Linear Programming: The Simplex Method II

Note Title 2/14/2022

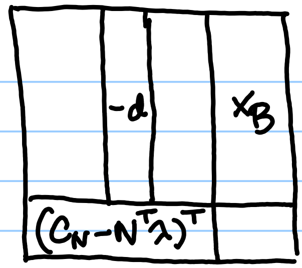
## Remaining Important Issues

- Linear algebra issues—maintaining an LU factorization of $B$ that can be used to solve for $\lambda$ and $d$.
- Selection of the entering index $q$ from among the negative components of $s_N$. (In general, there are many such components.)
- Handling of degenerate bases and degenerate steps, in which it is not possible to choose a positive value of $x_q^+$ without violating feasibility.
- Finding an initial $B$ / vertex.

**Procedure 13.1** (One Step of Simplex).

Given $\mathcal{B}, \mathcal{N}, x_B = B^{-1}b \geq 0, x_N = 0$;
Solve $B^T \lambda = c_B$ for $\lambda$,
Compute $s_N = c_N - N^T \lambda$; (* pricing *)
**if** $s_N \geq 0$
  **stop**; (* optimal point found *)
Select $q \in \mathcal{N}$ with $s_q < 0$ as the entering index;
Solve $Bd = A_q$ for $d$;
**if** $d \leq 0$
  **stop**; (* problem is unbounded *)
Calculate $x_q^+ = \min_{i \mid d_i > 0} (x_B)_i / d_i$, and use $p$ to denote the minimizing $i$;
Update $x_B^+ = x_B - dx_q^+, x_N^+ = (0, \ldots, 0, x_q^+, 0, \ldots, 0)^T$;
Change $\mathcal{B}$ by adding $q$ and removing the basic variable corresponding to column $p$ of $B$.

Linear algebra issues : Every simplex step requires solving three linear systems




$$B x_B = b \quad , \quad B^T\lambda = c_B \quad , \quad B d = A_q$$

(From numerical linear algebra:) It takes $O(m^3)$ time to compute $B^{-1}$ or to compute $PBQ = LU$ ( LU factorization with pivoting ).

The latter enjoys a smaller constant, and sometimes substantially cheaper by exploiting sparsity pattern in the coefficient matrix $B$.

Basic idea : compute the LU factorization of $B$ once (not 3 times!), $PBQ = LU$ reuse it for the 3 linear systems :

$$B x_B = b \iff LUQ^T x_B = P^T b \iff \boxed{Ly = P^T b, \; U\tilde{y} = y, \; x_B = Q\tilde{y}}$$

$$B^T\lambda = c_B \iff U^T L^T P\lambda = Q^T c_B \iff \boxed{U^T y = Q^T c_B, \; L^T\tilde{y} = y, \; \lambda = P^T\tilde{y}}$$

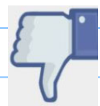
$$B d = A_q \iff LUQ^T d = P^T A_q \iff \boxed{Ly = P^T A_q, \; U\tilde{y} = y, \; d = Q\tilde{y}}$$

( $P, Q$ are permutation matrices.
We do <u>not</u> store them or multiply them to $m$-vectors as $m \times m$ matrices.)

solved using forward and backward substitutions, only $O(m^2)$ time.

Ex: In the "pricing" step, it would be bad if we compute $s_N$ by

$$s_N = c_N - (N^T B^{-T})\, c_B \quad \text{instead of} \quad s_N = c_N - N^T \underbrace{(B^{-T} c_B)}_{\lambda}.$$



Why?

---

More linear algebra issues:

Note that if we have to compute a LU factorization in every simplex step, the method is rather expensive for large $m$.

takes $O(m^3)$ time

Fortunately, since from one simplex step to the next the basis matrix $B$ only changes in one column, it is possible to "update the LU factorization" at much lower cost (rather than compute a LU factorization from scratch.)

Here is how it works:

If $\mathcal{B} = \{\cdots q \cdots\}$ becomes $\tilde{\mathcal{B}} = \{\cdots p \cdots\}$ in a simplex step, then the corresponding basis matrix changes in one column

$$\tilde{B} = [\cdots A_p \cdots] = [\cdots\cdots A_q \cdots] + [0 \cdots 0 \; A_p - A_q \; 0 \cdots 0]$$

$$= \underset{m\times m}{B} + \underset{\substack{m\times 1 \\ \parallel \\ R}}{(A_p - A_q)} \underset{\substack{1\times m \\ \parallel \\ S^T}}{[0 \cdots 1 \cdots 0]}$$

As such, $\tilde{B}$ is rank-1 correction of $B$ ($RS^T$ is a rank 1 matrix.)

