - a_{\beta_t}) \epsilon_t^T$, then the Sherman-Morrison formula gives

$$\bar{B}^{-1} = B^{-1} - \frac{1}{\tau} B^{-1} uv^T B^{-1}$$

with $\tau = 1 + v^T B^{-1} u$

and

$$u = a_{\nu_s} - a_{\beta_t} \text{ and } v = \epsilon_t$$

$\Rightarrow \bar{B}^{-1}$ can be found in $O(m^2)$ operations instead of $O(m^3)$. 
**All-inequality simplex method**

Similar methods are used to update the inverse when solving problems in all-inequality form.

In this case, the working-set matrix undergoes *row replacement*

\[ A_{k+1} = A_k + e_k (a_{j_k} - a_{w_k})^T \]

where \(a_{j_k}\) and \(a_{w_k}\) are the normals of the constraints that are entering and leaving the working set.

Row replacement is the same as column replacement on the transpose.

---

**Recap: complexity of the simplex method**

Given a particular linear program, can we predict how long it will take to be solved?

- **How much time does it take to do one iteration?**
  \[ \implies O(r^2) \text{ operations, where } r = \min \{m, n\} \]

- **How many iterations will be required overall?**
  The simplex method is guaranteed to terminate after a finite number of iterations.
  However, the number of iterations cannot be determined in advance because it depends on the data \(A, b\) and \(c\), and the starting point \(x_0\).

---

- In practice, many LPs have a matrix \(A\) that is *sparse*, i.e., \(A\) has only a few nonzero elements in each column.

In this case, the work to solve for \(p_B\) or \(\pi\) is \(O(dm)\).

- Modern methods define \(B = LU\), where \(L\) and \(U\) are triangular

  \[ Bx = b \implies Ly = b \quad \text{and} \quad Ux = y \]

  In this case \(L\) and \(U\) are updated to reflect the change in \(B\).

---

- A *worst-case* analysis ...
  - derives an *upper bound* on the number of iterations required for *any* problem instance.

  - provides a guarantee that the algorithm cannot take more than a certain number of iterations, no matter how unfavorable the circumstances.

- An *average-case analysis* ...
  - attempts to formalize the “expected” number of iterations by drawing conclusions about performance from the known or hypothesized nature of a “typical” \(A, b\) and \(c\).
Recall, if the problem contains \( m \) inequality constraints in \( n \) variables with \( m \geq n \), the number of vertices could be as large as
\[
\binom{m}{n} = \frac{m!}{n!(m-n)!}
\]
which gives the number of combinations of \( n \) active constraints chosen from the \( m \) possibilities.

As \( m \) and \( n \) increase, this number becomes enormous, and the simplex method would be hopelessly time-consuming if the number of iterations remotely approached this size.

**Klee-Minty Examples**

The Klee-Minty examples can be generalized to be of any dimension, and are usually described as “distortions of the hypercube”.

The feasible region for a Klee-Minty example involving \( n \) variables contains \( 2^n \) vertices, and the simplex method examines every vertex!

The simplex method has exponential complexity in the worst case.

Practical computation tends almost to the other extreme.

In practice, the number of simplex iterations needed to solve a “typical” standard-form problem with \( n \) variables and \( m \) constraints is usually a small multiple (say, 3–6) of \( m \).

The number of iterations also seems to display a sublinear growth with \( n \).

On the other hand, the worst-case bound is real, and the simplex method can require an exponential number of iterations.

The best known linear programs that cause worst-case behavior by the simplex method were devised in the early 1970s by Klee and Minty.

**Example:**

maximize \( \frac{6}{5}x_1 + x_2 \)
subject to the constraints:
- constraint #1: \( \frac{3}{2}x_1 + x_2 \leq \frac{5}{2} \)
- constraint #3: \( x_1 \geq 0 \)
- constraint #4: \( x_1 \leq \frac{7}{5} \)
- constraint #5: \( x_2 \geq 0 \)

**Interior Methods**

Consider an LP in standard form:

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

where \( A \) has \( m \) rows.

