High performance numerical linear algebra for the revised simplex method

Julian Hall[1] Qi Huangfu[2] Miles Lubin[3] Ken McKinnon[1] Others

[1]School of Mathematics, University of Edinburgh

[2]FICO

[3]Google

6th IMA Conference on Numerical Linear Algebra and Optimization

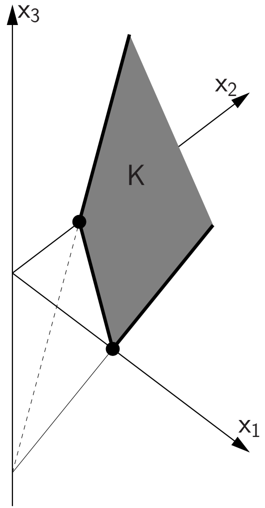
28 June 2018

THE UNIVERSITY of EDINBURGH

Overview

- Background
  - Primal simplex algorithm
  - Dual simplex algorithm
  - NLA challenge
- Hyper-sparsity
- Novel update techniques
- Parallel solution of structured LP problems
- Parallel solution of general LP problems

Solving LP problems: Characterizing a basis

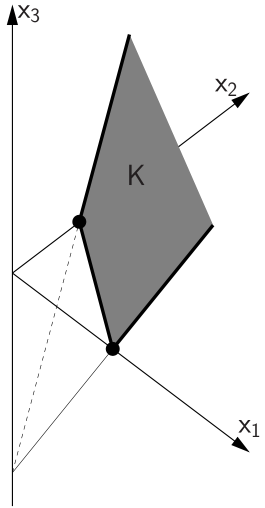


minimize $f = \boldsymbol{c}^T\boldsymbol{x}$ subject to $A\boldsymbol{x} = \boldsymbol{b}$ $\boldsymbol{x} \geq \boldsymbol{0}$

- A **vertex** of the **feasible region** $K \subset \mathbb{R}^n$ 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 $\{1, \ldots, n\}$
- 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
- Points $\boldsymbol{x} \in K$ characterized by
  $$\boldsymbol{x}_B = \widehat{\boldsymbol{b}} - B^{-1}N\boldsymbol{x}_N \quad \text{for some} \quad \boldsymbol{x}_N \geq \boldsymbol{0}$$
  where $\widehat{\boldsymbol{b}} = B^{-1}\boldsymbol{b}$

Solving LP problems: Optimality conditions



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

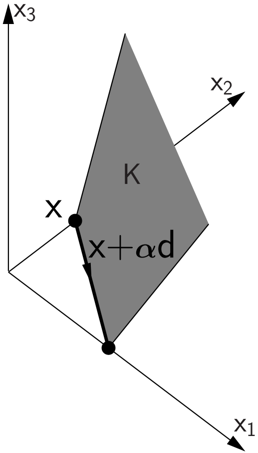
- Objective partitioned according to $\mathcal{B} \cup \mathcal{N}$ as

$$\begin{aligned} f &= \boldsymbol{c}_B^T\boldsymbol{x}_B + \boldsymbol{c}_N^T\boldsymbol{x}_N \\ &= \widehat{f} + \widehat{\boldsymbol{c}}_N^T\boldsymbol{x}_N \end{aligned}$$

  where $\widehat{f} = \boldsymbol{c}_B^T\widehat{\boldsymbol{b}}$ and $\widehat{\boldsymbol{c}}_N^T = \boldsymbol{c}_N^T - \boldsymbol{c}_B^T B^{-1} N$

- 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 $\widehat{b}_i / \widehat{a}_{iq} > 0$ for α and p to leave $\mathcal{B}$
5. Exchange p and q between $\mathcal{B}$ and $\mathcal{N}$
6. Go to 1

Solving dual LP problems: Optimality conditions

- Consider the **dual problem**

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

- For partition $\mathcal{B} \cup \mathcal{N}$ of $\{1, \ldots, n\}$
  - $\boldsymbol{y} = B^{-T}(\boldsymbol{c}_B - \boldsymbol{s}_B)$
  - $\boldsymbol{s} = \begin{bmatrix} \boldsymbol{s}_B \\ \boldsymbol{s}_N \end{bmatrix}$ for $\boldsymbol{s}_N = \widehat{\boldsymbol{c}}_N + N^T B^{-T}\boldsymbol{s}_B$; some $\boldsymbol{s}_B \geq \boldsymbol{0}$
  - Reduced objective is $f_D = \widehat{f} - \widehat{\boldsymbol{b}}^T\boldsymbol{s}_B$
- Solution is optimal 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: Choose a row

