LUSOL: A basis package for constrained optimization

Linear Algebra and Optimization Seminar
SCCM, Stanford University
Feb 8, 2006

Michael Saunders

Systems Optimization Laboratory (SOL)
Dept of Management Science & Engineering
Stanford University
saunders@stanford.edu

Abstract

A basis package allows linear systems $Ax = b$ to be solved when columns of the matrix A are replaced one by one. Stability and efficiency must be balanced when the matrices are large.

LUSOL maintains LU factors of sparse matrices of any shape (square or rectangular). Threshold Rook Pivoting is an important feature for revealing rank (and recovering from singularity). Updates are stabilized by the Bartels-Golub approach.

We review the open-source Fortran and C implementations and their use within the optimization packages MINOS, SNOPT, PATH, ZIP, and lp_solve.

LUSOL

Maintaining LU factors of a general sparse matrix A
Gill, Murray, Saunders, and Wright (1987)

Code contributors

F77	Saunders (1986–present)
	following Duff, Reid, Zlatev, Suhl and Suhl
MATLAB Fmex	Michael O'Sullivan (1999–present)
C (for lp_solve)	Kjell Eikland (2004–present)
MATLAB Cmex	Yin Zhang (2005–present)

Features

Square or rectangular A	
Rank-revealing LU	for "basis repair"
Stable updates	Bartels-Golub style

LUSOL

$$A = \boxed{\rule{0pt}{4em}\hspace{2.5em}} \quad \text{or} \quad \boxed{\rule{0pt}{2.5em}\hspace{2.5em}} \quad \text{or} \quad \boxed{\rule{0pt}{2.5em}\hspace{4.5em}} = LU$$

FACTOR [L,U,p,q] = luSOL(A)

SOLVE $Lx = y, \; L^Tx = y, \; Ux = y, \; U^Tx = y, \; Ax = y, \; A^Tx = y$

UPDATE Add, replace, delete a column
Add, replace, delete a row
Add a rank-one matrix

MULTIPLY $x = Ly, \; x = L^Ty, \; x = Uy, \; x = U^Ty, \; x = Ay, \; x = A^Ty$

LU Factorization

LU factors of a vector

$$\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} a & \\ b & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} a & & & \\ b & 1 & & \\ c & & 1 & \\ d & & & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & & & \\ b/a & 1 & & \\ c/a & & 1 & \\ d/a & & & 1 \end{pmatrix} \begin{pmatrix} a \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

We choose to **keep L well-conditioned**

$\Rightarrow |a|$ not too small

LU factors of two vectors

$$\begin{pmatrix} a & e \\ b & f \\ c & g \\ d & h \end{pmatrix} = \begin{pmatrix} 1 & & & \\ b/a & 1 & & \\ c/a & & 1 & \\ d/a & & & 1 \end{pmatrix} \begin{pmatrix} a & e \\ 0 & f-(b/a)e \\ 0 & g-(c/a)e \\ 0 & h-(d/a)e \end{pmatrix}$$

$$A = L \quad U$$

Forward substitution: "$L\,x = b$"

Forward sub gives 2nd col of U: $L\,U_2 = A_2$

Permute rows **and/or columns** to preserve stability

$$a \quad \text{not too small,} \qquad \mu = b/a$$

Gaussian elimination

$$\begin{pmatrix} 1 & \\ -\mu & 1 \end{pmatrix} \begin{pmatrix} a & c & e \\ b & d & f \end{pmatrix} = \begin{pmatrix} a & c & e \\ & d-\mu c & f-\mu e \end{pmatrix}$$

LU factorization (forward substitution)

$$\begin{pmatrix} a & c & e \\ b & d & f \end{pmatrix} = \begin{pmatrix} 1 & \\ \mu & 1 \end{pmatrix} \begin{pmatrix} a & c & e \\ & d-\mu c & f-\mu e \end{pmatrix}$$

$$L = \begin{pmatrix} 1 & \\ 100 & 1 \end{pmatrix}$$

$$\mathrm{cond}(L) \approx 100\,??$$

$$L = \begin{pmatrix} 1 & \\ 100 & 1 \end{pmatrix}$$

$$\mathrm{cond}(L) \approx 10000$$

$$L \;=\; \begin{pmatrix} 1 & \\ 100 & 1 \end{pmatrix}$$

$$\mathrm{cond}(L) \;\approx\; 10000$$

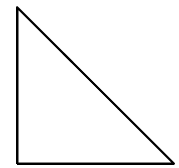
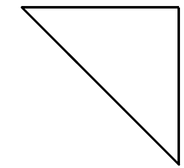
Fortunately, triangular solves

$$\begin{pmatrix} 1 & \\ \mu & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} b \\ e \end{pmatrix}$$

behave as if $\mathrm{cond}(L) \le 100$ (Wilkinson, 1963)