Similarly, after $k$ simplex steps, the basis matrix is a rank $k$ correction of $B$.

$B^+ = B + RS^T$, $R \in \mathbb{R}^{m\times k}$, each column is the difference between the entering and leaving column of the basis

$S \in \mathbb{R}^{m\times k}$, each column is a unit vector representing the location of the column being updated

By the Sherman-Morrison-Woodbury formula,

$$(B^+)^{-1} = B^{-1} - B^{-1} R \underbrace{(I + S^T B^{-1} R)^{-1}}_{k\times k} S^T B^{-1}$$

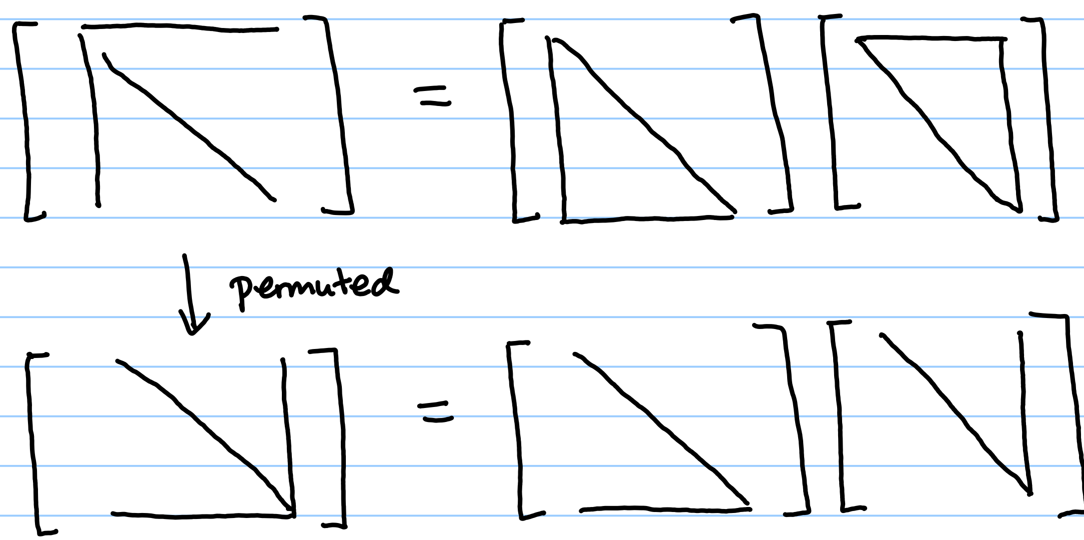
(Ex: Just multiply the R.H.S. to $B^+$ and check that the identity $I$ is recovered.)

Then we can solve $B^+ x = y$ or $(B^+)^T x = y$ by reusing the $L, U$ factors of $B$ and inverting a small $k\times k$ system.

This takes $O(\overset{\text{big}}{m^2}) + O(\overset{\text{small}}{k^3})$ operations, which is much smaller than $O(m^3)$.

But, when $k$ grows too large, this approach is guaranteed to get expensive and may get numerically unstable. Therefore, most simplex implementations periodically compute a fresh LU factorization of the current basis matrix $B$ and discard the accumulated updates.

Final remark: LU factorization for sparse matrices is a whole subject (within numerical linear algebra) on its own.




a sparse matrix with very dense L,U factors.

the same sparse matrix, suitably reordered, has very sparse L,U factors.

For the interest of time, I skip the sections of

- pricing and selection of the entering index

  Dantzig's original rule is to choose the most negative component of $s_N$.

- degenerate steps and cycling

  Cycling can indeed happen at a degenerate vertex.

  And there are clever anti-cycling rules that are incorporated into pratical simplex codes.

But I'll now show you:

- how to find a starting vertex / basis in general (Phase I procedure)

- how to avoid Phase I in special cases using the dual simplex method.

Here is a special case for which no Phase I is needed.

Consider a LP of the form $\min_x c^T x$ st $Ax \le b$, $x \ge 0$. —(*)

If $b \ge 0$, then $x=0$ is clearly a feasible point, and in fact a vertex of $\Lambda = \{x \in \mathbb{R}^n : Ax \le b, x \ge 0\}$.