The optimality conditions are:

\[
\begin{align*}
A^T \pi^* + z^* &= c, \quad z^* \geq 0 \\
Ax^* &= b, \quad x^* \geq 0 \\
x^* \cdot z^* &= 0
\end{align*}
\]

where \( x^* \cdot z^* \) is a vector with components \( x_i^* z_i^* \).

We can rewrite the conditions

\[
\begin{align*}
A^T \pi^* + z^* &= c \\
Ax^* &= b \\
x^* \cdot z^* &= 0
\end{align*}
\]

as a set of \( 2n + m \) equations \( F(x, \pi, z) = 0 \), where

\[
F(x, \pi, z) = \begin{pmatrix} c - A^T \pi - z \\ Ax - b \\ x \cdot z \end{pmatrix}
\]

These equations are **almost linear**. The only nonlinear terms come from the nonlinear equations \( x \cdot z = 0 \).

The **optimality conditions**:

\[
\begin{align*}
A^T \pi^* + z^* &= c, \quad z^* \geq 0 \\
Ax^* &= b, \quad x^* \geq 0 \\
x^* \cdot z^* &= 0
\end{align*}
\]

define \( 2n + m \) conditions on the \( 2n + m \) unknowns \( x^*, \pi^* \) and \( z^* \).

Suppose we have an approximate solution \( x, \pi \) and \( z \) with \( x \geq 0 \) and \( z \geq 0 \).

**Basic idea**: Define \( \Delta x, \Delta \pi \) and \( \Delta z \) such that

\[
F(x + \Delta x, \pi + \Delta \pi, z + \Delta z) \approx 0
\]

with \( x + \Delta x \geq 0 \) and \( z + \Delta z \geq 0 \).
We want $\Delta x$, $\Delta \pi$ and $\Delta z$ such that

$$c - A^T(\pi + \Delta \pi) - (z + \Delta z) = 0, \quad z + \Delta z \geq 0$$

$$A(x + \Delta x) - b = 0, \quad x + \Delta x \geq 0$$

$$(x + \Delta x) \cdot (z + \Delta z) = 0$$

Moving known quantities to the right-hand side, we get

$$-A^T \Delta \pi - \Delta z = z - c + A^T \pi$$

$$A \Delta x = b - Ax$$

$$\Delta x \cdot z + x \cdot \Delta z + \Delta x \cdot \Delta z = -x \cdot z$$

If we assume that the nonlinear term $\Delta x \cdot \Delta z \approx 0$, then we get linear equations for $\Delta x$, $\Delta \pi$ and $\Delta z$, i.e.,

$$-A^T \Delta \pi - \Delta z = z - c + A^T \pi$$

$$A \Delta x = b - Ax$$

$$\Delta x \cdot z + x \cdot \Delta z = -x \cdot z$$

To simplify the equations, we define

$$X = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

and

$$Z = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{pmatrix}$$

then

$$\Delta x \cdot z + x \cdot \Delta z = z \cdot \Delta x + x \cdot \Delta z$$

$$= Z \Delta x + X \Delta z$$

The linear equations are

$$-A^T \Delta \pi - \Delta z = z - c + A^T \pi$$

$$A \Delta x = b - Ax$$

$$Z \Delta x + X \Delta z = -X z$$

In matrix form, the equations are

$$\begin{pmatrix} 0 & -A^T & -I \\ A & 0 & 0 \\ Z & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta \pi \\ \Delta z \end{pmatrix} = \begin{pmatrix} c - A^T \pi - z \\ Ax - b \\ Xz \end{pmatrix}$$
How do we keep \( x + \Delta x \) and \( z + \Delta z \) feasible?

As \((x, z)\) is feasible, we can take the maximum feasible step along \((\Delta x, \Delta \pi, \Delta z)\).

i.e., define a step \( \alpha \) such that
\[
\begin{align*}
 x + \alpha \Delta x & \geq 0 \\
z + \alpha \Delta z & \geq 0
\end{align*}
\]

The step \( \alpha \) is then
\[
\alpha = \min \{1, \alpha_{\text{max}}\}
\]

Once \( \alpha \) is computed, the variables are updated as
\[
\begin{align*}
x & \leftarrow x + \alpha \Delta x \\
\pi & \leftarrow \pi + \alpha \Delta \pi \\
z & \leftarrow z + \alpha \Delta z
\end{align*}
\]

and the process is repeated.