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

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

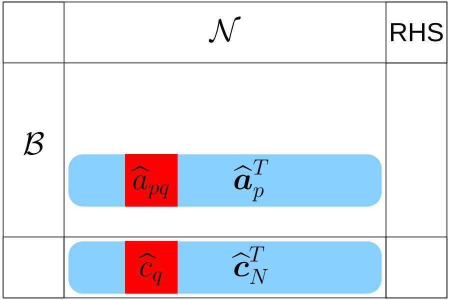
	$\mathcal{N}$	RHS
$\mathcal{B}$		$\widehat{\boldsymbol{b}}$ $\widehat{b}_p$

Dual simplex algorithm: Choose a column

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 $\widehat{c}_j / \widehat{a}_{pj} < 0$ for q to leave $\mathcal{N}$



Dual simplex algorithm: Update cost and RHS

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

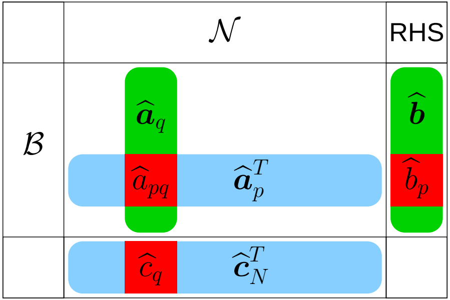
Scan $\widehat{b}_i < 0$ for p to leave $\mathcal{B}$

Scan $\widehat{c}_j / \widehat{a}_{pj} < 0$ for q to leave $\mathcal{N}$

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

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

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



Dual simplex algorithm: Data required

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 $\widehat{c}_j / \widehat{a}_{pj} < 0$ for q to leave $\mathcal{N}$

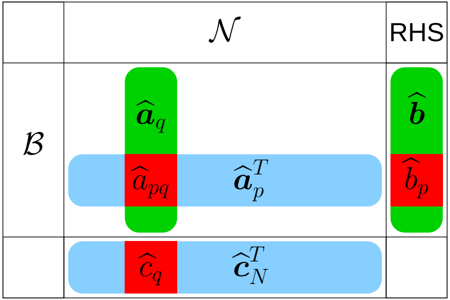
Update: Exchange p and q between $\mathcal{B}$ and $\mathcal{N}$

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 $\widehat{\boldsymbol{a}}_q = B^{-1} \boldsymbol{a}_q$



Solving LP problems: Primal or dual simplex?

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

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		Represent B^{-1}	INVERT

Update B^{-1} exploiting $\bar{B} = B + (\boldsymbol{a}_q - B\boldsymbol{e}_p)\boldsymbol{e}_p^T$ UPDATE

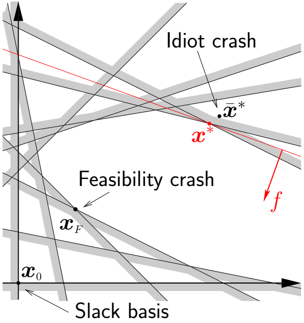
NLA Challenge

- Given initial $\mathcal{B}$ with nonsingular B
- **Each iteration:**
    - Solve $B\,\widehat{\boldsymbol{a}}_q = \boldsymbol{a}_q$
    - Solve $B^T\boldsymbol{\pi}_p = \boldsymbol{e}_p$
    - Column p of B replaced by $\boldsymbol{a}_q$ to give $\bar{B} = B + (\boldsymbol{a}_q - B\boldsymbol{e}_p)\boldsymbol{e}_p^T$
- **Challenge:**
    - Choose initial $\mathcal{B}$
    - Form $PBQ = LU$
        - Solve $B\boldsymbol{x} = \boldsymbol{b}$ for sparse $\boldsymbol{b}$
    - Solve $\bar{B}\boldsymbol{x} = \boldsymbol{b}$

NLA Challenge: Choose initial $\mathcal{B}$

Requirements of initial $\mathcal{B}$

- Must be a useful starting point for the simplex algorithm
- Corresponding matrix B must be
  - Nonsingular
  - Well conditioned
  - Have sparse representation $PBQ = LU$



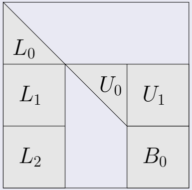
- **"Slack" basis** $(B = I)$ is simple choice $\boldsymbol{x}_0$
- Standard **crash** aims for feasible vertex $\boldsymbol{x}_F$
- **"Idiot" crash** aims for near-optimal point $\bar{\boldsymbol{x}}^*$

Galabova and H (2018)
Ivet's talk - 15:40 today!