When converted to the standard form $\min_x \begin{bmatrix} c^T & 0^T \end{bmatrix} \begin{bmatrix} x \\ x_s \end{bmatrix}$ s.t. $\begin{bmatrix} A & I \end{bmatrix} \begin{bmatrix} x \\ x_s \end{bmatrix} = b$, $\begin{bmatrix} x \\ x_s \end{bmatrix} \ge 0$

The origin of $\mathbb{R}^n$, as a vertex of $\Lambda \subseteq \mathbb{R}^n$, corresponds to the vertex/basis:

$$\begin{bmatrix} 0 \\ b \end{bmatrix} \text{ of } \Omega := \left\{ \begin{bmatrix} x \\ x_s \end{bmatrix} \in \mathbb{R}^{n+m} : \begin{bmatrix} A & I \end{bmatrix} \begin{bmatrix} x \\ x_s \end{bmatrix} = b, \begin{bmatrix} x \\ x_s \end{bmatrix} \ge 0 \right\}, \quad \begin{matrix} \mathcal{B} = \{n+1, \cdots, n+m\} \\ \mathcal{N} = \{1, \cdots, n\}. \end{matrix}$$

(Our earlier numerical example is of the form (*) with $b \ge 0$, that's why there's an obvious choice of a starting vertex and Phase I is not needed.)

## Phase I

To find a vertex / basis of $\Omega = \{x \mid Ax=b,\ x\geq 0\}$, consider the associated "Phase I problem"

$$\min_{z\in\mathbb{R}^m} e^T z \quad \text{s.t.} \quad Ax + Ez = b, \quad \begin{bmatrix} x \\ z \end{bmatrix} \geq 0$$

$$e = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}, \quad E = \begin{bmatrix} E_{11} & & 0 \\ & \ddots & \\ 0 & & E_{mm} \end{bmatrix}, \quad E_{ii} = \begin{cases} +1 & \text{if } b_j \geq 0 \\ -1 & \text{if } b_j < 0. \end{cases}$$

WTH!? To solve a LP that we don't know how to start solving, we solve yet another LP!?

Note:

1. $x=0$, $z_j = |b_j|$, $j=1,\cdots,m$, is a basic feasible pt. for this LP, with $E$ as the corresponding basis matrix.

2. This LP is never unbounded, as $\overset{\geq 0}{\hat{e}^T}\overset{\geq 0}{\hat{z}} \geq 0$.

3. If $\bar{x}$ is a vertex of $\Omega$, then $\begin{bmatrix} \bar{x} \\ 0 \end{bmatrix}$ is a minimizer of Phase I LP, with minimum value $0$.

   Conversely, if the minimum value is $0$, then the minimizer $\begin{bmatrix} \tilde{x} \\ \tilde{z} \end{bmatrix}$ must be such that $e^T\tilde{z} = 0$, But this implies $\tilde{z}=0$ and $A\tilde{x}=b$.

   This also means if the min. value of the Phase I LP is $>0$, then $\Omega = \emptyset$.

1 - 3 means we can solve the phase I LP using the version of the simplex method

we already know. If the min. value is found to be $> 0$, the original LP is infeasible. Otherwise, the min value is $0$, and we found a feasible point $\tilde{x}$ of $\Omega$.

A technicality here is that this $\tilde{x}$ may not be a basic feasible pt of $\Omega$, we are only guaranteed that $\begin{bmatrix} \tilde{x} \\ 0 \end{bmatrix}$ is a basic feasible point of $\hat{\Omega} := \{ \begin{bmatrix} x \\ z \end{bmatrix} : [A \;\; E] \begin{bmatrix} x \\ z \end{bmatrix} = b \}$, i.e. the basis $\hat{B}$ of $\hat{\Omega}$ may contain indices from the artificial variables $z$.

But this can be easily fixed: simply throw away from $\hat{B}$ any components of $z$, replace them with non-basic components of $x$ in a way that maintains non-singularity of the basis matrix $B$.

[This isn't quite what the 2nd edition of the book says.]

This gives a basis $B$ of $\Omega$, and we can then use it as the starting vertex of "Phase II".