Are we sure we can solve the system for \( \Delta x, \Delta \pi, \Delta z \)?

\( \Rightarrow \) Yes! It can be shown under certain conditions that the system is nonsingular.

Define the positive steps to the boundary of each constraint:
\[
\begin{align*}
\sigma_i^{(x)}(x) & = \begin{cases} 
 x_i - \lfloor \Delta x_i \rfloor & \text{if } \lfloor \Delta x_i \rfloor < 0 \\
 +\infty & \text{if } \lfloor \Delta x_i \rfloor \geq 0 
\end{cases} \\
\sigma_i^{(z)}(z) & = \begin{cases} 
 z_i - \lfloor \Delta z_i \rfloor & \text{if } \lfloor \Delta z_i \rfloor < 0 \\
 +\infty & \text{if } \lfloor \Delta z_i \rfloor \geq 0 
\end{cases}
\end{align*}
\]

Note that \( \pi \) has no upper or lower bounds.

Define the maximum feasible steps
\[
\alpha^{(x)} = \min_i \{\sigma_i^{(x)}\}, \quad \alpha^{(z)} = \min_i \{\sigma_i^{(z)}\}
\]

and compute the overall maximum feasible step
\[
\alpha_{\text{max}} = \min \{\alpha^{(x)}, \alpha^{(z)}\}
\]

\(-A^T \Delta \pi - \Delta z = z - c + A^T \pi\) \hspace{1cm} (1)
\[A \Delta x = b - Ax\] \hspace{1cm} (2)
\[Z \Delta x + X \Delta z = -Xz\] \hspace{1cm} (3)

If we use \( \Delta z \) from (1) and substitute in (3) we get
\[
Z \Delta x - XA^T \Delta \pi = X(z + A^T \pi - c) - Xz
\]

Using this equation with (2) gives
\[
\begin{pmatrix} Z & -XA^T \\ A & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta \pi \end{pmatrix} = - \begin{pmatrix} X(c - A^T \pi - z) + Xz \\ AX - b \end{pmatrix}
\]

from which we recover \( \Delta z \) as
\[
\Delta z = c - A^T(\pi + \Delta \pi) - z
\]
\[
\begin{pmatrix}
Z & -XA^T \\
A & 0
\end{pmatrix}
\begin{pmatrix}
\Delta x \\
\Delta \pi
\end{pmatrix}
= -\begin{pmatrix}
X(c - A^T \pi - z) + Xz \\
Ax - b
\end{pmatrix}
\]

If these equations are to define $\Delta x$, $\Delta \pi$, and $\Delta z$, they must be nonsingular.

At the very least, they should be nonsingular at $(x^*, \pi^*, z^*)$.

---

**Definition**

We say that strict complementarity holds between $z \geq 0$ and $x \geq 0$ if, for every $i$, $z_i > 0$ and $x_i = 0$ or $z_i = 0$ and $x_i > 0$.

i.e., $z_i x_i = 0$, but $z_i$ and $x_i$ cannot both be zero.

---

**Result**

Assume that $x^*$ is a basic solution of the LP:

\[
\begin{array}{c}
\text{minimize} & c^T x \\
\text{subject to} & Ax = b, \quad x \geq 0
\end{array}
\]

If strict complementarity holds at $x^*$, then the matrix

\[
\begin{pmatrix}
Z^* & -X^* A^T \\
A & 0
\end{pmatrix}
\]

is nonsingular at $(x^*, \pi^*, z^*)$.

---

How does this method perform in practice?

**Very, VERY Badly!**

Many of the $x$ and $z$ variables become jammed at 0 with $\alpha = 0$, which means that the algorithm never moves.

This happens because $(x, z)$ may want to approach the solution from the infeasible side of the constraints $x \geq 0$ and $z \geq 0$. 
This difficulty can be overcome by making a very simple change.

