Solving Linear and Integer Programs

Robert E. Bixby

Gurobi, Inc. and Rice University

Dual Simplex Algorithm

Some Motivation

- Dual simplex vs. primal (2002): Dual 2.7x faster
- Best algorithm of MIP
- There isn't much in books about implementing the dual.

Dual Simplex Algorithm

(Lemke, 1954: Commercial codes ~1990)

Input: A dual feasible basis B and vectors

$$X_B = A_B^{-1} b \quad \text{and} \quad \mathrm{D_N} = \mathrm{c_N} - \mathrm{A_N}^{\mathrm{T}} \mathrm{B}^{-\mathrm{T}} \mathrm{c_B}.$$

- **Step 1:** (Pricing) If $\mathrm{X_B} \geq 0$, stop, B is optimal; else let

  $$i = argmin\{X_{Bk} : k \in \{1,\ldots,m\}\}.$$

- **Step 2:** (BTRAN) Solve $B^T z = e_i$. Compute $\alpha_N = -A_N^T z$.
- **Step 3:** (Ratio test) If $\alpha_N \leq 0$, stop, (D) is unbounded; else, let

  $$j = argmin\{D_k / \alpha_k : \alpha_k > 0\}.$$

- **Step 4:** (FTRAN) Solve $A_B y = A_j$.
- **Step 5:** (Update) Set $B_i = j$. Update X_B (using y) and D_N (using α_N)

Dual Simplex Algorithm
(Lemke, 1954: Commercial codes ~1990)

Input: A dual feasible basis B and vectors

$$X_B = A_B^{-1} b \quad \text{and} \quad \mathrm{D_N = c_N - A_N{}^T B^{-T} c_B}.$$

- **Step 1:** (Pricing) If $\mathrm{X_B} \geq 0$, stop, B is optimal; else let

$$i = argmin\{X_{Bk} : k \in \{1,\ldots,m\}\}.$$

- **Step 2:** (BTRAN) Solve $B^T z = e_i$. Compute $\alpha_N = -A_N{}^T z$.
- **Step 3:** (Ratio test) If $\alpha_N \leq 0$, stop, (D) is unbounded; else, let

$$j = argmin\{D_k / \alpha_k : \alpha_k > 0\}.$$

- **Step 4:** (FTRAN) Solve $A_B y = A_j$.
- **Step 5:** (Update) Set $B_i = j$. Update X_B (using y) and D_N (using α_N)

Implementing the Dual Simplex Algorithm

Implementation Issues for Dual Simplex

1. **Finding an initial feasible basis, or the concluding that there is none**
2. **Pricing:** Dual steepest edge
3. **Solving the linear systems**
   - LU factorization and factorization update
   - BTRAN and FTRAN – exploiting sparsity
4. **Numerically stable ratio test:** Bound shifting and perturbation
5. **Bound flipping:** Exploiting “boxed” variables to combine many iterations into one.

Issue 0
Preparation: Bounds on Variables

In practice, simplex algorithms need to accept LPs in the following form:

$$\begin{aligned} &\textit{Minimize} && c^T x \\ &\textit{Subject to} && Ax = b \\ &&& l \le x \le u \end{aligned} \qquad (\mathrm{P}_{\mathrm{BD}})$$

where l is an n-vector of **lower bounds** and u an n-vector of **upper bounds**. In general, l is allowed to have $-\infty$ entries and u is allowed to have $+\infty$ entries. (Note that (P_{BD}) is in standard form if $l_j = 0, u_j = +\infty \; \forall \; j$.) Assuming all upper and lower bounds are finite, the corresponding dual is:

$$\begin{aligned} &\textit{Maximize} && b^T\pi + l^T r - u^T s \\ &\textit{Subject to} && A^T\pi + r - s = c \\ &&& \pi \textit{ free}, r \ge 0, s \ge 0 \end{aligned} \qquad (\mathrm{D}_{\mathrm{BD}})$$