Ex: If the LP is of the form $\min c^T x$ s.t. $Ax \le b$, $x \ge 0$, at least one entry of $b$ is negative. What do we do?

$$\begin{array}{l} \min c^T x \\ \text{s.t. } Ax \le b \\ \quad\;\; x \ge 0 \end{array} \xrightarrow{\text{"Standardize"}} \overset{(*)}{\begin{array}{l} \min c^T x \\ \text{s.t. } [A, I] \begin{bmatrix} x \\ x_s \end{bmatrix} = b \\ \quad\;\; x, x_s \ge 0 \end{array}} \xrightarrow{\text{Phase I}} \begin{array}{l} \min e^T z \\ \text{s.t. } [A, I, E] \begin{bmatrix} x \\ x_s \\ z \end{bmatrix} = b \\ \quad\;\; x, x_s, z \ge 0 \end{array}$$

solve this Phase I problem to get a basis $B$ for $(*)$.

Alternatively, consider the following Phase I problem (tailored for inequality constraints):

$$\begin{array}{l} \min\ c^T x \\ \text{s.t. } Ax \le b \\ \quad\ x \ge 0 \end{array} \longrightarrow \begin{array}{l} \min\ x_0 \\ \text{s.t. } Ax - \epsilon x_0 \le b \\ \quad\ x, x_0 \ge 0 \end{array} \xrightarrow{\text{Standardize}} \begin{array}{l} \min\ x_0 \\ \text{s.t. } [A\ \ -\epsilon\ \ I] \begin{bmatrix} x \\ x_0 \\ x_s \end{bmatrix} = b \\ \quad\ x, x_0, x_s \ge 0 \end{array}$$

Solve this Phase I problem to get a basis B for (*).

$$\epsilon_i = \begin{cases} 1 & \text{if } b_i < 0 \\ 0 & \text{if } b_i \ge 0 \end{cases}$$

Phase I (note: only 1 artificial variable needed.)

HW: Explain why and how this procedure would produce a basis for (*).

Explain why this approach is more efficient than the former approach.

Diet Problem: $m$ nutritional categories, indexed by $i=1,\cdots,m$
$n$ possible foods, indexed by $j=1,\cdots,n$

$x_j$ = # of units of food $j$ to be included in the diet

$c_j$ = cost of 1 unit of food $j$

$b_i$ = minimum daily requirement of nutrient $i$

$A_{ij}$ = amount of nutrient $i$ contained in one unit of food $j$

If one seeks the diet with lowest cost that achieves all the nutritional requirements, she is faced with following LP:

$$\min \ c_1x_1+\cdots+c_nx_n \quad \text{s.t.} \quad \begin{aligned} A_{11}x_1+\cdots+A_{1n}x_n &\geq b_1 \\ &\vdots \\ A_{m1}x_1+\cdots+A_{mn}x_n &\geq b_m \\ x_1,\cdots,x_n &\geq 0 \end{aligned}$$

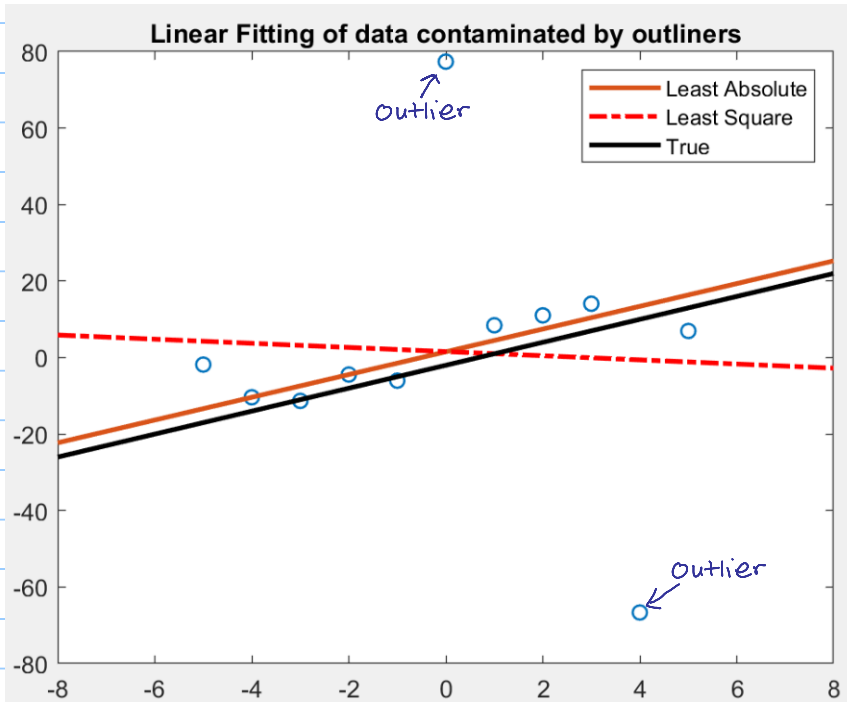
or $$\min_{x\in\mathbb{R}^n} c^Tx \quad \text{s.t.} \ \underbrace{Ax \geq b}, \ x \geq 0.$$

