Hyper-sparsity in the revised simplex method and how to exploit it

Julian Hall

Ken McKinnon

Department of Mathematics and Statistics

University of Edinburgh

25th June 2003

Overview

- The revised simplex method
- What is hyper-sparsity and how can it be exploited?
- Results, conclusions and extensions

The revised simplex method for LP

$$\begin{array}{rl} \text{minimize} & f = \boldsymbol{c}^T\boldsymbol{x} \\ \text{subject to} & A\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0} \\ \text{where} & \boldsymbol{x} \in I\!R^n \quad \text{and} \quad \boldsymbol{b} \in I\!R^m. \end{array}$$

At any stage in the revised simplex method

- the variables are partitioned into index sets

  $\mathcal{B}$ of m basic variables $\boldsymbol{x}_B \geq \boldsymbol{0}$

  $\mathcal{N}$ of $n - m$ nonbasic variables $\boldsymbol{x}_N = \boldsymbol{0}$

- the corresponding components of $\boldsymbol{c}$ and columns of A are

  the basic costs $\boldsymbol{c}_B$ and basis matrix B

  the non-basic costs $\boldsymbol{c}_N$ and matrix N

- there is a factored inverse B^{-1} of the basis matrix.

Revised simplex method

`CHUZC:` Scan the reduced costs $\hat{\boldsymbol{c}}_N$ for a good candidate q to enter the basis.

`FTRAN:` Form the pivotal column $\hat{\boldsymbol{a}}_q = B^{-1}\boldsymbol{a}_q$, where $\boldsymbol{a}_q$ is column q of A.

`CHUZR:` Scan the ratios $\hat{b}_i/\hat{a}_{iq}$ for the row p of a good candidate to leave the basis. Let $\alpha = \hat{b}_p/\hat{a}_{pq}$.
Update $\hat{\boldsymbol{b}} := \hat{\boldsymbol{b}} - \alpha\hat{\boldsymbol{a}}_q$.

`BTRAN:` Form $\boldsymbol{\pi}_p^T = \boldsymbol{e}_p^T B^{-1}$.

`PRICE:` Form the pivotal row $\hat{\boldsymbol{a}}_p^T = \boldsymbol{\pi}_p^T N$.
Update reduced costs $\hat{\boldsymbol{c}}_N^T := \hat{\boldsymbol{c}}_N^T - \hat{c}_q\hat{\boldsymbol{a}}_p^T$.

`If` (growth in factors) `then`

  `INVERT:` Form a factored representation of B^{-1}.

`else`

  `UPDATE:` Update the factored representation of B^{-1} corresponding to the basis change.

`end if`

Exploiting sparsity in solving $B\hat{\boldsymbol{a}}_q = \boldsymbol{a}_q$ and $B^T\boldsymbol{\pi} = \boldsymbol{e}_p$

- Maintain an eta file $\{p_k, \eta_k, \boldsymbol{\eta}_k\}_{k=1}^r$ so that

$$B^{-1} = \prod_{k=1}^{r} E_k \quad \text{where} \quad E_k = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & \boldsymbol{\eta}_k & & & \\ & & 1 & & & & \\ & & & \eta_k & & & \\ & & & & 1 & & \\ & & & \boldsymbol{\eta}_k & & \ddots & \\ & & & & & & 1 \end{bmatrix} \begin{matrix} \\ \\ \\ \leftarrow \text{row } p_k \\ \\ \\ \\ \end{matrix}$$

Entries in eta file correspond to columns in LU decomposition of B_0 and subsequent basis changes.

Sparse FTRAN

- $B\boldsymbol{x} = \boldsymbol{b}$ may be solved by transforming $\boldsymbol{b}$ into $\boldsymbol{x}$ as follows.

  `do` $k = 1, r$

  $\quad b_{p_k} := b_{p_k}/\eta_k$

  $\quad \boldsymbol{b} := \boldsymbol{b} - b_{p_k}\boldsymbol{\eta}_k$

  `end do`

- As etas are applied there is **fill-in** in the RHS.
- According to the problem
  - fill-in can accumulate rapidly so that $\boldsymbol{x}$ is dense
  - little fill-in occurs so that $\boldsymbol{x}$ is sparse

What is hyper-sparsity?

- Hyper-sparsity exists when the results of matrix-vector operations in the revised simplex method are sparse.
  - The pivotal column $\hat{\boldsymbol{a}}_q = B^{-1}\boldsymbol{a}_q$ is sparse.
  - The 'update $\boldsymbol{\pi}$' $\boldsymbol{\pi}^T = \boldsymbol{e}_p^T B^{-1}$ is sparse.
  - The pivotal row $\hat{\boldsymbol{a}}_p^T = \boldsymbol{\pi}^T N$ is sparse.
- Exploit hyper-sparsity both when forming and using these vectors.