(Issue 0 – Bounds on variables)
Basic Solution

A **basis** for (P_{BD}) is a triple (B,L,U) where B is an ordered m-element subset of $\{1,\ldots,n\}$ (as before), (B,L,U) is a partition of $\{1,\ldots,n\}$, $l_j > -\infty \ \forall\ j \in L$, and $u_j < +\infty \ \forall\ j \in U$. $N = L \cup U$ is the set of **nonbasic** variables. The associated **primal basic solution** X is given by $X_L = l_L$, $X_U = u_U$ and

$$X_B = A_B^{-1}(b - A_L l_L - A_U u_U).$$

This solution is **primal feasible** if

$$l_B \leq X_B \leq u_B.$$

The associated **dual basic variables** are π, r_L, and s_U with values: $\Pi = A_B^{-T} c_B$, $R_L = c_L - A_L^T \Pi$, and $S_U = -c_U + A_U^T \Pi$. It is **dual feasible** if

$$R_L \geq 0 \ \ and \ \ S_U \geq 0.$$

(Issue 0 – Bounds on variables)
The Full Story

- **Modify simplex algorithm**
  - Only the "Pricing" and "Ratio Test" steps must be changed substantially
  - The complicated part is the ratio test
- **Reference:** See Chvátal for the primal

Issue 1
The Initial Feasible Basis – Phase I

- **Two parts to the solution**
  1. Finding some initial basis (probably not feasible)
  2. Modified simplex algorithm to find a feasible basis

Reference for Primal: **R.E. Bixby (1992). “Implementing the simplex method: the initial basis”, *ORSA Journal on Computing* 4, 267—284.**

(Issue 1 – Initial feasible basis)
Initial Basis

- Primal and dual bases are the same. We begin in the context of the primal. Consider

$$\begin{aligned} \textit{Minimize} \quad & c^T x \\ \textit{Subject to} \quad & Ax = b \qquad\qquad (P_{BD}) \\ & l \le x \le u \end{aligned}$$

- **Assumption:** Every variable has some finite bound**.**
- **Trick:** Add **artificial variables** $x_{n+1},\ldots,x_{n+m}$:

$$Ax + I \begin{pmatrix} x_{n+1} \\ \cdot \\ \cdot \\ x_{n+m} \end{pmatrix} = b$$

where $l_j = u_j = 0$ for $j = n+1,\ldots,n+m$.

- **Initial basis:** $B = (n+1,\ldots,n+m)$ and for each $j \not\in B$, pick some finite bound and place j in L or U, as appropriate.

(Issue 1 – Initial feasible basis)
Solving the Phase I

- If the initial basis is not dual feasible, we consider the problem:

$$\textit{Maximize}\ \ \Sigma\,(d_j : d_j < 0)$$
$$\textit{Subject to}\ \ A^T\pi + d = c$$

- This problem is "locally linear": Define $\kappa \in \boldsymbol{R}^n$ by $\kappa_j = 1$ if $D_j < 0$, and 0 otherwise. Let

$$K = \{j: D_j < 0\}\ \ \textit{and}\ \ \underline{K} = \{j: D_j \geq 0\}$$

Then our problem becomes

$$\textit{Maximize}\ \ \kappa^T d$$
$$\textit{Subject to}\ \ A^T\pi + d = c$$
$$d_K \leq 0,\ d_{\underline{K}} \geq 0$$

- Apply dual simplex, and whenever d_j for $j \in K$ becomes 0, move it to $\underline{K}$.

(Issue 1 – Initial feasible basis)
Solving the Phase I – a Refinement

- Imagine performing the ratio test to determine which d_j will leave the basis given some d_{Bi} is entering:

  Case 1: $d_j < 0$ and hits 0   Case 2: $d_j > 0$ and hits 0

- Consider Case 2. Then a further increase in d_{Bi} will make dj < 0.
  - This can be handled by updating K. But is it desirable?
  - Update formula for "reduced cost":

    $$new\_reduced\_cost = old\_reduced\_cost \pm y_i$$

  - If the reduced cost does not change sign, we have a cheap update and can continue the step.
    - Note that this also improves numerical stability

Issue 2
Pricing

- The texbook rule: Choose the largest primal violation is **TERRIBLE**: For a problem in standard form

$$j = argmin\{X_{Bi} : i = 1,\ldots,m\}$$

- **Geometry is wrong:** Maximizes rate of change relative to axis; better to do relative to edge.
- Goldfard and Forrest 1992 suggested the following **steepest-edge** alternative

$$j = argmin\{X_{Bi}/\eta_i : i = 1,\ldots,m\}$$

where $\eta_i = \|e_i^T A_B^{-1}\|_2$, and gave an efficient update.