RANK-REVEALING LU

FACTOR

$[\mathsf{L},\mathsf{U},\mathsf{p},\mathsf{q}] = \mathsf{luSOL}(\mathsf{A})$ $\quad$ $\mathsf{L}(\mathsf{p},\mathsf{p}) =$  $\quad$ $\mathsf{U}(\mathsf{p},\mathsf{q}) =$ 

- Well defined for **any square or rectangular** A
- Permutations p, q balance **sparsity** and **stability**
- Markowitz strategy for suggesting **sparse** pivots
- **Stability** options:
  TPP Threshold Partial Pivoting
  TRP Threshold Rook Pivoting
  TCP Threshold Complete Pivoting

Partial Pivoting

$$
\begin{array}{ccccccc}
\cdot & \cdots & \cdots & \cdots & \cdots & \cdots & \cdot \\
 & \ddots & & & & & \vdots \\
 & & \ddots & & & & \vdots \\
 & & & 4.0 & \times & \times & \times \\
 & & & 2.0 & \times & \times & \times \\
 & & & 1.0 & \times & \times & \times \\
 & & & 4.0 & \times & \times & \times \\
 & & & 0.1 & \times & \times & \times
\end{array}
$$

Rook Pivoting

$$
\begin{array}{ccccccc}
\cdot & \cdots & \cdots & \cdots & \cdots & \cdots & \cdot \\
 & \ddots & & & & & \vdots \\
 & & \ddots & & & & \vdots \\
 & & & {\color{red}6.0} & {\color{blue}1.0} & {\color{blue}0.1} & {\color{blue}6.0} \\
 & & & {\color{blue}2.0} & \times & \times & \times \\
 & & & {\color{blue}1.0} & \times & \times & \times \\
 & & & {\color{blue}4.0} & \times & \times & \times \\
 & & & {\color{blue}0.1} & \times & \times & \times
\end{array}
$$

Complete Pivoting

$$
\begin{array}{ccccccc}
\cdot & \cdots & \cdots & \cdots & \cdots & \cdots & \cdot \\
 & \ddots & & & & & \vdots \\
 & & \ddots & & & & \vdots \\
 & & & \color{red}{9.0} & \color{blue}{1.0} & \color{blue}{0.1} & \color{blue}{6.0} \\
 & & & \color{blue}{2.0} & \color{blue}{\times} & \color{blue}{\times} & \color{blue}{\times} \\
 & & & \color{blue}{1.0} & \color{blue}{\times} & \color{blue}{\times} & \color{blue}{\times} \\
 & & & \color{blue}{4.0} & \color{blue}{\times} & \color{blue}{\times} & \color{blue}{9.0} \\
 & & & \color{blue}{0.1} & \color{blue}{\times} & \color{blue}{0.1} & \color{blue}{\times}
\end{array}
$$

TPP: Threshold Partial Pivoting

$$
\begin{array}{ccccccc}
\cdot & \cdots & \cdots & \cdots & \cdots & \cdots & \cdot \\
 & \ddots & & & & & \vdots \\
 & & \ddots & & & & \vdots \\
 & & & 2.0 & \times & & \times \\
 & & & 2.0 & & & \times \\
 & & & & \times & \times & \\
 & & & 4.0 & \times & \times & \times \\
 & & & & \times & & \times
\end{array}
$$

Require $|L_{ij}| \le 2.0$ (not 1.0)