NLA Challenge: Form $PBQ = LU$

Triangularisation

Identify row and column singletons until every active nonzero has positive Markowitz merit



Solve $B\boldsymbol{x} = \boldsymbol{r}$ as

$$\begin{aligned} L_0\boldsymbol{x}_L &= \boldsymbol{r}_L \\ B_0\boldsymbol{x}_0 &= \boldsymbol{r}_0 - L_2\boldsymbol{x}_L \\ U_0\boldsymbol{x}_U &= \boldsymbol{r}_U - L_1\boldsymbol{x}_L - U_1\boldsymbol{x}_0 \end{aligned}$$

- LP basis matrices are typically **highly reducible**: $\dim(B_0) \ll m$
- For network flow problems B is provably triangularisable

NLA Challenge: Solve $\bar{B}\boldsymbol{x} = \boldsymbol{r}$ using the product form update (PFI)

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

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

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

$\mu = \widehat{a}_{pq}$ 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)

NLA Challenge: Solve $\bar{B}\boldsymbol{x} = \boldsymbol{r}$ using the Forrest-Tomlin update (FT)

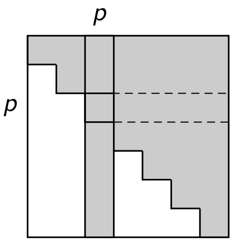
- Given

$$\bar{B} = B + (\boldsymbol{a}_q - B\boldsymbol{e}_p)\boldsymbol{e}_p^T \quad \text{where (wlog)} \quad B = LU$$

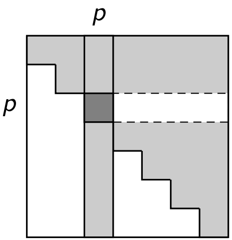
- Multiply $\bar{B}$ by L^{-1} to give

$$L^{-1}\bar{B} \;=\; U + (L^{-1}\boldsymbol{a}_q - U\boldsymbol{e}_p)\boldsymbol{e}_p^T \;=\; U + (\widetilde{\boldsymbol{a}}_q - \boldsymbol{u}_p)\boldsymbol{e}_p^T \;=\; U' \qquad \text{(a)}$$

- Eliminate entries in row p to give $R^{-1}U' = \bar{U}$ (b)



(a) Spiked upper: U'



(b) After elimination: $\bar{U}$

- Yields $\bar{B} = LR\bar{U}$
- Compute $\widetilde{\boldsymbol{a}}_q$ when forming $\widehat{\boldsymbol{a}}_q$
- Represent R like E
- FT more efficient than PFI with respect to sparsity

Forrest and Tomlin (1972)

NLA Challenge: Hyper-sparsity

NLA Challenge: Solve $B\mathbf{x} = \mathbf{r}$ for sparse $\mathbf{r}$

- Given $B = LU$, solve

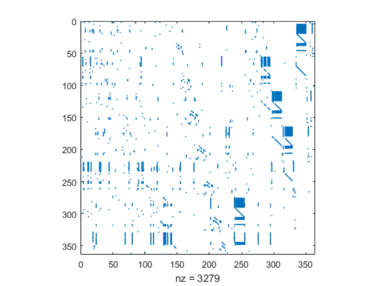
$$L\mathbf{y} = \mathbf{r}; \quad U\mathbf{x} = \mathbf{y}$$

- In revised simplex method, $\mathbf{r}$ is sparse: consequences?
  - If B is irreducible then $\mathbf{x}$ is full
  - If B is highly reducible then $\mathbf{x}$ can be **sparse**
- Phenomenon of **hyper-sparsity**
  - Exploit it when *forming* $\mathbf{x}$
  - Exploit it when *using* $\mathbf{x}$