(Issue 2 – Pricing)
Dual Steepest Edge

- **Idea:** Compute the rate of change of the objective per unit movement along the "corresponding" edge of the polyhedron of feasible solutions.
- **Setup**
  - Assume the problem is in standard form with a dual basic feasible solution specified by a basis B.
  - d_{Bi} = entering variable
    - $X_{Bi} < 0$
    - $d_{Bi} = \theta > 0 \Rightarrow \Delta\text{objective} = -\theta X_{Bi} > 0$

(Issue 2 – Pricing)
Dual Steepest Edte

- Old solution vector:

$$d_B = 0 \qquad d_N = D_N \qquad \pi = \Pi$$

- New solution vector:

$$\underline{d}_B = \theta e_i \qquad \underline{d}_N = D_N - \theta \alpha_N \qquad \underline{\pi} = \Pi - \theta z$$

where $\alpha_N = -A_N^T z$ and $A_B^T z = e_i$.

- Hence the change in the solution vector for $\theta = 1$ is given by

$$\Delta d_B = e_i \qquad \Delta d_N = -\Delta \alpha_N \qquad \Delta \pi = z$$

(Issue 2 – Pricing)
Dual Steepest Edge

- Hence the change in the solution vector for $\theta = 1$ is given by

  $$\Delta d_B = e_i \quad \Delta d_N = -\Delta\alpha_N \quad \Delta\pi = z$$

  And so the rate of change of the objective per unit movement along the edge is given by

  $$x_{Bi} / sqrt(e_i^T e_i + \alpha_N^T \alpha_N + z^T z)$$

- Goldfarb and Forrest observation: Projection onto the space of the π variables gives equally good iteration counts and is much simpler to compute

  $$x_{Bi}/sqrt(z^T z) = x_{Bi}/\|e_i^T A_B^{-1}\|_2$$

Example: Pricing

Model: dfl001

Pricing: Greatest infeasibility

```
Dual simplex - Optimal:  Objective =    1.1266396047e+07
Solution time = 1339.86 sec.  Iterations = 771647 (0)
```

Pricing: Goldfarb-Forrest steepest-edge

```
Dual simplex - Optimal:  Objective =    1.1266396047e+07
Solution time =   24.48 sec.  Iterations = 18898 (0)
```

Issue 3
Solving FTRAN, BTRAN

- **Computing LU factorization:** See Suhl & Suhl (1990). "Computing sparse LU factorization for large-scale linear programming basis", ORSA Journal on Computing 2, 325-335.
- **Updating the Factorization:** Forrest-Tomlin update is the method of choice. See Chvátal Chapter 24.
- **Exploiting sparsity:** This is the main recent development.

(Issue 3 – Solving FTRAN & BTRAN)

We must solve two linear systems per iterstion:

FTRAN	BTRAN
$A_B y = A_j$	$A_B^T z = e_i$

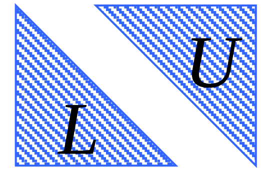
where

A_B	=	basis matrix	(very sparse)
A_j	=	entering column	(very sparse)
e_i	=	unit vector	(very sparse)

