Primal–Dual
Interior Point Algorithm
(Path–Following Algorithm)
for Linear Programming

This Hypercard stack was prepared by:
Dennis L. Bricker,
Dept. of Industrial Engineering,
University of Iowa,
Iowa City, Iowa 52242
e-mail: dbricker@icaen.uiowa.edu
Consider the primal/dual pair of LPs:

**Primal**

Minimize $c^t x$
subject to $Ax = b$
$x \geq 0$

**Dual**

Maximize $y^t b$
subject to $yA \leq c^t$

i.e.,

Maximize $b^t y$
subject to $A^t y \leq c$
Convert dual constraints to equalities:

**Primal**

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

**Dual**

\[
\begin{align*}
\text{Maximize } & \quad b^T y \\
\text{subject to } & \quad A^T y + z = c^T \\
& \quad z \geq 0
\end{align*}
\]
Use barrier functions to relax the non-negativity conditions:

**Primal**

Minimize $c^T x - \mu \sum_{j=1}^{n} \ln(x_j)$

subject to

$A^T x = b$

as $x \to 0$, 
$-\mu \ln(x) \to \infty$

**Dual**

Maximize $b^T y + \mu \sum_{j=1}^{n} \ln(z_j)$

subject to

$A^T y + z = c^T$

©D. Bricker, U. of I, 2000
Use Lagrange multipliers to relax the equality constraints:

\[ L_p(x, y) = c^t x - \mu \sum_{j=1}^{n} \ln(x_j) + y^t(A x - b) \]

\[ L_D(x, y, z) = b^t y + \mu \sum_{j=1}^{n} \ln(x_j) - x^t(A^t y + z - c) \]
The optimality conditions may be written:

\[ \frac{\partial L_P(x, y)}{\partial x} = 0, \quad \frac{\partial L_P(x, y)}{\partial y} = 0 \]

and

\[ \frac{\partial L_D(x, y, z)}{\partial x} = 0, \quad \frac{\partial L_D(x, y, z)}{\partial y} = 0, \quad \frac{\partial L_D(x, y, z)}{\partial z} = 0 \]
These reduce to the following optimality conditions

\[
\begin{align*}
A \mathbf{x} &= b \\
A^t \mathbf{y} + \mathbf{z} &= c \\
\mathbf{x}_j \mathbf{z}_j &= \mu, \quad j=1,2, \ldots, n
\end{align*}
\]
To solve the nonlinear system of equations, we might use the *Newton–Raphson* method:

Given an initial approximate solution \((x^0, y^0, z^0)\): an improved approximate solution is given by

\[
\begin{align*}
x^1 &= x^0 + \delta_x \\
y^1 &= y^0 + \delta_y \\
z^1 &= z^0 + \delta_z
\end{align*}
\]

where \(\delta_x\), \(\delta_y\), and \(\delta_z\) are found by solving a linear system.
**Notation**

\[ X = \text{diag}\{x_1, x_2, \ldots, x_n\} \]

\[ Z = \text{diag}\{z_1, z_2, \ldots, z_n\} \]

\[ e = [1, 1, \ldots, 1] \]

Then the constraints

\[ x_j z_j = \mu, \ j=1,2, \ldots, n \]

may be written

\[ XZ e = \mu e \]
We wish to solve the **nonlinear** system

\[
\begin{align*}
A \delta_x & = -[A x^0 - b] \\
A^t \delta_y + \delta_z & = -[A^t y^0 + z^0 - c] \\
X \delta_x + Z \delta_y & = -[X Z e - \mu e]
\end{align*}
\]