What problems (may) exhibit hyper-sparsity?

- Problems with a significant network structure
- Problems with Dantzig-Wolfe (row linked block diagonal) structure
  - Multicommodity flow problems
- Problems with Benders (column linked block diagonal) structure
- Other general LP problems exhibit hyper-sparsity for no obvious reason

FTRAN with sparse RHS

- In general, when solving LP problems, $B\boldsymbol{x} = \boldsymbol{b}$ has a **sparse RHS** $\boldsymbol{a}_q$.
- $B\boldsymbol{x} = \boldsymbol{b}$ may be solved by transforming $\boldsymbol{b}$ into $\boldsymbol{x}$ as follows.

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

- In presence of hyper-sparsity, the dominant cost is the test for zero.

Hyper-sparse FTRAN

During each INVERT:

- For each row $i = 1, \ldots, m$,
  - record the index k of the *first* eta for which $p_k = i$
  - record the index k of the *second* eta for which $p_k = i$

During each FTRAN:

Let $\mathcal{R}$ be the set of indices of nonzeros in $\boldsymbol{b}$.

Let $\mathcal{E}$ be the set of etas with pivots corresponding to nonzeros in $\boldsymbol{b}$.

1 Scan $\mathcal{E}$ to determine the index k of the next eta to be applied.

If k is undefined then FTRAN is complete.

Apply eta k:

add to $\mathcal{R}$ the indices of any RHS entries where nonzeros are created.

add to $\mathcal{E}$ etas with pivots where nonzeros are created.

Go to 1.

Hyper-sparse CHUZR

Recall: Scan the ratios $\hat{b}_i/\hat{a}_{iq}$

- Following hyper-sparse FTRAN, the indices of the nonzeros in $\hat{a}_{iq}$ are known.

Hyper-sparse BTRAN with unit RHS

- $B^T\boldsymbol{x} = \boldsymbol{b}$ may be solved by transforming $\boldsymbol{b}$ into $\boldsymbol{x}$ as follows.

  `do` $k = r,\ 1,\ -1$

  $\qquad b_{p_k} := (b_{p_k} + \boldsymbol{b}^T\boldsymbol{\eta}_k)/\eta_k$

  `end do`

- No way to exploit hypersparsity properly with 'column-wise' eta file.
- Form additional 'row-wise' eta file
  - After INVERT: Form a 'row-wise' copy of the eta file.
  - Pass row-wise eta file to hyper-sparse forward solution code.

Hyper-sparse PRICE

Recall: $\hat{\boldsymbol{a}}_p^T = \boldsymbol{\pi}^T N$.

- Form $\hat{\boldsymbol{a}}_p^T$ as a linear combination of rows of N corresponding to nonzeros in $\boldsymbol{\pi}^T$.
- Maintain list of indices of nonzeros in $\hat{\boldsymbol{a}}_p^T$.

Requires row-wise representation of N to be maintained.

Hyper-sparse CHUZC

Recall: Reduced costs updated according to

$$\hat{\boldsymbol{c}}_N^T := \hat{\boldsymbol{c}}_N^T - \hat{c}_q \hat{\boldsymbol{a}}_p^T$$

- $\hat{\boldsymbol{c}}_N^T = \boldsymbol{c}_N^T - \boldsymbol{c}_B^T B^{-1} N$ is typically dense
- $\hat{\boldsymbol{a}}_p^T$ is sparse—few reduced costs change
- Initially: full CHUZC to form list of $(\leq s)$ good candidates
- Each iteration:
  - Form list of $(\leq s)$ good candidates from those with changed reduced cost
  - Merge lists to form new list of $(\leq s)$ good candidates
- Periodically: full CHUZC to pick up good candidates which have never changed

Hyper-sparse (Tomlin) INVERT

- The Tomlin INVERT calculates columns of the active submatrix as required
- Column k requires the solution of $(\prod_{i=1}^{k-1} L_i)\hat{\boldsymbol{a}}_k = \boldsymbol{a}_k$
- Apply techniques for hyper-sparse FTRAN

Hyper-sparse (product-form) UPDATE

Recall: B^{-1} is represented as $B^{-1} = E_U^{-1} B_0^{-1}$, where B_0^{-1} is obtained by INVERT.

Eta file for E_U^{-1} requires packed form of $\boldsymbol{\eta} = \begin{bmatrix} \hat{a}_{1q} \\ \vdots \\ \hat{a}_{p-1\,q} \\ -1/\hat{a}_{pq} \\ \hat{a}_{p+1\,q} \\ \vdots \\ \hat{a}_{nq} \end{bmatrix}$

This is formed at almost no cost during hyper-sparse CHUZR.