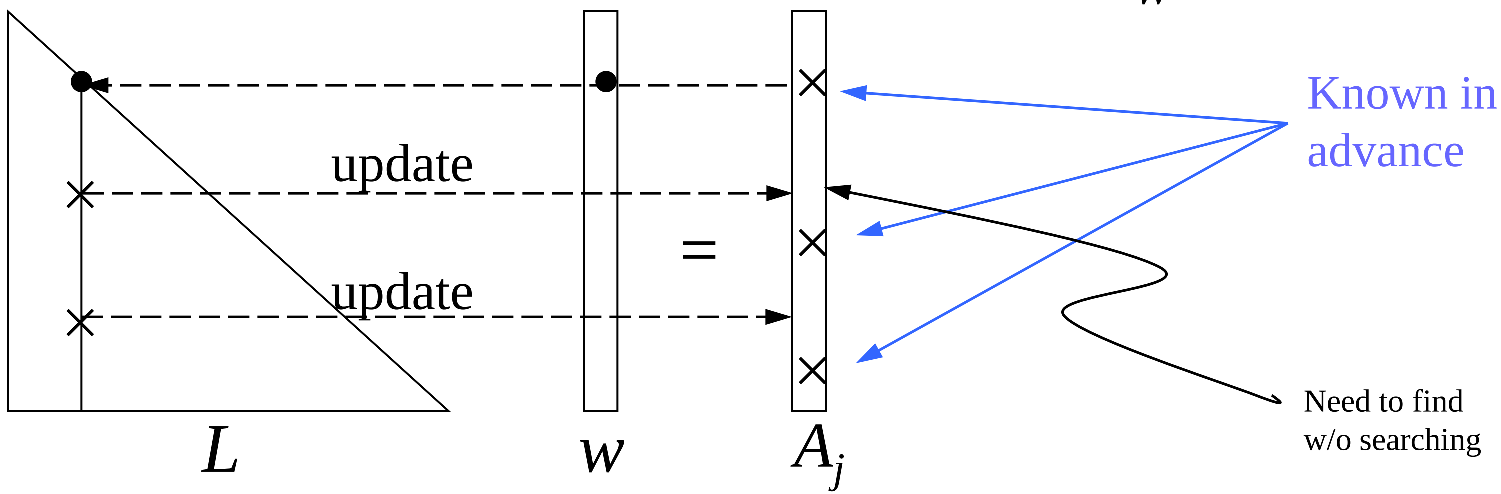
$\Rightarrow$ y an z are typically very sparse

Example: Model pla85900 (from TSP)

Constraints	85900
Variables	144185
Average |y|	15.5

$A_B =$ 

Triangular solve: $Lw = A_j \quad (A_B y = L(\underbrace{Uy}_{w}) = A_j)$



Graph structure: Define an acyclic digraph $D = (\{1,\ldots,m\}, E)$ where $(i,j) \in E \Leftrightarrow l_{ij} \neq 0$ and $i \neq j$.

Solving using *D*: Let $X = \{i \in V: A_{ij} \neq 0\}$. Compute

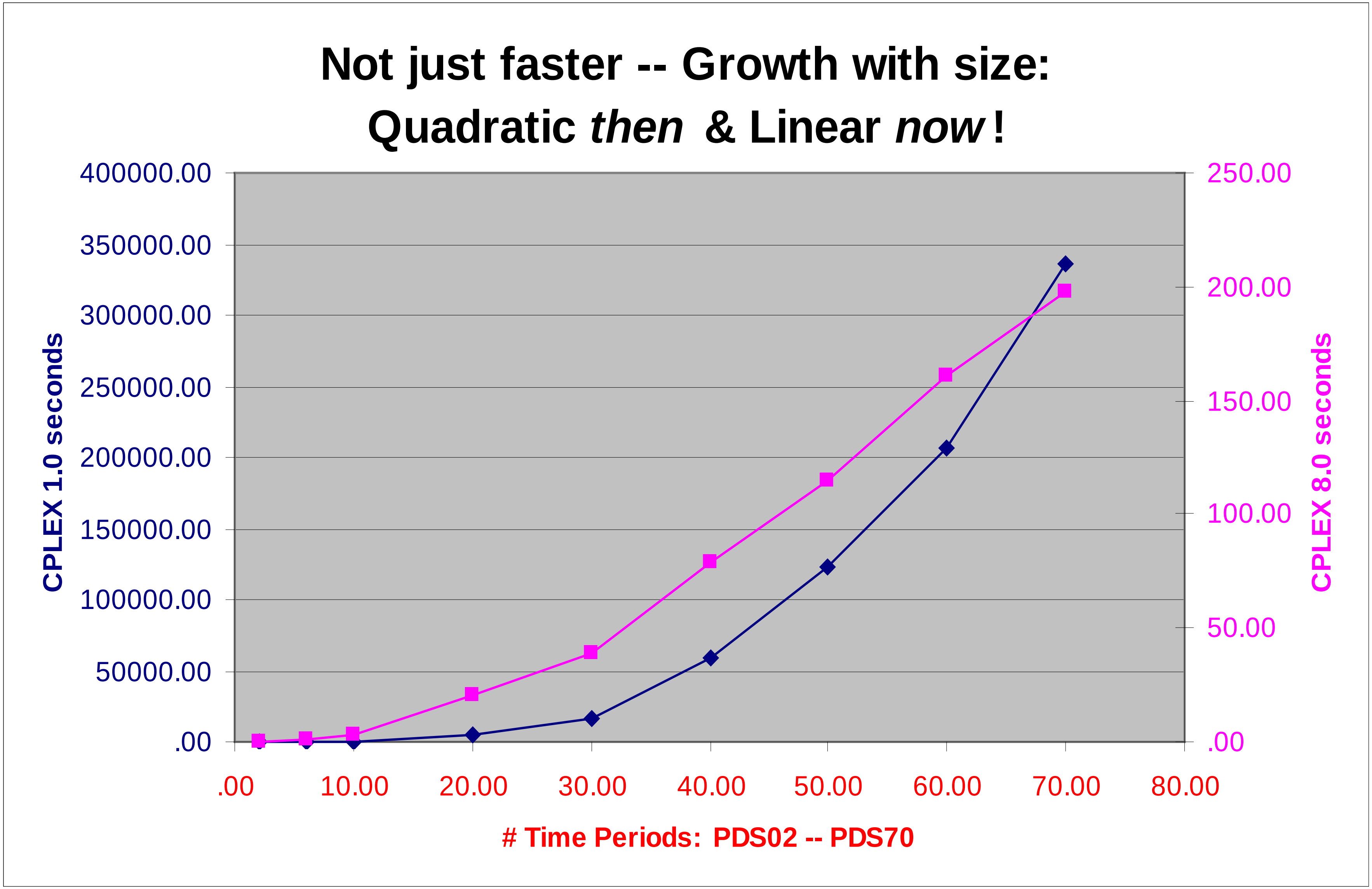
$$\underline{X} = \{j \in V: \exists \text{ a directed path from } j \text{ to } X\}.$$

