Novel update techniques for the revised simplex method (and their application)

Qi Huangfu[1] Julian Hall[2] Others

[1]FICO

[2]School of Mathematics, University of Edinburgh

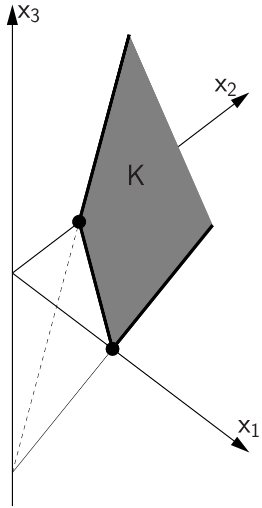
ERGO

30 November 2016

Overview

- Background
  - Dual vs primal simplex method
- Three COAP prizes
  - Hyper-sparsity in the revised simplex method (2005)
  - Data parallel simplex for massive stochastic LPs (2013)
  - Novel update techniques for the revised simplex method (2015)
- The future

Solving LP problems: Characterizing the feasible region

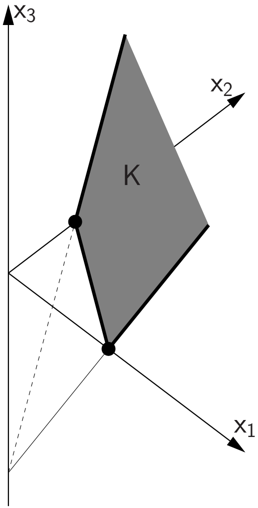


minimize $f = \boldsymbol{c}^T \boldsymbol{x}$ subject to $A\boldsymbol{x} = \boldsymbol{b}$ $\boldsymbol{x} \geq \boldsymbol{0}$

- $A \in \mathbb{R}^{m \times n}$ is full rank
- Solution of $A\boldsymbol{x} = \boldsymbol{b}$ is a $n - m$ dim. **hyperplane** in $\mathbb{R}^n$
- Intersection with $\boldsymbol{x} \geq \boldsymbol{0}$ is the **feasible region** K
- A vertex of K has
  - m **basic** components, $i \in \mathcal{B}$ given by $A\boldsymbol{x} = \boldsymbol{b}$
  - $n - m$ zero **nonbasic** components, $j \in \mathcal{N}$

  where $\mathcal{B} \cup \mathcal{N}$ partitions $\{1, \ldots, n\}$
- A solution of the LP occurs at a **vertex** of K

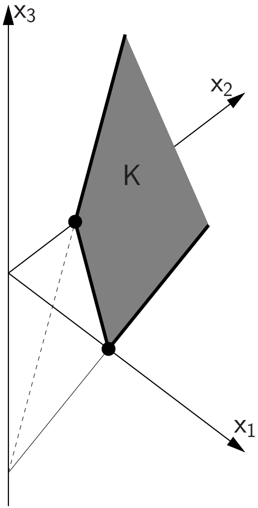
Solving LP problems: Characterizing the feasible region



minimize $\quad f = \boldsymbol{c}^T\boldsymbol{x} \qquad$ subject to $\quad A\boldsymbol{x} = \boldsymbol{b} \qquad \boldsymbol{x} \geq \boldsymbol{0}$

- Equations partitioned according to $\mathcal{B} \cup \mathcal{N}$ as
$$B\boldsymbol{x}_B + N\boldsymbol{x}_N = \boldsymbol{b}$$
with nonsingular **basis matrix** B
- Solution set characterized by $\boldsymbol{x}_B = B^{-1}(\boldsymbol{b} - N\boldsymbol{x}_N)$
- At a **vertex** of K, $\begin{bmatrix}\boldsymbol{x}_B\\ \boldsymbol{x}_N\end{bmatrix} = \begin{bmatrix}\widehat{\boldsymbol{b}}\\ \boldsymbol{0}\end{bmatrix}$ where $\widehat{\boldsymbol{b}} = B^{-1}\boldsymbol{b} \geq \boldsymbol{0}$
- K is $\boldsymbol{x}_B = \widehat{\boldsymbol{b}} - B^{-1}N\boldsymbol{x}_N$ for some $\boldsymbol{x}_N \geq \boldsymbol{0}$