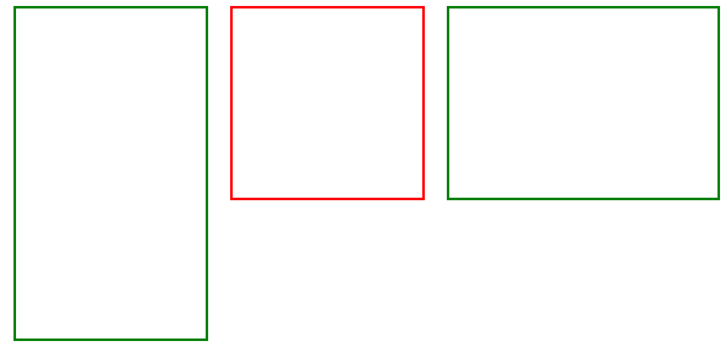
TRP: Threshold Rook Pivoting

$$
\begin{array}{ccccccc}
\cdot & \cdots & \cdots & \cdots & \cdots & \cdots & \cdot \\
 & \ddots & & & & & \vdots \\
 & & \ddots & & & & \vdots \\
 & & & {\color{red}4.0} & {\color{blue}1.0} & & {\color{blue}7.0} \\
 & & & {\color{blue}2.0} & & & \times \\
 & & & & \times & \times & \\
 & & & {\color{blue}4.0} & \times & \times & \times \\
 & & & & \times & & \times
\end{array}
$$

TCP: Threshold Complete Pivoting

$$
\begin{array}{ccccccc}
\cdot & \cdots & \cdots & \cdots & \cdots & \cdots & \cdot \\
 & \ddots & & & & & \vdots \\
 & & \ddots & & & & \vdots \\
 & & & \color{red}{5.0} & \color{blue}{1.0} & & \color{blue}{7.0} \\
 & & & \color{blue}{2.0} & & & \color{blue}{\times} \\
 & & & & \color{blue}{\times} & \color{blue}{\times} & \\
 & & & \color{blue}{4.0} & \color{blue}{\times} & \color{blue}{\times} & \color{blue}{9.0} \\
 & & & & \color{blue}{\times} & & \color{blue}{\times}
\end{array}
$$

Rank-Revealing Factors

$$A = XDY^T =$$ 

X, Y well-conditioned

$$\mathrm{cond}(A) \approx \mathrm{cond}(D)$$

- SVD UDV^T
- QR with column interchanges QDR
- LU with Rook Pivoting LDU
- LU with Complete Pivoting LDU

- QLP $A = QR = QLP^T$

Stability tolerance τ

$$PAQ = LDU$$

Threshold pivoting bounds elements of L and/or U:

$$\left.\begin{array}{l}\text{TPP}\end{array}\right\} \quad |L_{ij}| \le \tau \approx 100 \text{ or } 10 \text{ or } 5$$

$$\left.\begin{array}{l}\text{TRP}\\ \text{TCP}\end{array}\right\} \quad |L_{ij}|, |U_{ij}| \le \tau \approx 3 \text{ or } 2 \text{ or } 1.1$$

TRP, TCP are more **Rank-Revealing** with low τ:

$$\begin{aligned} \operatorname{cond}(L),\ \operatorname{cond}(U) &< (1+\tau)^n \\ \operatorname{cond}(D) &\approx \operatorname{cond}(A) \end{aligned}$$

The need for rank-revealing LU

$$A = \begin{pmatrix} \delta & 1 & 1 & 1 \\ & \delta & 1 & 1 \\ & & \delta & 1 \\ & & & \delta \end{pmatrix} = LDU \qquad \delta \text{ small}$$

TPP would give $\quad L = I, \quad D = \delta I, \quad \mathrm{rank}(A) = 4$ or 0 (!)

TRP or TCP would give

$$\begin{pmatrix} 1 & 1 & 1 & \delta \\ \delta & 1 & 1 & \\ & \delta & 1 & \\ & & \delta & \end{pmatrix} \approx L \begin{pmatrix} 1 & 1 & 1 & \delta \\ & 1 & 1 & -\delta^2 \\ & & 1 & \delta^3 \\ & & & -\delta^4 \end{pmatrix} \qquad \mathrm{rank}(A) \approx 3$$

Implementing TRP

At each stage of Gaussian elimination:

$$A \leftarrow A - lu^T$$

$$\alpha_j = \text{biggest element in col } j \qquad \begin{bmatrix} \alpha_j \\ \text{x} \\ \text{x} \end{bmatrix}$$

$$\beta_i = \text{biggest element in row } i \qquad [\ \beta_i \quad \text{x} \quad \text{x}\ ]$$

Implementing TCP

At each stage of Gaussian elimination:

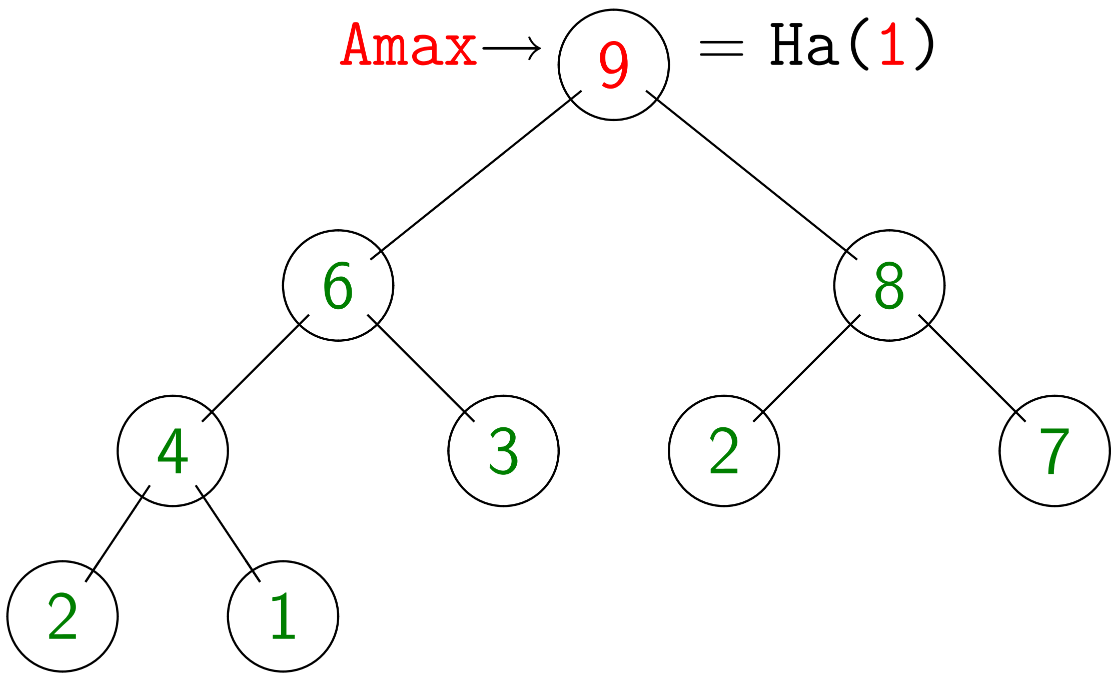
$$A \leftarrow A - lu^T$$

$$\alpha_j = \text{biggest element in col } j \qquad \begin{bmatrix} \alpha_j \\ \text{x} \\ \text{x} \end{bmatrix}$$

$$\texttt{Amax} = \text{biggest element in } A \qquad (= \max\ \alpha_j = \max\ \beta_i)$$

TCP: Store α_j in a Heap

Thanks to John Gilbert



α_j	$=$	Ha(k)	9.0	6.0	8.0	...	
j	$=$	Hj(k)	2	17	1	...	
k	$=$	Hk(j)	3	1	15	...	(location of j in heap)

FACTOR RESULTS

Problem memplus from Harwell-Boeing collection

$A = 18000 \times 18000$, 126000 nonzeros, Scaled

		τ	nnz(L+U)	Time
TCP		100.0	140000	2
		10.0	579000	30
	RR	3.99	2460000	475
	RR	2.5	2890000	6610
	RR	1.5	7080000	27875
TRP		100.0	142000	5
		10.0	141000	5
	RR	3.99	142000	5
	RR	2.50	146000	6
	RR	1.99	166000	7
	RR	1.58	172000	7
	RR	1.26	174000	8
PP (SuperLU, colamd)		1.0	4470000	≈ 250

TCP Profile

Harwell-Boeing Problem memplus

$A = 18000 \times 18000$, 126000 nonzeros

TCP, $\tau = 10.0$, LU $=$ 578000 nonzeros

Markowitz	Find stable pivot	65.0%
Dense CP	600×600	18.5%
Elimination	The algebra	7.4%
Update α_j	for modified cols	4.7%
Update heap		0.1% (!)

Tracking $\max |A_{ij}|$ is easy!