$\underline{X}$ can be computed in time linear in $|E(\underline{X})| + |\underline{X}|$.

PDS Models

"Patient Distribution System": Carolan, Hill, Kennington, Niemi, Wichmann, *An empirical evaluation of the KORBX algorithms for military airlift applications*, Operations Research 38 (1990), pp. 240-248

MODEL	ROWS	CPLEX1.0 1988	CPLEX5.0 1997	CPLEX8.0 2002	SPEEDUP 1.0→8.0
pds02	2953	**0.4**	**0.1**	**0.1**	**4.0**
pds06	9881	**26.4**	**2.4**	**0.9**	**29.3**
pds10	16558	**208.9**	**13.0**	**2.6**	**80.3**
pds20	33874	**5268.8**	**232.6**	**20.9**	**247.3**
pds30	49944	**15891.9**	**1154.9**	**39.1**	**406.4**
pds40	66844	**58920.3**	**2816.8**	**79.3**	**743.0**
pds50	83060	**122195.9**	**8510.9**	**114.6**	**1066.3**
pds60	99431	**205798.3**	**7442.6**	**160.5**	**1282.2**
pds70	114944	**335292.1**	**21120.4**	**197.8**	**1695.1**
		Primal Simplex	Dual Simplex	Dual Simplex	


Not just faster -- Growth with size:
Quadratic *then* & Linear *now*!
400000.00
350000.00
300000.00
250000.00
200000.00
150000.00
100000.00
50000.00
.00
CPLEX 1.0 seconds
250.00
200.00
150.00
100.00
50.00
.00
CPLEX 8.0 seconds
.00
10.00
20.00
30.00
40.00
50.00
60.00
70.00
80.00
Time Periods: PDS02 -- PDS70

Issue 4
Ratio Test and Finiteness

The "standard form" dual problem is

$$\begin{aligned} &\textit{Maximize} \quad b^T\pi \\ &\textit{Subject to} \quad A^T\pi + d = c \\ &\qquad\qquad\qquad\qquad d \geq 0 \end{aligned}$$

Feasibility means

$$\boldsymbol{d \geq 0}$$

However, in practice this condition is replaced by

$$\boldsymbol{d \geq -\varepsilon e}$$

where $e^T=(1,\ldots,1)$ and $\varepsilon = 10^{-6}$. Reason: Degeneracy. In 1972 Paula Harris suggested exploiting this fact to improve numerical stability.