**Basic idea:**
- Start with $x_0 > 0$ and $z_0 > 0$
- Perturb the equations so that the solution is moved *inside* the region $x \geq 0$ and $z \geq 0$.
- Keep the iterates *strictly* feasible, i.e., keep $x > 0$ and $z > 0$.

$\Rightarrow$ $x$ and $z$ lie in the interior of the region $x \geq 0$ and $z \geq 0$.

Define a small positive $\mu$ (say $\mu = 10^{-2}$).

Perturb the optimality conditions to give

$$
A^T \pi + z = c, \quad z \geq 0 \\
Ax = b, \quad x \geq 0 \\
x \cdot z = \mu e
$$

where $e$ is the $n$-vector of ones $(1, 1, \ldots, 1)$.

The perturbation forces the $x_i$ and $z_i$ to satisfy

$$
x_i z_i = \mu > 0
$$

which means that we converge to $x_i > 0$ and $z_i > 0$.

The equations are now $F_\mu(x, \pi, z) = 0$, where

$$
F_\mu(x, \pi, z) = \begin{pmatrix}
c - A^T \pi - z \\
Ax - b \\
x \cdot z - \mu e
\end{pmatrix}
$$

The linear equations to be solved at each step are:

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

*only the right-hand side has changed!*

We keep $x$ and $z$ *interior* by stopping short of the boundary.

Define a damping parameter $0 < \theta < 1$ (e.g., $\theta = 0.99$)

If $\alpha_{\text{max}} = \min\{\alpha^{(x)}_{\text{max}}, \alpha^{(z)}_{\text{max}}\}$ as before,

$$
\alpha \triangleq \min\{1, \theta \alpha_{\text{max}}\}
$$

This keeps $x$ and $z$ *inside* the region $x \geq 0$ and $z \geq 0$.

Once an *approximate solution* of $F_\mu(x, \pi, z) = 0$ is found, $\mu$ is reduced and the process is repeated.
Interior Method Algorithm

Set $\mu = 10^{-2}$;
Choose $x$, $\pi$ and $z$ with $x^T > 0$ and $z > 0$;
while $\|F(x, \pi, z)\| \geq 10^{-6}$ do
  while $\|F_\mu(x, \pi, z)\| \geq 10^{-6}$ do
    Compute $(\Delta x, \Delta \pi)$ from the system
    \[
    \begin{pmatrix}
        0 & -A^T & -I \\
        A & 0 & 0 \\
        Z & 0 & X
    \end{pmatrix}
    \begin{pmatrix}
        \Delta x \\
        \Delta \pi \\
        \Delta z
    \end{pmatrix}
    =
    -\begin{pmatrix}
        c - A^T \pi - z \\
        Ax - b \\
        x \cdot z - \mu e
    \end{pmatrix};
    \]
    Set $\alpha(\max)_{\Delta x}$ = maximum feasible step along $\Delta x$;
    Set $\alpha(\max)_{\Delta z}$ = maximum feasible step along $\Delta z$;
    Compute $\alpha_{\max} = \min\{ \alpha(\max)_{\Delta x}, \alpha(\max)_{\Delta z} \}$;
    Set $\alpha = \min\{ 1, 0.99 \alpha_{\max} \}$;
    $x \leftarrow x + \alpha \Delta x$; \quad $\pi \leftarrow \pi + \alpha \Delta \pi$; \quad $z \leftarrow z + \alpha \Delta z$;
  end do
  $\mu \leftarrow \mu/10$;
end do

This is the **primal-dual interior method**.
- The equations
  \[
  \begin{pmatrix}
    0 & -A^T & -I \\
    A & 0 & 0 \\
    Z & 0 & X
  \end{pmatrix}
  \begin{pmatrix}
    \Delta x \\
    \Delta \pi \\
    \Delta z
  \end{pmatrix}
  =
  -\begin{pmatrix}
    c - A^T \pi - z \\
    Ax - b \\
    x \cdot z - \mu e
  \end{pmatrix}
  \]
  must be solved from scratch at each step.
  - The equations involve all columns of $A$
  - Suitable choices of $\mu$ give polynomial complexity
  - In practice, very few iterations are needed
  \[
  \bullet \quad \text{more than 30–50 iterations is very unusual}
  \]