TRP Profile

CUTE Problem BRATU2D

$A = 4900 \times 4900$, 24000 nonzeros

TRP, $\tau = 1.26$, LU $=$ 206000 nonzeros

Update β_i	for modified rows	57.7%
Markowitz	Find stable pivot	31.4%
Elimination	The algebra	4.0%
Dense CP	228×228	2.0%
Update α_j	for modified cols	1.6%

SOLVE

SOLVE

Dense rhs

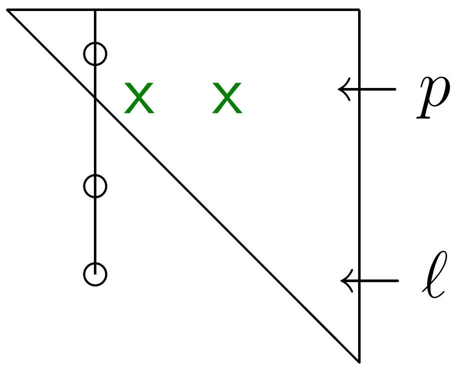
- Currently, $Lx = y$, $L^T x = y$, $\ldots$ assume rhs y is dense

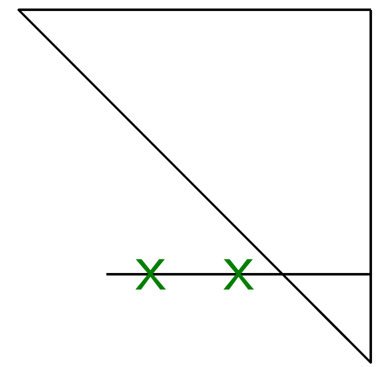
Sparse rhs (future)

- Gilbert and Peierls (1988), CPLEX 7.1 (2001)
- $Lx = y$ requires L **column-wise**
- $L^T x = y$ needs second copy of L (**row-wise**)
- Similarly for U, U^T (not good when updates modify U)
- Product-form update would be ok: $B_k = L_0 U_0 E_1 E_2 \ldots E_k$
- Prefer Block-LU updates

Bartels-Golub updates

à la Reid 1976, 1982, 2004 LA05, LA15

$U' =$ 

$U'' \equiv P^T U' P =$ 

- Avoid Hessenberg matrix
- Use **cyclic permutation**
- Eliminate x using $\begin{pmatrix} 1 & \\ \mu & 1 \end{pmatrix}$ or $\begin{pmatrix} 1 & \\ \mu & 1 \end{pmatrix} \begin{pmatrix} & 1 \\ 1 & \end{pmatrix}$

Block-LU updates

(part of Hanh Huynh's thesis)

Replacing columns, rows, etc is equivalent to solving with a bordered system:

$$\begin{pmatrix} B_0 & V \\ W^T & D \end{pmatrix} = \begin{pmatrix} L_0 & \\ Z^T & I \end{pmatrix} \begin{pmatrix} U_0 & Y \\ & C \end{pmatrix}$$

$$B_0 = L_0 U_0, \qquad L_0 Y = V, \qquad {U_0}^T Z = W$$

Y and Z are likely to be **sparse**

C is small and dense

LUMOD maintains $LC = U$ $\quad (L = \square \quad U = \;)$

APPLICATIONS

LUSOL in MINOS and SQOPT

BR factorization   rank detection for square B

$$B = \boxed{} = LU, \quad PLP^T = \begin{pmatrix} L_1 & \\ L_2 & L_3 \end{pmatrix}, \quad PUQ = \begin{pmatrix} U_1 & U_2 \\ & \ddots \end{pmatrix}$$

TRP or TCP, $\tau \le 2.5$, keep only diag(PUQ)

BS factorization   basis detection for rectangular $W = (B \;\; S)$

$$W^T = \boxed{} = LU, \quad PLP^T = \begin{pmatrix} L_1 & \\ L_2 & I \end{pmatrix}, \quad PUQ = \begin{pmatrix} U_1 \\ 0 \end{pmatrix}$$

TPP, TRP, or TCP $\tau \le 2.5$, keep only P

New B = first m columns of WP^T

Deficient-basis simplex methods

Ping-Qi Pan:

- A revised dual projective pivot algorithm for linear programming, SIOPT, to appear 2005
- A revised primal deficient-basis simplex algorithm for linear programming, SIOPT, submitted June 2005