(Issue 4 – Ratio test & finiteness)

STD. RATIO TEST $j_{enter} = argmin\{D_j / \alpha_j : \alpha_j > 0\}$

Motivation: Feasibility $\Rightarrow$ step length θ satisfies

$$D_N - \theta\alpha_N \geq 0$$

However, the bigger the step length, the bigger the change in the objective. So, we choose

$$\theta_{max} = min\{D_j / \alpha_j : \alpha_j > 0\}$$

Using ε, we have

$$\theta^{\varepsilon}_{max} = min\{(D_j + \varepsilon)/\alpha_j : \alpha_j > 0\} > \theta_{max}$$

HARRIS RATIO TEST $j_{enter} = argmax\{\alpha_j : D_j / \alpha_j \leq \theta^{\varepsilon}_{max}\}$

(Issue 4 – Ratio test & finiteness)

- **Advantages**
  - Numerical stability – α_{jenter} = "pivot element"
  - Degeneracy – Reduces # of 0-length steps
- **Disadvantage**
  - $D_{jenter} < 0 \Rightarrow$ objective goes in wrong direction
- **Solution: BOUND SHIFTING**
  - If $D_{jenter} < 0$, we replace the lower bound on d_{jenter} by something less than its current value.
  - Note that this shift changes the problem and must be removed: 5% of cases, this produces dual infeasibility $\Rightarrow$ process is iterated.

Example: Bound-Shifting Removal

```
Problem 'pilot87.sav.gz' read.
Reduced LP has 1809 rows, 4414 columns, and 70191 nonzeros.

Iteration log . . .
Iteration:     1   Scaled dual infeas =             0.697540
Iteration:   733   Scaled dual infeas =             0.000404
Iteration:   790   Dual objective     =          -185.892207
...
Iteration: 16326   Dual objective     =           302.786794
Removing shift (3452).
Iteration: 16417   Scaled dual infeas =             0.207796
Iteration: 16711   Scaled dual infeas =             0.000021
Iteration: 16726   Dual objective     =           296.758656
Elapsed time =  104.36 sec. (17000 iterations).
Iteration: 17072   Dual objective     =           300.965492
...
Iteration: 17805   Dual objective     =           301.706409
Removing shift (76).
Iteration: 17919   Scaled dual infeas =             0.000060
Iteration: 17948   Dual objective     =           301.708660
Elapsed time =  114.42 sec. (18000 iterations).
Removing shift (10).
Iteration: 18029   Scaled dual infeas =             0.000050
Iteration: 18039   Dual objective     =           301.710058
Removing shift (1).

Dual simplex - Optimal:  Objective =   3.0171034733e+002
Solution time =  116.44 sec.  Iterations = 18095 (1137)
```

Shift 1: $\varepsilon = 10^{-7}$

Shift 2: $\varepsilon = 10^{-8}$

Shift 3: $\varepsilon = 10^{-9}$

```
Gurobi Optimizer version 2.0.0
Copyright (c) 2009, Gurobi Optimization, Inc.

Read MPS format model from file cont1.mps.bz2
cont1: 160792 Rows, 80795 Columns, 440387 NonZeros
Optimize a model with 160792 Rows, 80795 Columns and 440387 NonZeros

Presolve removed 40397 rows and 40397 columns
Presolve time: 0.31 sec.
Presolved: 120395 Rows, 40398 Columns, 359593 Nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0      handle free variables                          0s
   17434    1.6725221e-02   6.416129e+01   0.000000e+00      5s
   20749    1.6929624e-02   4.681255e+00   0.000000e+00     10s
   ...
   32371    2.2108293e-02   9.527316e+00   0.000000e+00    101s
   32953    2.2381550e-02   3.618798e+01   0.000000e+00    110s
   ...
   37997    2.5924066e-02   1.204414e+02   0.000000e+00    200s
   38579    2.6442899e-02   6.255491e+01   0.000000e+00    212s
   ...
   42853    3.0820162e-02   7.662419e+01   0.000000e+00    300s
   43400    3.1467196e-02   8.031314e+01   0.000000e+00    311s
   ...
   46184    3.4566856e-02   7.302474e+01   0.000000e+00    372s
   46822    3.6248845e-02   1.600513e-01   0.000000e+00    386s
   46994    3.6272914e-02   0.000000e+00   0.000000e+00    390s
   47222    1.4881820e-02   0.000000e+00   3.185893e+00    400s
   47415    1.4864227e-02   0.000000e+00   4.802191e+01    406s
   47830    1.4649598e-02   0.000000e+00   7.049439e-01    420s
   48267    1.4450227e-02   0.000000e+00   7.578008e+00    431s
   48815    1.2095665e-02   0.000000e+00   6.917880e-01    444s
   49144    1.0459973e-02   0.000000e+00   6.762116e-01    452s
   49241    8.7824861e-03   0.000000e+00   0.000000e+00    471s

Solved in 49241 iterations and 471.05 seconds
Optimal objective  8.782486112e-03
```

Switched to Primal

(Issue 4 – Ratio test & finiteness)

Finiteness: Bound shifting is closely related to the "perturbation" method employed if no progress is being made in the objective.