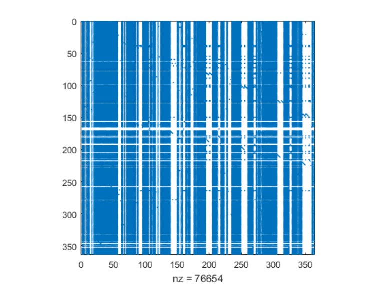
NLA Challenge: Hyper-sparsity

Inverse of a sparse matrix and solution of $B\boldsymbol{x} = \boldsymbol{r}$

Optimal B for LP problem `stair`

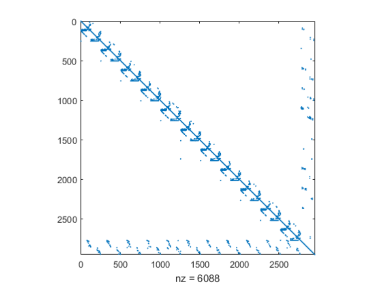


B^{-1} has density of 58%, so $B^{-1}\boldsymbol{r}$ is typically dense

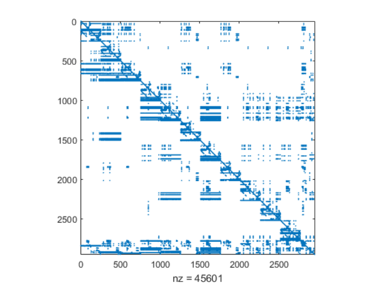


Inverse of a sparse matrix and solution of $B\boldsymbol{x} = \boldsymbol{r}$

Optimal B for LP problem pds-02



B^{-1} has density of 0.52%, so $B^{-1}\boldsymbol{r}$ is typically sparse—when $\boldsymbol{r}$ is sparse



NLA Challenge: Hyper-sparsity

- Use solution of $L\boldsymbol{x} = \boldsymbol{b}$
  - To illustrate the phenomenon of hyper-sparsity
  - To demonstrate how to exploit hyper-sparsity
- Apply principles to other triangular solves in the simplex method

Recall: Solve $L\boldsymbol{x} = \boldsymbol{b}$ using

function `ftranL`$(L, \boldsymbol{b}, \boldsymbol{x})$

$\quad \boldsymbol{r} = \boldsymbol{b}$

$\quad$ **for all** $j \in \{1, \ldots, m\}$ **do**

$\qquad$ **for all** $i : L_{ij} \neq 0$ **do**

$$r_i = r_i - L_{ij} r_j$$

$\quad \boldsymbol{x} = \boldsymbol{r}$

When $\boldsymbol{b}$ is **sparse**

- Inefficient until $\boldsymbol{r}$ fills in

NLA Challenge: Hyper-sparsity

Better: Check r_j for zero

function $\texttt{ftranL}(L, \boldsymbol{b}, \boldsymbol{x})$

$\quad \boldsymbol{r} = \boldsymbol{b}$

$\quad$ **for all** $j \in \{1, \dots, m\}$ **do**

$\quad\quad$ **if** $r_j \neq 0$ **then**

$\quad\quad\quad$ **for all** $i : L_{ij} \neq 0$ **do**

$\quad\quad\quad\quad r_i = r_i - L_{ij} r_j$

$\quad \boldsymbol{x} = \boldsymbol{r}$

When $\boldsymbol{x}$ is **sparse**

- Few values of r_j are nonzero
- Check for zero dominates
- Requires more efficient identification of set $\mathcal{X}$ of indices j such that $r_j \neq 0$

Gilbert and Peierls (1988)
H and McKinnon (1998–2005)

NLA Challenge: Hyper-sparsity

Recall: major computational components

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

BTRAN: Form $\boldsymbol{\pi}_p = B^{-T}\boldsymbol{e}_p$

- Transposed triangular solves
- $L^T\boldsymbol{x} = \boldsymbol{b}$ has $x_i = b_i - \boldsymbol{l}_i^T\boldsymbol{x}$
  - Hyper-sparsity: $\boldsymbol{l}_i^T\boldsymbol{x}$ typically zero
  - Also store L (and U) row-wise and use FTRAN code

PRICE: Form $\widehat{\boldsymbol{a}}_p^T = \boldsymbol{\pi}_p^T N$

- Hyper-sparsity: $\boldsymbol{\pi}_p^T$ is sparse
- Store N row-wise
- Form $\widehat{\boldsymbol{a}}_p^T$ as a combination of rows of N for nonzeros in $\boldsymbol{\pi}_p^T$

H and McKinnon (1998–2005)

NLA Challenge: Novel update techniques

NLA Challenge: Alternative product form update

- **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 takes B out as a factor on the left so $\bar{B} = BE$
- **Idea:** Why not take it out on the right!

