On solving mixed sparse-dense linear least-squares problems Part I: A Schur complement approach

Jennifer Scott

University of Reading and STFC Rutherford Appleton Laboratory

Miroslav Tůma

Charles University and Academy of Sciences of the Czech Republic

Sparse Days at CERFACS, 6–8 September 2017

An aside ...

- With the exception of 2002 and 2014, I have attended all the *Sparse Days at CERFACS* meetings between 2001–2017 and, apart from 2006, I have given a talk each time.
- I also attended and presented at each of the wonderful St. Girons Conferences (1994, 2003, 2015).
- So 16 meetings ... not sure how many can equal or beat this (apart from Iain!)?
- HUGE thank you to Iain for all the great opportunities these meetings have offered.

Introduction

Least squares are used across a wide range of disciplines:

- simple curve fitting
- estimation of satellite image sensor characteristics
- powering internet mapping services
- exploration seismology
- NMR spectroscopy
- ultrasound for medical imaging
- aerospace systems
- neural networks
- data assimilation for weather forecasting and climate modelling

Introduction

Linear least squares (LS) arises on its own and as subproblem of nonlinear least squares problem.
Solve

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad \|Ax - b\|_2,$$

where $A \in \mathbb{R}^{m \times n}$ with $m \geq n$ is large and sparse and $b \in \mathbb{R}^m$.

Focus today on A having small number of "dense" rows.

The LS solution x satisfies the $n \times n$ *normal equations*

$$Cx = A^T b, \qquad C = A^T A.$$

If A is full rank, C is positive definite so use sparse Cholesky solver eg CHOLMOD (Davis), or `HSL_MA87` (Hogg, Reid and Scott) to compute factorization

$$C = LL^T,$$

where L is lower triangular.

Solve $Lz = A^T b$ then $L^T x = z$.

The LS solution x satisfies the $n \times n$ *normal equations*

$$Cx = A^T b, \qquad C = A^T A.$$

If A is full rank, C is positive definite so use sparse Cholesky solver eg CHOLMOD (Davis), or `HSL_MA87` (Hogg, Reid and Scott) to compute factorization

$$C = LL^T,$$

where L is lower triangular.

Solve $Lz = A^T b$ then $L^T x = z$.

Or use sparse QR factorization eg SPQR (Davis) or qr_mumps (Buttari et al).

Now suppose

$$A = \begin{pmatrix} A_s \\ A_d \end{pmatrix}$$

where $A_s \in \mathbb{R}^{m_s \times n}$ is sparse but rows of $A_d \in \mathbb{R}^{m_d \times n}$ are dense ($m_s \gg m_d$). Then

$$C = \begin{pmatrix} A_s^T & A_d^T \end{pmatrix} \begin{pmatrix} A_s \\ A_d \end{pmatrix} = A_s^T A_s + A_d^T Ad$$

And so C is dense and sparse Cholesky or QR cannot be used.

This talk: A Schur complement approach

Next talk: A preconditioned conjugate gradient method

A Schur complement approach

$$\min_x \left\| \begin{pmatrix} A_s \\ A_d \end{pmatrix} x - \begin{pmatrix} b_s \\ b_d \end{pmatrix} \right\|_2 .$$

This is equivalent to solving the order $m + n$ augmented system

$$\begin{pmatrix} I & & A_s \\ & I & A_d \\ A_s^T & A_d^T & 0 \end{pmatrix} \begin{pmatrix} r_s \\ r_d \\ x \end{pmatrix} = \begin{pmatrix} b_s \\ b_d \\ 0 \end{pmatrix},$$

where r is the residual vector

$$r = \begin{pmatrix} r_s \\ r_d \end{pmatrix} = \begin{pmatrix} b_s \\ b_d \end{pmatrix} - \begin{pmatrix} A_s \\ A_d \end{pmatrix} x.$$

Important: We assume throughout that A has been scaled by normalising each column by its 2-norm so that $|a_{ii}| = 1$ and $|a_{ij}| \leq 1$.

Eliminate first m_s rows and columns to obtain the reduced augmented system of order $(m_d + n) \times (m_d + n)$

$$K \begin{pmatrix} x \\ r_d \end{pmatrix} = \begin{pmatrix} -A_s^T b_s \\ b_d \end{pmatrix}, \qquad K = \begin{pmatrix} -C_s & A_d^T \\ A_d & I \end{pmatrix}.$$

$C_s = A_s^T A_s$ is the reduced normal matrix.