Speedup of EMSOL for problems exhibiting hyper-sparsity

Problem	speedup in total solution time and computational components							
	Solution	CHUZC	I-FTRAN	CHUZR	I-BTRAN	U-BTRAN	PRICE	INVERT
80BAU3B	3.34	3.05	5.13	1.93	3.51	6.72	6.06	1.34
FIT2P	1.75	18.91	1.30	0.93	12.22	3.59	13.47	0.87
GREENBEA	2.71	1.33	1.30	1.13	3.60	19.87	3.45	2.83
GREENBEB	2.44	1.39	1.35	1.21	3.69	21.88	3.44	2.78
STOCFOR3	1.85	4.47	1.14	0.96	7.26	56.99	7.61	3.16
WOODW	3.40	1.82	1.70	1.30	4.17	11.72	5.14	1.53
DCP1	3.25	1.58	2.36	1.19	3.87	6.25	6.71	1.70
DCP2	5.32	1.60	8.24	2.51	6.21	13.99	6.20	8.63
CRE-A	3.05	2.54	4.00	1.72	3.64	6.50	4.48	1.14
CRE-C	2.89	2.88	4.67	1.97	3.58	6.53	4.97	1.08
KEN-11	22.84	19.36	98.04	13.93	27.22	9.90	66.36	1.02
KEN-13	12.12	6.31	104.09	7.31	12.87	9.19	17.60	0.94
KEN-18	15.27	6.63	263.94	15.27	13.91	13.07	19.92	1.01
PDS-06	17.48	15.25	24.07	3.57	21.58	35.76	28.18	1.02
PDS-10	10.36	10.67	11.24	1.85	16.60	49.99	17.55	0.96
PDS-20	10.35	8.58	5.96	1.68	14.33	189.19	15.40	1.44
Mean	5.21	4.38	7.03	2.28	7.64	15.44	9.71	1.55

Comparison of EMSOL with CPLEX and SOPLEX

	EMSOL	CPLEX 6.5 (Devex)			SOPLEX 1.2 (Devex)		
Problem	Solution time CPU (s)	Solution time CPU (s)	EMSOL solution speedup	EMSOL iteration speedup	Solution time CPU (s)	EMSOL solution speedup	EMSOL iteration speedup
80BAU3B	9.43	14.39	1.53	1.37	20.50	2.17	1.78
FIT2P	67.34	42.08	0.62	0.47	1058.79	15.72	1.30
GREENBEA	24.63	21.18	0.86	1.00	150.53	6.11	1.38
GREENBEB	19.55	18.31	0.94	1.20	76.36	3.91	1.56
STOCFOR3	201.21	41.15	0.20	0.25	154.11	0.77	0.64
WOODW	3.09	3.99	1.29	1.50	21.41	6.93	2.58
DCP1	14.64	16.30	1.11	1.22	330.35	22.56	1.69
DCP2	483.64	1412.89	2.92	1.40	1507.40	3.12	1.24
CRE-A	4.49	6.29	1.40	1.45	5.07	1.13	1.09
CRE-C	3.50	5.91	1.69	1.66	3.39	0.97	1.06
KEN-11	9.19	13.57	1.48	1.24	46.99	5.11	2.20
KEN-13	100.77	91.41	0.91	0.79	417.26	4.14	2.05
KEN-18	1421.50	786.08	0.55	0.48	5167.60	3.64	1.91
PDS-06	8.20	5.49	0.67	0.90	44.22	5.39	3.37
PDS-10	83.83	26.92	0.32	0.65	158.79	1.89	2.37
PDS-20	1059.19	238.48	0.23	0.53	1754.17	1.66	2.73
Mean			0.84	0.90		3.47	1.67

How much of all this is new?

Very little is published on the (detailed) computational techniques which make commercial codes run fast.

- Storing row-wise copy of L to speed up BTRAN suggested by Reid.
- Forrest-Tomlin INVERT precludes one-off processing of B_0^{-1}.
- CPLEX (≥ 6.5) exploits hyper-sparsity using Gilbert-Peierls algorithm.

Conclusions and extensions

- Considerable improvement in efficiency for a significant class of problems.
- Speedup is limited by computational cost of the relatively small number iterations where hypersparsity is not exhibited.
  - Can this number of iterations be reduced by modified column and row selection strategies?
- Heuristic approach to hyper-sparse FTRAN and BTRAN may be inefficient for very large problems.
  - May need Gilbert-Peierls.
- 'Partial' PRICE and other algorithmic refinements used by CPLEX and SO-PLEX but not EMSOL.