Newton–Raphson Method: given \((x^0, y^0, z^0)\), solve the **linear** system

\[
\begin{align*}
A \delta_x & = -[A x^0 - b] \\
A^t \delta_y + \delta_z & = -[A^t y^0 + z^0 - c] \\
X \delta_x + Z \delta_y & = -[X Z e - \mu e]
\end{align*}
\]
That is, solve

\[
\begin{bmatrix}
  A & 0 & 0 \\
  0 & A^t & I \\
  Z & 0 & X
\end{bmatrix}
\begin{bmatrix}
  \delta_x \\
  \delta_y \\
  \delta_z
\end{bmatrix}
= \begin{bmatrix}
  d_P \\
  -d_D \\
  \mu e - X Z e
\end{bmatrix}
\]

where \( d_P = b - Ax^0 \)

\( d_D = A^t y^0 + z^0 - c \)  \hspace{1cm} \leftarrow \text{primal infeasibility}

\( d_D = A^t y^0 + z^0 - c \)  \hspace{1cm} \leftarrow \text{dual infeasibility}

and then compute the improved approximation

\[
\begin{cases}
  x^1 = x^0 + \delta_x \\
  y^1 = y^0 + \delta_y \\
  z^1 = z^0 + \delta_z
\end{cases}
\]

©D. Bricker, U. of IA, 2000
Solving the linear system:

\[ \delta_x = Z^{-1} \left[ \mu e - XZ \, e - X \, \delta_z \right] \]

\[ \delta_z = -d_D - A^t \, \delta_y \]

\[ \Rightarrow \begin{bmatrix} A \, Z^{-1} \, X \, A^t \end{bmatrix} \delta_y = b - \mu A \, Z^{-1} \, e - AZ^{-1} \, X \, d_D \]

or

\[ \delta_y = \left[ A \, Z^{-1} \, X \, A^t \right]^{-1} \left( b - \mu A \, Z^{-1} \, e - AZ^{-1} \, X d_D \right) \]
Computing

\[ \delta_y = \left[ A Z^{-1}X A^t \right]^{-1} \left( b - \mu A Z^{-1}e - AZ^{-1}X d_D \right) \]

by using matrix inversion is computationally costly for large problems...
other methods for solving the linear system for \( \delta_y \) are preferred.
After computing the step \((\delta_x, \delta_y, \delta_z)\),
\[
\begin{align*}
x^1 &= x^0 + \delta_x \\
y^1 &= y^0 + \delta_y \\
z^1 &= z^0 + \delta_z
\end{align*}
\]

An alternative would be to go (almost) as far as possible in the \(x\) direction and the \((y,z)\) direction:
\[
\begin{align*}
x^1 &= x^0 + \alpha_p \delta_x \\
y^1 &= y^0 + \alpha_D \delta_y \\
z^1 &= z^0 + \alpha_D \delta_z
\end{align*}
\]

for stepsizes \(\alpha_p\) and \(\alpha_D\), respectively.
\[ \alpha_p = \tau \min_j \left\{ \frac{-x_j^0}{\delta_{xj}} : \delta_{xj} < 0 \right\} \]

\[ \alpha_D = \tau \min_j \left\{ \frac{-z_j^0}{\delta_{zj}} : \delta_{zj} < 0 \right\} \]

for \( 0 < \tau < 1 \)

\( \tau = 1 \) will result in one of the \( x \) and \( z \) variables reaching zero!

e.g., \( \tau = 0.995 \)
Generally, only one Newton–Raphson step is used, so that the nonlinear system is only approximately solved.

This completes one iteration. As $\mu \to 0$, the values of $x, y, z$ will converge to the optimal primal and dual solutions. The path followed by $(x, y, z)$ is referred to as the central path and the algorithm as a path-following algorithm.

©D Bricker, U of IA, 2000
Reduction of $\mu$:

$$\mu = \frac{c^t x^1 - b^t y^1}{\theta(n)}$$

suggested value of parameter $\theta$:

$$\theta(n) = \begin{cases} 
n^2 & \text{if } n \leq 5,000 \\
n \sqrt{n} & \text{if } n > 5,000 
\end{cases}$$
Termination criterion:

\[
\frac{c^t x^k - b^t y^k}{1 + |b^t y^k|} < \epsilon
\]

The number of iterations required is rather insensitive to the size of the problem, and is usually between 20 and 80 for most problems.