Eliminate first m_s rows and columns to obtain the reduced augmented system of order $(m_d + n) \times (m_d + n)$

$$K \begin{pmatrix} x \\ r_d \end{pmatrix} = \begin{pmatrix} -A_s^T b_s \\ b_d \end{pmatrix}, \qquad K = \begin{pmatrix} -C_s & A_d^T \\ A_d & I \end{pmatrix}.$$

$C_s = A_s^T A_s$ is the reduced normal matrix.

Compute Cholesky factorization $C_s = L_s L_s^T$. Then

$$K = \begin{pmatrix} L_s & \\ B_d & I \end{pmatrix} \begin{pmatrix} -I & \\ & S_d \end{pmatrix} \begin{pmatrix} L_s^T & B_d^T \\ & I \end{pmatrix},$$

where B_d and the Schur complement S_d are given by

$$L_s B_d^T = -A_d^T \quad \text{and} \quad S_d = I + B_d B_d^T.$$

Solve $L_s B_d^T = -A_d^T$ by forward substitution.

Then compute $S_d = I + B_d B_d^T$ using dense linear algebra.

Then solve the reduced augmented system by solving

$$\begin{pmatrix} -L_s & \\ -B_d & S_d \end{pmatrix} \begin{pmatrix} y_s \\ y_d \end{pmatrix} = \begin{pmatrix} -A_s^T b_s \\ b_d \end{pmatrix},$$

followed by

$$\begin{pmatrix} L_s^T & B_d^T \\ & I \end{pmatrix} \begin{pmatrix} x \\ r_d \end{pmatrix} = \begin{pmatrix} y_s \\ y_d \end{pmatrix}.$$

This reduces to solving

$$\begin{aligned} L_s y_s &= A_s^T b_s, \\ S_d r_d &= b_d + B_d y_s, \\ L_s^T x &= y_s - B_d^T r_d. \end{aligned}$$

Thus dominant costs are

- sparse factorization of $m_s \times m_s$ matrix
- triangular solves
- (small) dense Cholesky factorization.

This reduces to solving

$$\begin{aligned} L_s y_s &= A_s^T b_s, \\ S_d r_d &= b_d + B_d y_s, \\ L_s^T x &= y_s - B_d^T r_d. \end{aligned}$$

Thus dominant costs are

- sparse factorization of $m_s \times m_s$ matrix
- triangular solves
- (small) dense Cholesky factorization.

What about using an iterative solver in place of a direct solver?

Preconditioning the Schur complement approach

Right-preconditioned reduced augmented system is

$$KM^{-1}\begin{pmatrix} w_s \\ w_d \end{pmatrix} = \begin{pmatrix} -A_s^T b_s \\ b_d \end{pmatrix}, \qquad M\begin{pmatrix} x \\ r_d \end{pmatrix} = \begin{pmatrix} w_s \\ w_d \end{pmatrix},$$

where M is the chosen preconditioner.

Let $C_s \approx \tilde{L}_s \tilde{L}_s^T$ be an incomplete Cholesky factorization.

Then set

$$M = \begin{pmatrix} \tilde{L}_s & \\ \tilde{B}_d & I \end{pmatrix} \begin{pmatrix} -I & \\ & \tilde{S}_d \end{pmatrix} \begin{pmatrix} \tilde{L}_s^T & \tilde{B}_d^T \\ & I \end{pmatrix},$$

where

$$\tilde{L}_s \tilde{B}_d^T = -A_d^T \quad \text{and} \quad \tilde{S}_d = I + \tilde{B}_d \tilde{B}_d^T.$$

We will refer to M as the block IC preconditioner.

Application of block IC preconditioner M

Input: $\tilde{L}_s$, $\tilde{B}_d$, the Cholesky factors of $\tilde{S}_d$, and $z = \begin{pmatrix} z_s \\ z_d \end{pmatrix}$.

Output: $y = \begin{pmatrix} y_s \\ y_d \end{pmatrix} = M^{-1} z$.

1. Solve $\tilde{L}_s u_s = -z_s$.
2. Compute $u_d = z_d + \tilde{B}_d u_s$.
3. Solve $\tilde{S}_d y_d = u_d$ using Cholesky factors of $\tilde{S}_d$.
4. Form $u_s \Leftarrow u_s - \tilde{B}_d^T y_d$.
5. Solve $\tilde{L}_s^T y_s = u_s$.