Solving LP problems: Optimality conditions



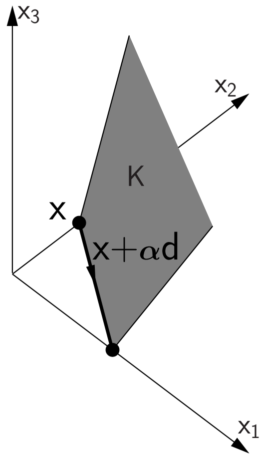
minimize $f = \boldsymbol{c}^T\boldsymbol{x}$ subject to $A\boldsymbol{x} = \boldsymbol{b}$ $\boldsymbol{x} \geq \boldsymbol{0}$

- Objective partitioned according to $\mathcal{B} \cup \mathcal{N}$ as

$$\begin{aligned} f &= \boldsymbol{c}_B^T\boldsymbol{x}_B + \boldsymbol{c}_N^T\boldsymbol{x}_N \\ &= \boldsymbol{c}_B^T(\widehat{\boldsymbol{b}} - B^{-1}N\boldsymbol{x}_N) + \boldsymbol{c}_N^T\boldsymbol{x}_N \\ &= \widehat{f} + \widehat{\boldsymbol{c}}_N^T\boldsymbol{x}_N \end{aligned}$$

  where $\widehat{f} = \boldsymbol{c}_B^T\widehat{\boldsymbol{b}}$ and $\widehat{\boldsymbol{c}}_N^T = \boldsymbol{c}_N^T - \boldsymbol{c}_B^T B^{-1}N$ is the vector of **reduced costs**
- Partition yields an optimal solution if there is
  - Primal feasibility $\widehat{\boldsymbol{b}} \geq \boldsymbol{0}$
  - Dual feasibility $\widehat{\boldsymbol{c}}_N \geq \boldsymbol{0}$

The simplex algorithm: Definition



At a feasible vertex $\boldsymbol{x} = \begin{bmatrix} \widehat{\boldsymbol{b}} \\ \boldsymbol{0} \end{bmatrix}$ corresponding to $\mathcal{B} \cup \mathcal{N}$

1. If $\widehat{\boldsymbol{c}}_N \geq \boldsymbol{0}$ then **stop: the solution is optimal**
2. Scan $\widehat{c}_j < 0$ for q to leave $\mathcal{N}$
3. Let $\widehat{\boldsymbol{a}}_q = B^{-1}N\boldsymbol{e}_q$ and $\boldsymbol{d} = \begin{bmatrix} -\widehat{\boldsymbol{a}}_q \\ \boldsymbol{e}_q \end{bmatrix}$
4. Scan $\widehat{b}_i / \widehat{a}_{iq}$ for α and p to leave $\mathcal{B}$
5. Exchange p and q between $\mathcal{B}$ and $\mathcal{N}$
6. Go to 1

The dual simplex algorithm

Solving dual LP problems: Characterizing the feasible region

- Consider the **dual problem**

$$\text{maximize} \quad f_D = \boldsymbol{b}^T \boldsymbol{y} \qquad \text{subject to} \quad A^T \boldsymbol{y} + \boldsymbol{s} = \boldsymbol{c} \qquad \boldsymbol{s} \geq \boldsymbol{0}$$

- For a partition $\mathcal{B} \cup \mathcal{N}$ of the equations with nonsingular **basis matrix** B

$$\begin{bmatrix} B^T \\ N^T \end{bmatrix} \boldsymbol{y} + \begin{bmatrix} \boldsymbol{s}_B \\ \boldsymbol{s}_N \end{bmatrix} = \begin{bmatrix} \boldsymbol{c}_B \\ \boldsymbol{c}_N \end{bmatrix}$$

- So $\quad \boldsymbol{y} = B^{-T}(\boldsymbol{c}_B - \boldsymbol{s}_B) \quad$ and $\quad \boldsymbol{s}_N = \boldsymbol{c}_N - N^T \boldsymbol{y} = \widehat{\boldsymbol{c}}_N + N^T B^{-T} \boldsymbol{s}_B$
- At a **vertex** of the dual feasible region K_D, $\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{s}_N \\ \boldsymbol{s}_B \end{bmatrix} = \begin{bmatrix} B^{-T} \boldsymbol{c}_B \\ \widehat{\boldsymbol{c}}_N \\ \boldsymbol{0} \end{bmatrix}$, where $\widehat{\boldsymbol{c}}_N \geq \boldsymbol{0}$
- Points in K_D are given by $\boldsymbol{s}_B \geq \boldsymbol{0}$

Solving dual LP problems: Optimality conditions

- Consider the **dual problem**

$$\text{maximize} \quad f_D = \boldsymbol{b}^T \boldsymbol{y} \qquad \text{subject to} \quad A^T \boldsymbol{y} + \boldsymbol{s} = \boldsymbol{c} \qquad \boldsymbol{s} \geq \boldsymbol{0}$$