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

- T is formed of known data and readily invertible (like E for PFI)
  Naturally compute $\widehat{\boldsymbol{e}}_p$ when solving $B^T\boldsymbol{\pi}_p = \boldsymbol{e}_p$

NLA Challenge: Middle product form update

- **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!

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

- M is formed of known data and readily invertible (like E for PFI)
  Naturally compute $\widetilde{\boldsymbol{a}}_q$ when solving $B\,\widehat{\boldsymbol{a}}_q = \boldsymbol{a}_q$ and $\widetilde{\boldsymbol{e}}_p$ when solving $B^T\boldsymbol{\pi}_p = \boldsymbol{e}_p$

NLA Challenge: Collective Forrest-Tomlin update

- Update Forrest-Tomlin representation of B after multiple basis changes
- Don't have data to perform a sequence of standard FT updates
- Have to perform elimination corresponding to multiple spikes

Huangfu and H (2013)

NLA Challenge: Parallel solution of structured LP problems

NLA Challenge: Parallel solution of stochastic MIP problems

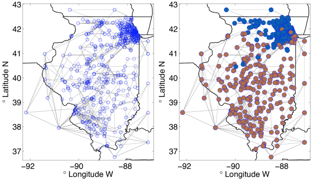
Two-stage stochastic LPs have column-linked block angular (BALP) structure

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

- Variables $\boldsymbol{x}_0 \in \mathbb{R}^{n_0}$ are **first stage** decisions
- Variables $\boldsymbol{x}_i \in \mathbb{R}^{n_i}$ 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

NLA Challenge: Parallel solution of stochastic MIP problems

- 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`



PIPS-S: Exploiting problem structure

Convenient to permute the LP thus:

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

PIPS-S: Exploiting problem structure

- Inversion of the basis matrix B is key to revised simplex efficiency

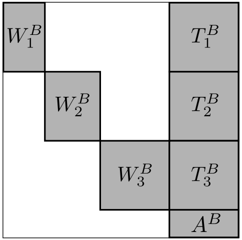
$$B = \begin{bmatrix} W_1^B & & & T_1^B \\ & \ddots & & \vdots \\ & & W_N^B & T_N^B \\ & & & A^B \end{bmatrix}$$

- W_i^B are columns corresponding to n_i^B basic variables in scenario i
- $\begin{bmatrix} T_1^B \\ \vdots \\ T_N^B \\ A^B \end{bmatrix}$ are columns corresponding to n_0^B basic first stage decisions

PIPS-S: Exploiting problem structure

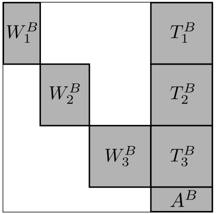
- Inversion of the basis matrix B is key to revised simplex efficiency

$$B = \begin{bmatrix} W_1^B & & & T_1^B \\ & \ddots & & \vdots \\ & & W_N^B & T_N^B \\ & & & A^B \end{bmatrix}$$

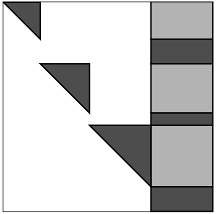


- B is nonsingular so
  - W_i^B are "tall": full column rank
  - $\begin{bmatrix} W_i^B & T_i^B \end{bmatrix}$ are "wide": full row rank
  - A^B is "wide": full row rank
- Scope for parallel inversion is immediate and well known

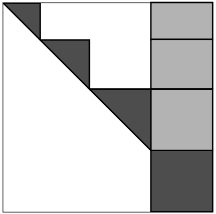
PIPS-S: Exploiting problem structure



- Eliminate sub-diagonal entries in each W_i^B (independently)



- Apply elimination operations to each T_i^B (independently)



- Accumulate non-pivoted rows from the W_i^B with A^B and complete elimination

PIPS-S: Overview

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

Lubin, H, Petra and Anitescu (2013)

PIPS-S: Results

On Fusion cluster: Performance relative to `clp`

Dimension	Cores	`Storm`	`SSN`	`UC12`	`UC24`
$m + n = O(10^6)$	1	0.34	0.22	0.17	0.08
	32	8.5	6.5	2.4	0.7
$m + n = O(10^7)$	256	299	45	67	68

On Blue Gene

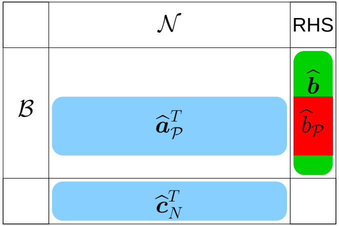
- Instance of `UC12`
- $m + n = O(10^8)$
- Requires 1 TB of RAM
- Runs from an advanced basis