cannot start at $x=0$ / $B=\{n+1,\cdots,n+m\}$ ! What to do? use Phase I?

$L^1$- regression problem

$$\min_{x\in\mathbb{R}^d} \|\underset{\substack{m\times d \\ m>>d}}{A}x-b\|_1 \longrightarrow \begin{array}{l}\min\ \gamma_1+\cdots+\gamma_m \\ \text{s.t.}\ -\gamma\le Ax-b\le\gamma \\ \qquad\quad \gamma\ge 0\end{array} \longrightarrow \begin{array}{ll}\min & [0\cdots 0, 1,\cdots,1][x_1,\cdots,x_d,\gamma_1,\cdots,\gamma_m]^T \\ \text{s.t.} & Ax-\gamma\le b \\ & -Ax-\gamma\le -b\end{array}$$

$$\xrightarrow{x=x^+-x^-} \begin{array}{l}\min\ \mathcal{C}^T\mathcal{X} \\ \text{s.t.}\ \ \mathcal{A}\mathcal{X}\le\mathcal{B} \\ \qquad\quad \mathcal{X}\ge 0\end{array} \qquad \mathcal{C}=\begin{bmatrix}0\\ \vdots\\ 0\\ 1\\ \vdots\\ 1\end{bmatrix}\begin{matrix}\}2d\\ \\ \}m\end{matrix},\quad \mathcal{X}=\begin{bmatrix}x^+\\ x^-\\ \gamma\end{bmatrix},\quad \mathcal{A}=\begin{bmatrix}A & -A & -I\\ -A & A & -I\end{bmatrix},\quad \mathcal{B}=\underbrace{\begin{bmatrix}b\\ -b\end{bmatrix}}.$$




But there is no way this $\mathcal{B}$ is $\ge 0$
(well, unless when $b=0$, in which case the problem is meaningless.)

Cannot start at $\mathcal{X}=0$!

What to do? use Phase I?

In each case, note that while the $b$ vector in the constraints $Ax \leq b$ does not satisfy $b \geq 0$, the cost vector $c$ in the objective $c^T x$ satisfies $c \geq 0$.

What we can do is to apply the simplex method to the dual LP as a way to solve the primal LP, this would free us from the need of an expensive Phase I procedure.

Details to be presented next.

## Recap and clarifications

Standard form: $\min c^Tx$ s.t. $Ax=b,\ x\geq 0 \quad \xleftrightarrow{\text{dual}} \quad \max b^T\lambda$ s.t. $A^T\lambda\leq c,\ \lambda$-free

Canonical form: $\min c^Tx$ s.t. $Ax\geq b,\ x\geq 0 \quad \xleftrightarrow{\text{dual}} \quad \max b^T\lambda$ s.t. $A^T\lambda\leq c,\ \lambda\geq 0$

$\downarrow$ Standardize $\qquad\qquad\qquad\qquad$ $\downarrow$ Standardize

$$\max \begin{bmatrix} c \\ 0 \end{bmatrix}^T \begin{bmatrix} x \\ h \end{bmatrix} \text{ s.t. } [A\ {-I_{m\times m}}]\begin{bmatrix} x \\ h \end{bmatrix} = b,\quad x,h\geq 0 \quad \xleftrightarrow{\text{dual}} \quad \max \begin{bmatrix} 0 \\ b \end{bmatrix}^T \begin{bmatrix} s \\ \lambda \end{bmatrix} \text{ s.t. } [I_{n\times n}\ A^T]\begin{bmatrix} s \\ \lambda \end{bmatrix} = c,\quad s,\lambda\geq 0$$