- Substitute $\boldsymbol{y} = B^{-T}(\boldsymbol{c}_B - \boldsymbol{s}_B)$ into the objective to give

$$f_D \;=\; \boldsymbol{b}^T B^{-T}(\boldsymbol{c}_B - \boldsymbol{s}_B) \;=\; \widehat{f} - \widehat{\boldsymbol{b}}^T \boldsymbol{s}_B$$

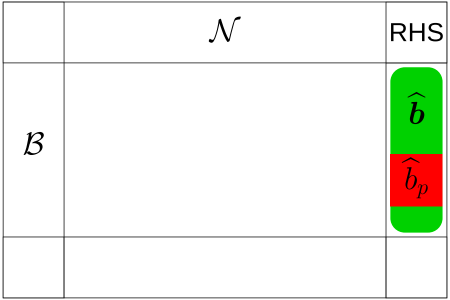
- Partition yields an optimal solution if there is
  - Dual feasibility $\widehat{\boldsymbol{c}}_N \geq \boldsymbol{0}$
  - Primal feasibility $\widehat{\boldsymbol{b}} \geq \boldsymbol{0}$
- **Dual simplex algorithm** for an LP is *primal algorithm* applied to the *dual problem*
- Structure of dual equations allows dual simplex algorithm to be applied to primal simplex tableau

Dual simplex algorithm

Assume $\widehat{\boldsymbol{c}}_N \geq \boldsymbol{0}$ Seek $\widehat{\boldsymbol{b}} \geq \boldsymbol{0}$

Scan $\widehat{b}_i < 0$ for p to leave $\mathcal{B}$

Scan $\widehat{c}_j / \widehat{a}_{pj}$ for q to leave $\mathcal{N}$

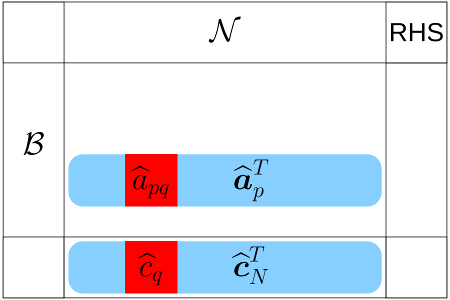


Dual simplex algorithm

Assume $\widehat{\boldsymbol{c}}_N \geq \mathbf{0}$ Seek $\widehat{\boldsymbol{b}} \geq \mathbf{0}$

Scan $\widehat{b}_i < 0$ for p to leave $\mathcal{B}$

Scan $\widehat{c}_j / \widehat{a}_{pj}$ for q to leave $\mathcal{N}$

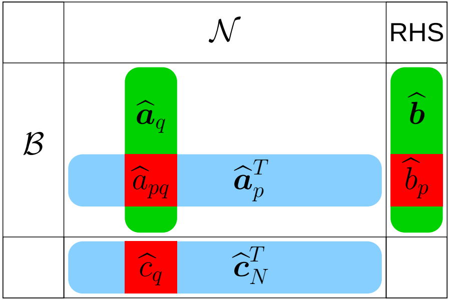


Dual simplex algorithm

Assume $\widehat{c}_N \geq \mathbf{0}$ Seek $\widehat{\boldsymbol{b}} \geq \mathbf{0}$

Scan $\widehat{b}_i < 0$ for p to leave $\mathcal{B}$

Scan $\widehat{c}_j / \widehat{a}_{pj}$ for q to leave $\mathcal{N}$



Update: Exchange p and q between $\mathcal{B}$ and $\mathcal{N}$

Update $\widehat{\boldsymbol{b}} := \widehat{\boldsymbol{b}} - \alpha_p \widehat{\boldsymbol{a}}_q$ $\qquad \alpha_p = \widehat{b}_p / \widehat{a}_{pq}$

Update $\widehat{\boldsymbol{c}}_N^T := \widehat{\boldsymbol{c}}_N^T - \alpha_d \widehat{\boldsymbol{a}}_p^T$ $\qquad \alpha_d = \widehat{c}_q / \widehat{a}_{pq}$

Data required

- Pivotal row $\widehat{\boldsymbol{a}}_p^T = \boldsymbol{e}_p^T B^{-1} N$
- Pivotal column $\widehat{\boldsymbol{a}}_q = B^{-1} \boldsymbol{a}_q$

Why does it work?