Problem of null columns

When A is partitioned into sparse and dense parts, A_s often contains some null columns.

In this case, $C_s = A_s^T A_s$ is rank deficient and Cholesky factorization breaks down.

How to deal with this?

$$A = \begin{pmatrix} A_1 & A_2 \end{pmatrix} \equiv \begin{pmatrix} A_{s_1} & 0 \\ A_{d_1} & A_{d_2} \end{pmatrix}$$

with $A_1 \in \mathbb{R}^{m \times n_1}$ and $A_2 \in \mathbb{R}^{m \times n_2}$ $(n = n_1 + n_2,\ n_2 \ll n_1)$.

Lemma (Scott and Tůma '17)

Let $z \in \mathbb{R}^{n_1}$ and $W \in \mathbb{R}^{n_1 \times n_2}$ be the solutions to

$$\min_z \|A_1 z - b\|_2 \quad \text{and} \quad \min_W \|A_1 W - A_2\|_F .$$

Then the solution of the original LS problem is

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} z - W x_2 \\ x_2 \end{pmatrix}$$

with

$$x_2 = (A_2^T (A_2 - A_1 W))^{-1} A_2^T (b - A_1 z).$$

Notes

Once we have solved $min_z \|A_1 z - b\|_2$, what is the extra work in solving $min_W \|A_1 W - A_2\|_F$?
Recall

$$\begin{aligned} L_{s1} y_s &= A_s^T b_s, \\ S_d r_d &= b_d + B_d y_s, \\ L_{s1}^T z &= y_s - B_d^T r_d. \end{aligned}$$

The columns of A_2 are $\begin{pmatrix} 0 \\ a_2 \end{pmatrix}$ and so, for each, solve

$$\begin{aligned} S_d r_d &= a_2, \\ L_{s1}^T w &= B_d^T r_d. \end{aligned}$$

Provided n_2 not large, this is cheap for a direct solver (solve simultaneously for all m_d cols).

With null columns removed, A_{s1} can still be rank deficient.

Propose employing a shift $\alpha > 0$

$$C_{s1}(\alpha) = A_{s1}^T A_{s1} + \alpha I = L_{s1}(\alpha) L_{s1}(\alpha)^T.$$

so that

$$C_{s1} \approx L_{s1}(\alpha) L_{s1}(\alpha)^T.$$

Employ the factor $L_{s1}(\alpha)$ to obtain a preconditioner to solve the original problem (hybrid method that combines a direct solver with an iterative one!).

Numerical experiments

- For sparse Cholesky, `HSL_MA87` (Hogg, Reid and Scott '10) (DAG-based code for multicore machines).
- For dense Cholesky, `dpotrf`.
- For incomplete Cholesky, `HSL_MI35` (Scott and Tuma '14).
- Use preconditioned GMRES as iterative solver.
- Experiments performed using 8× Intel i7-4790 (3.6 GHz) with 15.6 Gbytes memory.
- Timings are elapsed times in seconds.

Example 1: PDE1

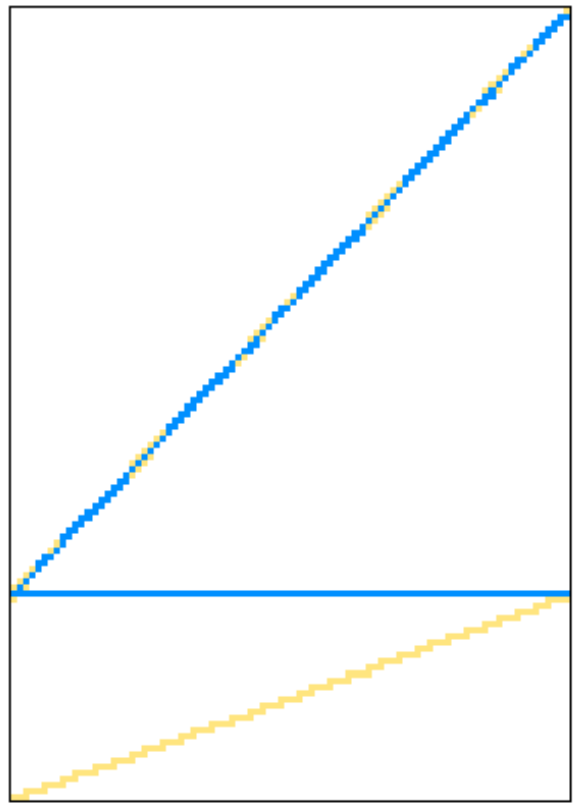
- $m = 271,792, n = 270,595$
- Single row with 66% density
- C is essentially dense. Not able to run direct solver.