Cores	Iterations	Time (h)	Iter/sec
1024	Exceeded execution time limit		
2048	82,638	6.14	3.74
4096	75,732	5.03	4.18
8192	86,439	4.67	5.14

NLA Challenge: Parallel solution of general LP problems

NLA Challenge: Parallel solution of general LP problems

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

Huangfu and H (2011–2014)

NLA Challenge: `HiGHS` (2011–date)

Overview

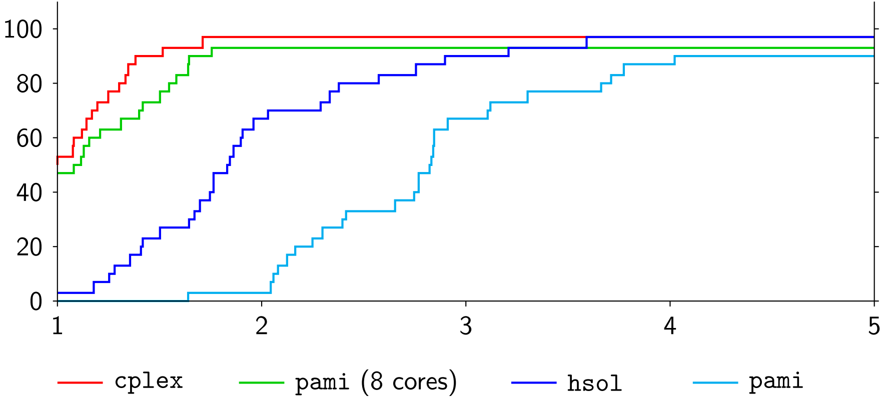
- Written in C++ to study parallel simplex
- Dual simplex with steepest edge and BFRT
- Forrest-Tomlin update
  - complex and inherently serial
  - efficient and numerically stable

Concept

- High performance serial solver (`hsol`)
- Exploit limited task and data parallelism in standard dual RSM iterations (`sip`)
- Exploit greater task and data parallelism via minor iterations of dual SSM (`pami`)
- Test-bed for research
- Work-horse for consultancy

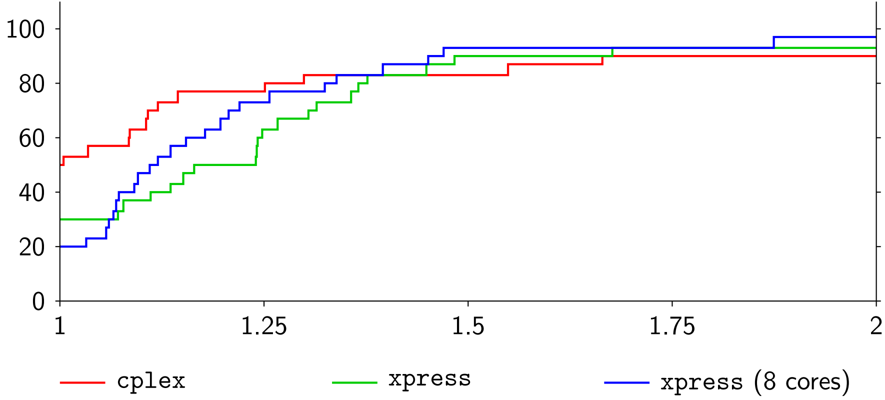
Huangfu, H and Galabova (2011–date)

HiGHS: cplex vs pami vs hsol



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

HiGHS: Impact



- pami ideas incorporated in FICO Xpress (Huangfu 2014)
- Xpress has been the fastest simplex solver for most of the past five years

To close

Conclusions

- Revised simplex method offers NLA challenges
- Novel techniques of practical value are hard to find but get noticed
- Look for alternative algorithms for fast (approximate) solution of LPs

Slides:
http://www.maths.ed.ac.uk/hall/NLAO18

Code:
https://github.com/ERGO-Code/HiGHS

I. L. Galabova and J. A. J. Hall.
A quadratic penalty algorithm for linear programming and its application to linearizations of quadratic assignment problems.
Technical Report ERGO-18-009, School of Mathematics, University of Edinburgh, 2018.

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

Q. Huangfu and J. A. J. Hall.
Parallelizing the dual revised simplex method.
Mathematical Programming Computation, 10(1):119–142, 2018.

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.