Objective improves by $-\dfrac{\widehat{b}_p \times \widehat{c}_q}{\widehat{a}_{pq}}$ each iteration

Simplex method: Computation

Standard simplex method (SSM): Major computational component

	$\mathcal{N}$	RHS
$\mathcal{B}$	$\widehat{N}$	$\widehat{\boldsymbol{b}}$
	$\widehat{\boldsymbol{c}}_N^T$	

Update of tableau: $\widehat{N} := \widehat{N} - \dfrac{1}{\widehat{a}_{pq}}\widehat{\boldsymbol{a}}_q\widehat{\boldsymbol{a}}_p^T$

where $\widehat{N} = B^{-1}N$

- Hopelessly inefficient for sparse LP problems
- Prohibitively expensive for large LP problems

Revised simplex method (RSM): Major computational components

Pivotal row via	$B^T\boldsymbol{\pi}_p = \boldsymbol{e}_p$	BTRAN	and	$\widehat{\boldsymbol{a}}_p^T = \boldsymbol{\pi}_p^T N$	PRICE
Pivotal column via	$B\,\widehat{\boldsymbol{a}}_q = \boldsymbol{a}_q$	FTRAN		Invert B	

Simplex method: Product form update (PFI) for B

Each iteration: Exchange p and q between $\mathcal{B}$ and $\mathcal{N}$

- Column p of B replaced by $\boldsymbol{a}_q$ to give $\bar{B} = B + (\boldsymbol{a}_q - B\boldsymbol{e}_p)\boldsymbol{e}_p^T$
- Take B out as a factor on the left

$$\bar{B} \quad = \quad B[I + (B^{-1}\boldsymbol{a}_q - \boldsymbol{e}_p)\boldsymbol{e}_p^T] \quad = \quad BE$$

$$\text{where} \quad E \quad = \quad I + (\widehat{\boldsymbol{a}}_q - \boldsymbol{e}_p)\boldsymbol{e}_p^T \quad = \quad \begin{bmatrix} 1 & & \eta_1 & & \\ & \ddots & \vdots & & \\ & & \mu & & \\ & & \vdots & \ddots & \\ & & \eta_m & & 1 \end{bmatrix}$$

$\mu = \widehat{a}_{pq}$ is the **pivot**; remaining entries in $\widehat{\boldsymbol{a}}_q$ form the **eta vector** $\boldsymbol{\eta}$

- Can solve $\bar{B}\boldsymbol{x} = \boldsymbol{r}$ as $B\boldsymbol{x} = \boldsymbol{r}$ then $\boldsymbol{x} := E^{-1}\boldsymbol{x}$ as

$$x_p := x_p/\mu \qquad \text{then} \qquad \boldsymbol{x} := \boldsymbol{x} - x_p\boldsymbol{\eta}$$

Dantzig and Orchard-Hays (1954)

Simplex method: Dual simplex algorithm enhancements

Row selection: Dual steepest edge (DSE)

- Weight $\widehat{b}_i$ by w_i: measure of $\|B^{-T}\boldsymbol{e}_i\|_2$
- Requires $B^{-1}\boldsymbol{\pi}_p$ (FTRAN-DSE) but can reduce iteration count significantly

Column selection: Bound-flipping ratio test (BFRT)

For general bounded LPs

$$\text{minimize} \quad f = \boldsymbol{c}^T\boldsymbol{x} \qquad \text{subject to} \quad A\boldsymbol{x} = \boldsymbol{b} \qquad \boldsymbol{l} \le \boldsymbol{x} \le \boldsymbol{u}$$

Dual feasibility: For $j \in \mathcal{N}$, $s_j \ge 0$ if $x_j = l_j$ but $s_j \le 0$ if $x_j = u_j$

- BFRT maximizes the dual objective whilst remaining dual feasible
- Dual variables may change sign if corresponding primal variables can flip bounds
- Requires $B^{-1}\boldsymbol{a}_F$ (FTRAN-BFRT) but can reduce iteration count significantly

Simplex algorithm: Primal or dual?

Primal simplex algorithm

- Traditional variant
- Solution generally not primal feasible when (primal) LP is tightened

Dual simplex algorithm

- Preferred variant
- Easier to get dual feasibility
- More progress in many iterations
- Solution dual feasible when primal LP is tightened

Parallelising the simplex method

History

- SSM: Good parallel efficiency of $\widehat{N} := \widehat{N} - \frac{1}{\widehat{a}_{pq}}\widehat{\boldsymbol{a}}_q\widehat{\boldsymbol{a}}_p^T$ *was* achieved
  Many! (1988–date)
- Only parallel *revised* simplex is worthwhile: goal of H and McKinnon in 1992!