Take advantage of degeneracy:

$$A = \left[\begin{array}{c|c} \quad B \quad & \qquad\qquad N \qquad\qquad \end{array}\right], \qquad Bx_B = b$$

Thm: $\mathrm{cond}(B) \le \mathrm{cond}(B \;\; a)$ Apply LUSOL to rectangular B

Symmetric indefinite systems

$$A = LDL^T, \qquad D = \begin{pmatrix} \square & & & & & \\ & \square & & & & \\ & & \boxed{} & & & \\ & & & \square & & \\ & & & & \boxed{} & \\ & & & & & \ddots \end{pmatrix} \qquad (1 \times 1, \quad 2 \times 2)$$

- Bunch-Parlett, Bunch-Kaufman strategies don't bound $|L_{ij}|$
- Duff and Reid, Harwell Subroutine Library:

MA27, MA57	**do bound** $|L_{ij}|$
Equivalent to	Threshold Rook Pivoting
Hence:	MA27 and MA57 are rank-revealing

lp_solve

An open-source Linear and Mixed Integer Programming solver

http://groups.yahoo.com/group/lp_solve/

- GNU LGPL, implemented in C, runs on most platforms
- Repository for a C implementation of LUSOL created by Kjell Eikland (F77 $\rightarrow$ Pascal $\rightarrow$ C)
  Factor includes dynamic reallocation of storage
  http://groups.yahoo.com/group/lp_solve/files/LUSOL/
- Choice of BFPs
  LUSOL is now the default

Yin Zhang (Rice Univ, Houston, TX)

29 Nov 2005 BUG REPORT (not all bad)

In lp_solve@yahoogroups.com, Kjell Eikland wrote:

Thank you for submitting this, and I will pass it on to Michael Saunders. What you are reporting is not a typical scenario in linear programming (although lp_solve's presolve can possibly face it), but I guess you are using LUSOL for LU decomposition and straight equation-solving?

Yin Zhang:

Yes, I'm using LUSOL for LU decomposition. For one of my research projects, I need a rank-revealing LU decomposition method that can handle large matrices of size 1,000,000 by 50,000 (with ≈5M nonzeros). The lu() in matlab/umfpack is not rank-revealing. So I wrote a mex wrapper for LUSOL. But then I found the above bug.

MAS:

On such large systems, you should be using TRP – it's much more efficient than TCP and essentially as reliable for rank-detection (as long as the factor tolerance is pretty close to 1).

Yin Zhang:

For some reason, even TCP seems to be pretty fast on my matrices. TRP with 1.1 factor also works very well. This may be due to the structure of my matrices (am working on large-scale network inference, in particular, inferring link delay from end-to-end path delay measurements, so my matrices are "routing matrices", which are highly sparse and most nonzero entries are 1.)

P.S. I'm a big fan of many of your packages. A couple of years back I did some work with Dave Donoho on traffic matrix estimation and used the PDSCO package. It worked really well, and we acknowledged you in the paper.

SUMMARY

LUSOL features

Square or rectangular A	
Normal sparse LU	Threshold Partial Pivoting
Rank-revealing LU	for "basis repair"
	Threshold Rook Pivoting
	Threshold Complete Pivoting
Stable updates	add, replace, delete, rank-one
	Bartels-Golub style

Future Tasks

FACTOR	Improve β_i (max element in each row)
	Special handling of dense columns
SOLVE	Sparse rhs's
UPDATE	Block-LU with new black-box FACTORs
	(F90, Hanh's thesis)
Language	F77 $\rightarrow$ C always possible via f2c
	F77 $\rightarrow$ Pascal $\rightarrow$ C (Kjell Eikland, lp_solve)
	F90 $\rightarrow$ C? (NAG F95 compiler?)
	Fmex (Mike O'Sullivan), Cmex (Yin Zhang)

COIN-OR project

Future Tasks

FACTOR	Improve β_i (max element in each row) Special handling of dense columns
SOLVE	Sparse rhs's
UPDATE	Block-LU with new black-box FACTORs (F90, Hanh's thesis)
Language	F77 $\rightarrow$ C always possible via f2c F77 $\rightarrow$ Pascal $\rightarrow$ C (Kjell Eikland, lp_solve) F90 $\rightarrow$ C? (NAG F95 compiler?) Fmex (Mike O'Sullivan), Cmex (Yin Zhang)

COIN-OR project

Documentation