Remove dense row:

- A_s is full rank.
- $density(C_s) = 4.5 \times 10^{-5}$
- $nnz(L) = 2 \times 10^7$
- Schur complement direct solver solution time 1.21 seconds.

Example 2: scagr7-2b

$m = 13,247, n = 9,743$



A row of A is defined to be dense if its density is at least ρ.

m_d is the number of dense rows.

n_2 is the number of null cols in A_s.

ρ	0	0.1	0.01	0.001
m_d	0	1	7	263
n_2	0	1	1	257
$density(C_s)$	$4.15\times^{-2}$	1.45×10^{-2}	1.73×10^{-4}	5.74×10^{-4}
$nfactor$	8.64×10^6	5.11×10^6	1.62×10^5	1.45×10^5
$nflops$	1.08×10^{10}	3.55×10^9	3.81×10^6	3.21×10^6
Time	1.19	0.29	0.08	0.49 (0.33*)

* Time for the solves needed to deal with the 257 null cols.

Note: C does not have to be very dense for it to be advantageous to use the Schur complement approach.

Same example: Schur complement with GMRES + block IC

ρ	0	0.1	0.01	0.001
m_d	0	1	7	263
n_2	0	1	1	257
$density(C_s)$	$4.15\times^{-2}$	1.45×10^{-2}	1.73×10^{-4}	5.74×10^{-4}
Times:				
Compute IC	0.34	0.39	0.02	0.05
GMRES (Iters)	0.31 (370*)	0.08 (99)	0.001 (3)	0.012 (2)
		0.05 (73)	0.001 (2)	2.94 (2)
Total	0.65	0.47	0.019	3.00

* For $\rho = 0$ use IC preconditioned LSMR (dense rows ignored).

If n_2 large, better to ignore these null columns and use shift.
For $\rho = 0.001$ this reduces total time to 0.07.

Appending single "dense" row: LSMR with IC preconditioner

Problem	m	n	Its	T_p	T_s	T_{total}
$\rho = 0.01$						
GL7d20	1,911,124	1,437,546	40	166	19	185
LargeRegFile	2,111,154	801,374	70	127	7.7	134
relat9	9,746,232	274,667	90	20	29	49
$\rho = 0.1$						
GL7d20			–	>600	–	>600
LargeRegFile			–	>600	–	>600
relat9			90	64	29	92
$\rho = 0.5$						
GL7d20			–	>600	–	>600
LargeRegFile			–	>600	–	>600
relat9			–	>600	–	>600

ρ is the density of the added row.

Appending single dense row ($\rho = 1.0$)

Cannot solve these examples using LSMR + IC as IC is prohibitively expensive (in time) to compute.

Results for Schur complement using GMRES + block IC:

	Its	T_p	T_s	T_{total}
GL7d20	32	187	39	226
LargeRegFile	11	1.61	0.53	2.14
CONT11_L	142	9.23	22.8	32.0

Appending $m_d \geq 1$ dense rows ($\rho = 1.0$)

Results for Schur complement using GMRES + block IC:

Problem	$m_d = 1$		$m_d = 50$		$m_d = 100$	
	Its	T_{total}	Its	T_{total}	Its	T_{total}
`GL7d20`	32	226	22	222	22	231
`LargeRegFile`	11	2.14	9	3.28	9	4.83
`CONT11_L`	142	32.0	12	13.0	13	15.9

Increase in time with m_d is because the dense factorization of S_d becomes more expensive (it is trivial with $m_d = 1$).

Merci de votre attention!

Paper/reports on solving LS problems available from
http://www.numerical.rl.ac.uk/people/j_scott/

Project supported by EPSRC grant EP/M025179/1

Finally, at RAL we have two vacancies at post doc level: one to join Iain in the NLAFET project and the other as a numerical analysis researcher to work in part on sparse (non)linear least squares. **Please ask if you or someone you know might be interested!**