Parallel primal revised simplex method

- Overlap computational components for different iterations
  Wunderling (1996), H and McKinnon (1995-2005)
- Modest speed-up *was* achieved on general sparse LP problems

Parallel dual revised simplex method

- Only immediate parallelism is in forming $\pi_p^T N$
- When $n \gg m$ significant speed-up *was* achieved Bixby and Martin (2000)

Exploiting hyper-sparsity (1998–2003)

Exploiting hyper-sparsity in the revised simplex method

Recall: major computational components

- BTRAN: Form $\boldsymbol{\pi}_p = B^{-T}\boldsymbol{e}_p$
- PRICE: Form $\widehat{\boldsymbol{a}}_p^T = \boldsymbol{\pi}_p^T N$
- FTRAN: Form $\widehat{\boldsymbol{a}}_q = B^{-1}\boldsymbol{a}_q$

Phenomenon of hyper-sparsity

- Vectors $\boldsymbol{e}_p$ and $\boldsymbol{a}_q$ are sparse
- Results $\boldsymbol{\pi}_p$, $\widehat{\boldsymbol{a}}_p^T$ and $\widehat{\boldsymbol{a}}_q$ may be sparse—because B^{-1} is sparse

Exploiting hyper-sparsity: When solving $Bx = \boldsymbol{r}$

Traditional technique transforms $\boldsymbol{r}$ into $\boldsymbol{x}$

$$\texttt{do } k = 1,\ K$$

$$r_{p_k} := r_{p_k} / \mu_k$$
$$\boldsymbol{r} := \boldsymbol{r} - r_{p_k} \boldsymbol{\eta}_k$$

$$\texttt{end do}$$

Exploiting hyper-sparsity: When solving $Bx = \boldsymbol{r}$

When $\boldsymbol{r}$ is sparse skip $\boldsymbol{\eta}_k$ if r_{p_k} is zero

```
do k = 1, K
    if (r_{p_k} .ne. 0) then
        r_{p_k} := r_{p_k}/μ_k
        r := r − r_{p_k} η_k
    end if
end do
```

- When $\boldsymbol{x}$ is sparse, the dominant cost is the test for zero
- Requires efficient identification of vectors $\boldsymbol{\eta}_k$ to be applied

Gilbert and Peierls (1988)
H and McKinnon (1998–2005)
COAP best paper prize (2005)

Data parallelism for stochastic LPs (2012–13)

Stochastic MIP problems: General

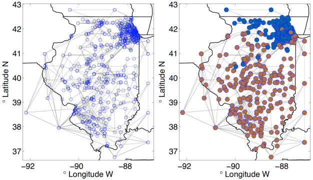
Two-stage stochastic LPs have column-linked block angular structure

$$
\begin{array}{lccccccccccc}
\text{minimize} & \mathbf{c}_0^T\mathbf{x}_0 & + & \mathbf{c}_1^T\mathbf{x}_1 & + & \mathbf{c}_2^T\mathbf{x}_2 & + & \dots & + & \mathbf{c}_N^T\mathbf{x}_N & & \\
\text{subject to} & A\mathbf{x}_0 & & & & & & & & & = & \mathbf{b}_0 \\
 & T_1\mathbf{x}_0 & + & W_1\mathbf{x}_1 & & & & & & & = & \mathbf{b}_1 \\
 & T_2\mathbf{x}_0 & & & + & W_2\mathbf{x}_2 & & & & & = & \mathbf{b}_2 \\
 & \vdots & & & & & & \ddots & & & & \vdots \\
 & T_N\mathbf{x}_0 & & & & & & & + & W_N\mathbf{x}_N & = & \mathbf{b}_N \\
 & \mathbf{x}_0 \geq \mathbf{0} & & \mathbf{x}_1 \geq \mathbf{0} & & \mathbf{x}_2 \geq \mathbf{0} & & \dots & & \mathbf{x}_N \geq \mathbf{0} & &
\end{array}
$$

- Variables $\mathbf{x}_0 \in \mathbb{R}^{n_0}$ are **first stage** decisions
- Variables $\mathbf{x}_i \in \mathbb{R}^{n_i}$ for $i = 1, \dots, N$ are **second stage** decisions
  Each corresponds to a **scenario** which occurs with modelled probability
- The objective is the expected cost of the decisions
- In stochastic MIP problems, some/all decisions are discrete