### KKT of the canonical primal-dual pair:

$\mathcal{L}(x,\lambda,s) = c^Tx - \lambda^T(Ax-b) - s^Tx$

$\nabla_x\mathcal{L}(x,\lambda,s) = c - A^T\lambda - s \rightsquigarrow =0$

(1) $s = c - A^T\lambda$ $\qquad$ (2) $\lambda\geq 0,\ s\geq 0$

(3) $x\geq 0,\ Ax\geq b$ $\qquad$ (4) $\lambda^T(Ax-b)=0,\ s^Tx=0$

$\bar{\mathcal{L}}(\lambda,x,h) = -b^T\lambda - x^T(c-A^T\lambda) - h^T\lambda$

$\nabla_\lambda\bar{\mathcal{L}}(\lambda,x,h) = -b + Ax - h = 0 \rightsquigarrow =0$

(1) $h = Ax - b$ $\qquad$ (2) $x\geq 0,\ h\geq 0$

(3) $\lambda\geq 0,\ A^T\lambda\leq c$ $\qquad$ (4) $x^T(c-A^T\lambda)=0,\ h^T\lambda=0$

$\Downarrow \qquad \Updownarrow$

$$\boxed{\begin{array}{lcccl} 0\leq & x & \perp & c-A^T\lambda & \geq 0 \\ 0\leq & Ax-b & \perp & \lambda & \geq 0 \end{array}}$$

At optimality:

$0\leq$ primal vars $\perp$ dual slacks $\geq 0$

$0\leq$ primal slacks $\perp$ dual vars $\geq 0$

- The canonical form is easier to visualize geometrically (for me, at least.)
- The simplex method works best with the standard form.
- When it comes to duality, the canonical form is nicer because the primal-dual pair and KKT conditions look more symmetrical.
- But of course the simplex method can be easily applied to a canonical form LP.

**Procedure 13.1** (One Step of Simplex).

Given $\mathcal{B}, \mathcal{N}, x_B = B^{-1}b \geq 0, x_N = 0$;
Solve $B^T\lambda = c_B$ for $\lambda$,
Compute $s_N = c_N - N^T\lambda$; (* pricing *)
**if** $s_N \geq 0$
  **stop**; (* optimal point found *)
Select $q \in \mathcal{N}$ with $s_q < 0$ as the entering index;
Solve $Bd = A_q$ for $d$;
**if** $d \leq 0$
  **stop**; (* problem is unbounded *)
Calculate $x_q^+ = \min_{i \mid d_i > 0} (x_B)_i / d_i$, and use $p$ to denote the minimizing $i$;
Update $x_B^+ = x_B - dx_q^+$, $x_N^+ = (0, \ldots, 0, x_q^+, 0, \ldots, 0)^T$;
Change $\mathcal{B}$ by adding $q$ and removing the basic variable corresponding to column $p$ of $B$.

$x = \begin{bmatrix} x \\ h \end{bmatrix} \begin{matrix} \nearrow x_B \\ \searrow x_N = 0 \end{matrix}$

$\mathcal{A} = [A, I_{m \times m}]$

the $A$ in a canonical form LP

the '$\mathcal{A}$' in the corresponding standard form LP.

$\boxed{\text{Input}: \mathcal{B}, \mathcal{A}, b}$

$\boxed{\text{Output}: \mathcal{B}, x_B, \lambda, s_N}$

Solves the primal

Solves the dual

the $\mathcal{N}$-components of the dual slack variables $S = C - A^T\lambda$ at optimality ($S_B = 0$)

(we don't really need to output it, it's probably not needed in practice.)

- We may turn this around and apply the same algorithm to the dual as a way to solve the primal. This avoids Phase I in the case when $c \geq 0$.

This is called the dual simplex method.

The dual simplex method solves the (primal) LP in a totally different way: it produces iterates that approach an optimal vertex from the exterior of the feasible region.

It is because when the simplex method is applied to the primal, we have:

$$
\begin{array}{r|c|c|l}
 & x_N & 1 & \\
\hline
x_B = & -B^{-1}N & B^{-1}b & \geq 0 \\
\hline
c^Tx = & (c_N - N^T B^{-T} c_B)^T & c_B^T B^{-1} b & \\
\hline
\end{array}
$$

$\leftarrow$ meaning $x$ is always primal feasible

has negative entries before optimality, meaning $s_N \ngeq 0$, meaning $\lambda$ is not dual feasible

The situation is reversed in the dual simplex method.