"No progress" $\Rightarrow$

$$d_j \geq -\varepsilon \qquad j = 1,\ldots,n$$

is replaced by

$$d_j \geq -\varepsilon - \varepsilon_j \qquad j = 1,\ldots,n,$$

where ε_j is random uniform on $[0,\varepsilon]$.

Implementation detail: For a basis B, we initially perturb only the bounds on the variables in d_N. Bound perturbations are then introduced for other d_j variables when j enters N for the first time.

Issue 5
Bound Flipping

- If the current basis is not optimal, then there is a B_i such that
  - Case 1: $X_{Bi} < l_{Bi}$, or
  - Case 2: $X_{Bi} > u_{Bi}$.
- **Consider Case 2 (Case 1 is similar).** Then the corresponding dual move is to consider increasing the dual non-basic variable s_{Bi} to some $\theta > 0$, leaving all other non-basics at 0. The resulting values of the basic variables are given by
  - $R_L^\theta = R_L - \theta\, \alpha_L \geq 0$
  - $S_U^\theta = S_U + \theta\, \alpha_U \geq 0$
- The maximum step length is then given by

$$\theta_{max} = min\{\theta_{max}{}^r, \theta_{max}{}^r\}$$

where

$$\theta_{max}{}^r = min\{ r_j/\alpha_j\colon \alpha_j > 0, j \in L\} \;\; and \;\; \theta_{max}{}^s = min\{-s_j/\alpha_j\colon \alpha_j < 0, j \in U\}$$

Issue 5
Bound Flipping

- Now suppose that $\theta_{max} = r_j/\alpha_j$, $\alpha_j > 0$. Then the normal simplex step would be to remove r_j from the basis and replace it by s_{Bi}.

- However, instead of doing this, we consider replacing r_j by s_j in the basis, which is possible if $u_j < +\infty$. Since r_j and s_j are dual slacks in the same constraint, and have opposite signs, r_j becoming negative translates to s_j becoming positive, and preserves dual feasibility.

- That is, we consider setting $L \leftarrow L \backslash \{j\}$ and $U \leftarrow U \cup \{j\}$. This is a good idea $\Leftrightarrow$ the $updated\_X_{Bi} > u_{Bi}$. But it is easy to show that

$$updated\_X_{Bi} = X_{Bi} + \alpha_j (l_j - u_j) < X_{Bi} \quad (\text{since } \alpha_j > 0).$$

- So it is easy to determine whether this “flipping” is desirable: If $updated\_X_{Bi} > u_{Bi}$. In this case we obtain a cheap basis update and can continue with the ratio test.

Example: Bound Flipping

w/o flipping

```
Problem 'fit2d.sav.gz' read.
Initializing dual steep norms . . .

Iteration log . . .
Iteration:     1   Dual objective     =        -80412.550000
Perturbation started.
Iteration:   203   Dual objective     =        -80412.550000
Iteration:  1313   Dual objective     =        -80412.548666
Iteration:  2372   Dual objective     =        -77028.548350
Iteration:  3413   Dual objective     =        -71980.245530
Iteration:  4316   Dual objective     =        -70657.605570
Iteration:  5151   Dual objective     =        -68994.477061
Iteration:  5820   Dual objective     =        -68472.659371
Removing perturbation.

Dual simplex - Optimal:  Objective =  -6.8464293294e+004
Solution time =   18.74 sec.  Iterations = 5932 (0)
```

w/ flipping

```
Problem 'fit2d.sav.gz' read.
Initializing dual steep norms . . .

Iteration log . . .
Iteration:     1   Dual objective     =        -77037.550000

Dual simplex - Optimal:  Objective =  -6.8464293294e+004
Solution time =    1.88 sec.  Iterations = 201 (0)
```