Stochastic MIP problems: For Argonne

- Power systems optimization project at Argonne
- Integer second-stage decisions
- Stochasticity from wind generation
- Solution via branch-and-bound
  - Solve root using parallel IPM solver `PIPS`
    Lubin, Petra *et al.* (2011)
  - Solve nodes using parallel dual simplex solver `PIPS-S`



Parallel distributed-memory simplex for large-scale stochastic LP problems

Scope for parallelism

- Parallel Gaussian elimination yields **block LU** decomposition of B
- Scope for parallelism in block forward and block backward substitution
- Scope for parallelism in `PRICE`

Implementation

- Distribute problem data over processes
- Perform data-parallel `BTRAN`, `FTRAN` and `PRICE` over processes
- Used MPI
- Solved LP with 500 million variables/constraints on 8192 Blue Gene cores

Lubin, H *et al.*

COAP best paper prize (2013)

Novel update techniques for the revised simplex method (2012–13)

Novel update techniques for the revised simplex method: 1

Alternative product form update (APF)

- **Recall:** Column p of B is replaced by $\boldsymbol{a}_q$ to give $\bar{B} = B + (\boldsymbol{a}_q - B\boldsymbol{e}_p)\boldsymbol{e}_p^T$
  - Traditional PFI (1954) takes B out as a factor on the left so $\bar{B} = BE$
- **Idea:** Why not take it out on the right!

$$\begin{aligned} \bar{B} &= [I + (\boldsymbol{a}_q - B\boldsymbol{e}_p)\boldsymbol{e}_p^T B^{-1}]B = TB \\ \text{where} \quad T &= I + (\boldsymbol{a}_q - \boldsymbol{a}_{p'})\widehat{\boldsymbol{e}}_p^T \end{aligned}$$

- T is formed of known data and readily invertible (like E for PFI)
- **But:** Is this useful?

Novel update techniques for the revised simplex method: 2

Middle product form update (MPF)

- **Recall:** Column p of B is replaced by $\boldsymbol{a}_q$ to give $\bar{B} = B + (\boldsymbol{a}_q - B\boldsymbol{e}_p)\boldsymbol{e}_p^T$
- **Idea:** Substitute $B = LU$ and take factors L on the left and U on the right!

$$\begin{aligned} \bar{B} &= LU + (\boldsymbol{a}_q - B\boldsymbol{e}_p)\boldsymbol{e}_p^T \\ &= LU + LL^{-1}(\boldsymbol{a}_q - B\boldsymbol{e}_p)\boldsymbol{e}_p^T U^{-1} U \\ &= L(I + (\tilde{\boldsymbol{a}}_q - U\boldsymbol{e}_p)\tilde{\boldsymbol{e}}_p^T)U \\ &= LTU \qquad \text{where} \quad T = I + (\tilde{\boldsymbol{a}}_q - \boldsymbol{u}_p)\tilde{\boldsymbol{e}}_p^T \end{aligned}$$

- T is formed of known data and readily invertible (like E for PFI)
- **But:** Is this useful?

Novel update techniques for the revised simplex method: 3

Forrest-Tomlin update (1972)

- **Recall:** Column p of B is replaced by $\boldsymbol{a}_q$ to give $\bar{B} = B + (\boldsymbol{a}_q - B\boldsymbol{e}_p)\boldsymbol{e}_p^T$
- **Idea:** Substitute $B = LU$ and take factor L on the left

$$\bar{B} \quad = \quad L[U + (L^{-1}\boldsymbol{a}_q - U\boldsymbol{e}_p)\boldsymbol{e}_p^T] \quad = \quad LU'$$

where $U' = U + (\tilde{\boldsymbol{a}}_q - \boldsymbol{u}_p)\boldsymbol{e}_p^T$ is a "spiked" upper triangular matrix

- Elementary elimination operations given by $\bar{L}$ remove the spike and give $\bar{B} = L\bar{L}\bar{U}$

Multiple Forrest-Tomlin update

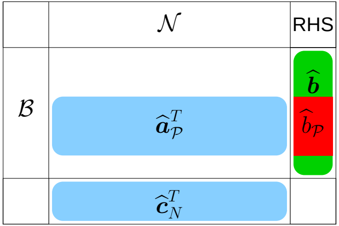
- Update Forrest-Tomlin representation of B after multiple basis changes
- Not just a "broad spike" to be eliminated
- **But:** Is this useful?

