

A Parallel Implementation of the Spectral Bundle Method for SDP

Madhu V. Nayakkankuppam

Joint work with Yevgen Tymofyeyev

Department of Mathematics & Statistics, UMBC

Acknowledgments

- Jorge Moré and Argonne National Lab
- UMBC's Designated Research Initiative Fund

SDP Vs. EVO

$$\text{(SDP)} \quad \min \langle C, X \rangle \quad \text{s.t.} \quad AX = b; \quad X \succeq 0.$$

When feasible set is compact, the dual of (SDP) is equivalent to

$$\text{(EVO)} \quad \min_{y \in \mathbb{R}^m} \lambda_{\max}(A^*y - C) - \langle b, y \rangle =: f(y),$$

which is unconstrained minimization of a nonsmooth, convex function. The proximal bundle method [Kiwiel, '90] can be used to solve EVO [Helmberg & Rendl, '99]. With a cutting plane model for $f(x)$, proximal model:

$$\phi(y; y^k) = \min_{y \in \mathbb{R}^m} \left\{ \max_{g \in G \subseteq \partial_\epsilon f(y)} f(y^k) + \langle g, y - y^k \rangle \right\} + \frac{\rho}{2} \|y - y^k\|^2.$$

Subproblem in each bundle iteration:

$$\min_{y \in \mathbb{R}^m} \phi(y; y^k).$$

Kiwiel's Proximal Bundle Method

- Set $k = 0$, and start with y^k, G^k .
- Let (\bar{y}, \bar{g}) be the optimal solution to the subproblem of minimizing the cutting plane model.
- If $f(y^{k+1}) - f(y) < \delta$ (given), STOP.
- If \bar{y} produces “significant descent”, set $y^{k+1} = \bar{y}$ (serious step); otherwise set $y^{k+1} = y^k$ (null step).
- Add one or more subgradients from $\partial f(\bar{y})$ to G^k (“aggregating”, if necessary) to obtain G^{k+1} . Go to Step 2.

The subsequence of $\{y^k\}$ of serious steps is a minimizing sequence for f .

Subgradient Computation

With $Z = A^*y - C$,

$$\partial_\epsilon f(y) = \{AW - b : W \succeq 0, \text{tr}(W) = 1, \langle W, Z \rangle \geq \lambda_{\max}(Z) - \epsilon\}.$$

Computing ϵ -subgradients requires “ ϵ -eigenvectors” of $\lambda_{\max}(Z) = \lambda_{\max}(A^*y - C)$. In principle, this is straightforward with the Lanczos method which only requires matrix-vector products

$$w = Zu.$$

But handling block diagonal structure is tricky: Lanczos will converge to eigenvectors **without** block structure when a multiple eigenvalue is split across different blocks.

Block-Structured Lanczos

- 1: Given Z with s diagonal blocks, l, d ($1 < l < d$), $u_0, w_0 = Zu_0$.
- 2: Set $\text{ACTIVE} = \{1, \dots, s\}$.
{Start synchronized block-wise Lanczos processes}
- 3: Set $j = 0$.
- 4: **repeat**
- 5: **if** length of Lanczos factorization is d **then**
- 6: **(R)** Perform implicit restart to compress to l vectors.
- 7: **end if**
- 8: **for** each block $k \in \text{ACTIVE}$ **do**
- 9: Obtain u_{j+1}^k from u_j^k , resulting in a Lanczos factorization of length $j + 1$.
- 10: **(CO)** Reorthogonalize, if necessary.
- 11: **(MV)** Compute $w_{j+1} = Zu_{j+1}$.
- 12: **(ABS)** Compute residual error bounds; update ACTIVE .
- 13: **end for**
- 14: **until** convergence.

Parallelization

Data Distribution

- A_i 's distributed such that work in matrix-vector products equitably shared.
- Polyhedral part of bundle distributed just like the A_i 's.
- Semidefinite part of bundle stored on all processors.

Parallel Portion

- Matrix-vector products in Lanczos.
- Forming data for subproblem.

Serial Portion

- Lanczos reorthogonalization.
- Lanczos implicit restarts.
- Subproblem solution, descent test, iterate update, aggregation.

Results on Medium-Sized Problems

Prob	m	blk	Description	Hardware
r1	16,384	[500, 500, 500, 500]	Random (sparsity 1.0e-06)	All runs on Chiba City, Argonne's 512-CPU cluster
r2	8,192	150 blocks of size 50	Random (sparsity 1.0e-03)	
th1	19,899	2000	Lovász ϑ -function	
th2	67,489	5000	Lovász ϑ -function	
q20	98,556	[20, 190, 190, 400]	Quantum chemistry	

Prob	# blocks	time for e.v.		% savings
		off	on	
r1	4	1:58:54	1:31:32	23
r2	50	2:29:00	1:48:56	27
q20	4	1:53:15	1:48:14	5

Prob	1	2	4	8	16	32
r1	23:30:24	12:14:35	07:48:34	04:07:26	02:45:44	2:19:39
r3	13:39:38	07:09:08	05:41:31	02:35:32	01:34:26	1:35:18
th1	01:51:00	00:54:25	00:30:00	00:19:08	00:11:30	0:12:12
th2	—	—	—	10:33:03	07:31:26	4:05:58
q20*	—	—	15:19:14	04:31:22	03:32:32	1:40:28

*: lower bound only.

Looking Ahead

General improvements

- Better speedups with Myrinet
- Better use of MPI (overlapped communication and computation?)
- Convex quadratic subdifferential models?

Lanczos

- Selective Orthogonalization.
- Block Lanczos methods.
- Chebyshev acceleration?? Not a clear advantage in parallel, where matrix-vector products are expensive!
- Abandon Lanczos. Jacobi parallelizes better!