COAP best paper prize (2015)

Multiple iteration parallelism (2011-14)

Multiple iteration parallelism with pami

- Perform standard dual simplex minor iterations for rows in set $\mathcal{P}$ ($|\mathcal{P}| \ll m$)
- Suggested by Rosander (1975) but never implemented efficiently *in serial*



- Task-parallel multiple BTRAN to form $\boldsymbol{\pi}_{\mathcal{P}} = B^{-1}\boldsymbol{e}_{\mathcal{P}}$
- Data-parallel PRICE to form $\widehat{\boldsymbol{a}}_p^T$ (as required)
- Task-parallel multiple FTRAN for primal, dual and weight updates

Huangfu and H (2011–2014)

Novel update procedures: are they useful?

Update 1: Alternative product form update (APF)

- **Recall:** $\bar{B} = TB$ with T easily invertible
- Used to get $\boldsymbol{\pi}_{\mathcal{P}} = \bar{B}^{-1}\boldsymbol{e}_{\mathcal{P}}$ from $B^{-1}\boldsymbol{e}_{\mathcal{P}}$ in `pami`
- Used to compute multiple $B^{-1}\boldsymbol{a}_F$ efficiently after multiple BFRT in `pami`

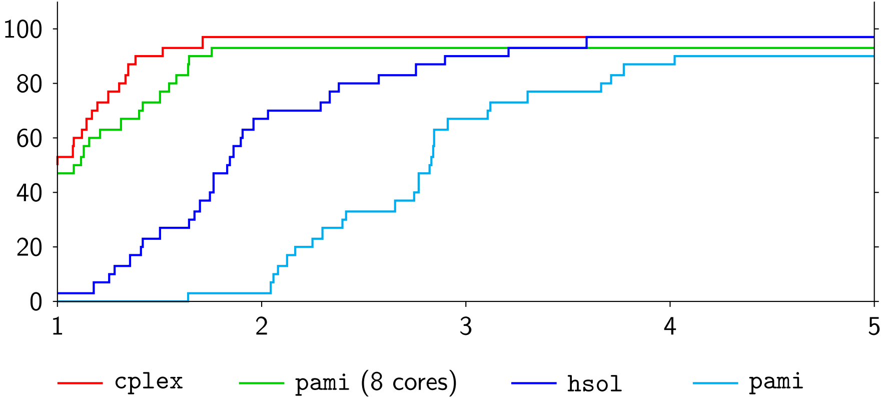
Update 2: Middle product form update (MPF)

- **Recall:** $\bar{B} = LTU$ with T easily invertible
- Not used by `pami`!
- Used by Google in `glop`

Update 3: Multiple Forrest-Tomlin update

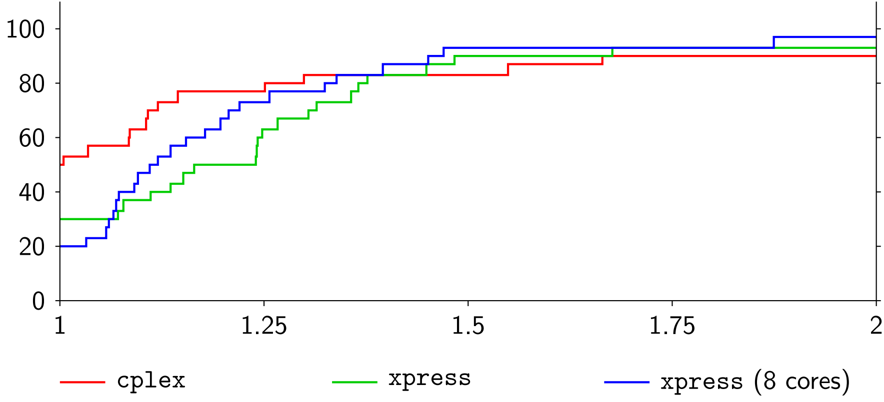
Used to perform multiple Forrest-Tomlin updates after minor iterations in `pami`

Multiple iteration parallelism: cplex vs pami vs hsol



- pami is less efficient than hsol in serial
- pami speedup more than compensates
- pami performance approaching cplex

Multiple iteration parallelism: cplex vs xpress



- pami ideas incorporated in FICO Xpress (Huangfu 2014)
- Xpress simplex solver now fastest commercial simplex solver

The future: `hsol`, an open source high performance simplex solver

hsol: An open source parallel simplex solver

- hsol and its pami variant are high performance codes
- Useful open-source resource?
  - Reliable?
  - Well-written?
  - Better than clp?
- Limitations
  - No presolve
  - No crash
  - No advanced basis start

hsol: Performance and reliability

Extended testing using 159 test problems

- 98 Netlib
- 16 Kennington
- 4 Industrial
- 41 Mittelmann

Exclude 7 which are “hard”

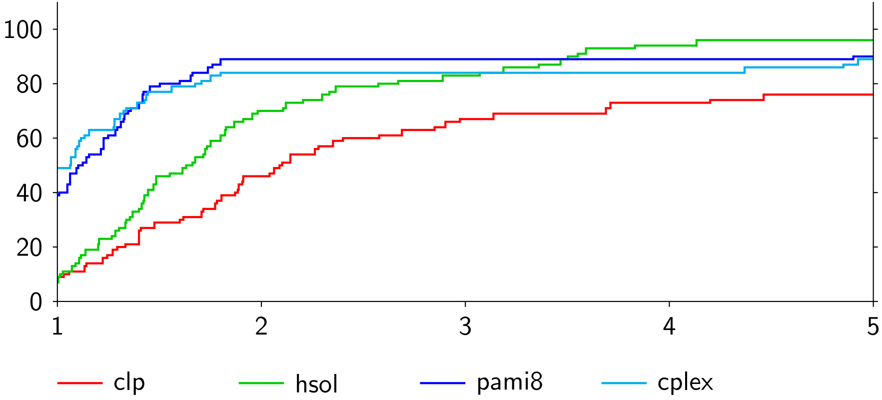
Performance

Benchmark against clp (v1.16) and cplex (v12.5)

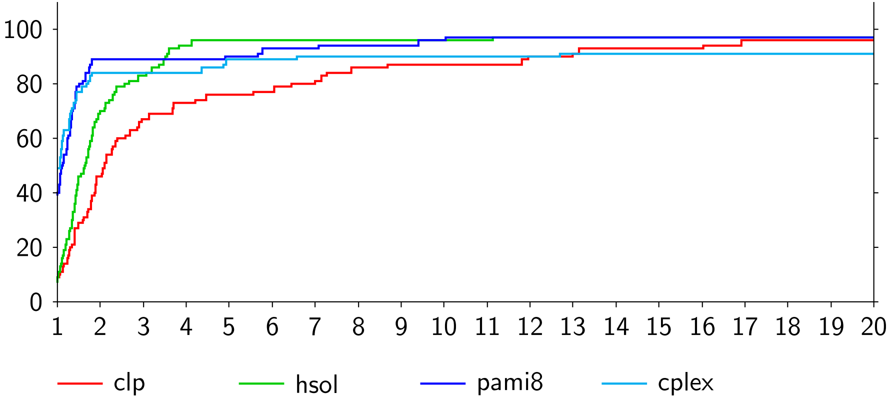
- Dual simplex
- No presolve
- No crash

Ignore results for 82 LPs with minimum solution time below 0.1s

hsol: Performance



hsol: Reliability



hsol: Addressing limitations

Presolve

- Presolve (and corresponding postsolve) has been implemented

  Remove redundancies in the LP to reduce problem dimension — Galabova (2016)

Crash

Value for dual simplex of standard primal crash procedure is being studied

- Dual steepest edge is expensive to initialize when $B \neq I$
- Dual "Devex" costs nothing to initialize but is less effective — H (2016)

Advanced basis start

- Essential for hsol to be used in MIP and SLP solvers
- Only complication is dual steepest edge vs dual "Devex" issue

Conclusions

- Three COAP Prizes
  - Hyper-sparsity: advance in serial simplex
  - Stochastic LP: large scale data-parallelism for special problems
  - Novel updates: (lead to) task and data parallelism in a leading commercial solver
- High performance open-source parallel simplex solver in preparation
- After 70 years, the simplex method still has research potential!

Slides: http://www.maths.ed.ac.uk/hall/ERGO_16.11.30/

References

J. A. J. Hall and K. I. M. McKinnon.
Hyper-sparsity in the revised simplex method and how to exploit it.
Computational Optimization and Applications, 32(3):259–283, December 2005.

Q. Huangfu and J. A. J. Hall.
Parallelizing the dual revised simplex method.
Technical Report ERGO-14-011, School of Mathematics, University of Edinburgh, 2014.
Accepted for publication in Mathematical Programming Computation.

Q. Huangfu and J. A. J. Hall.
Novel update techniques for the revised simplex method.
Computational Optimization and Applications, 60(4):587–608, 2015.

M. Lubin, J. A. J. Hall, C. G. Petra, and M. Anitescu.
Parallel distributed-memory simplex for large-scale stochastic LP problems.
Computational Optimization and Applications, 55(3):571–596, 2013.