Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación

Lecturas en Ciencias de la Computación
ISSN 1316-6239

Accelerated Cimmino's method for saddle point problems

Luis Manuel Hernández-Ramos

RT 2008-06

Centro de Cálculo Científico y Tecnológico de la UCV
CCCT-UCV
Caracas, Octubre, 2008.

Accelerated Cimmino's method for saddle point problems

Luis Manuel Hernández-Ramos [*†]

October 13, 2008

Abstract

We solve saddle point problems by accelerated versions of the classical Cimmino's method. For that, saddle point problems are reviewed and reformulated as best approximation problems. In that setting, low-cost optimization techniques are used for accelerating Cimmino's method. Encouraging numerical results coming from KKT systems, Stokes problems and domain decomposition problems are presented.

AMS: .
Keywords: Saddle point problem, Cimmino's method, Conjugate gradient method, Spectral gradient method.

1 Introduction

In recent years, saddle point problems have been gaining popularity in many areas of scientific computing. They arise in many applications such as: KKT systems in optimization, mixed formulations in fluid dynamics and domain decomposition discretizations for the parallel solution of Partial Differential Equations (PDE) [4, 12, 23]. Parallel computing is becoming an effective way for solving large-scale numerical problems that arise in scientific and engineering applications. However, algorithms for solving saddle point problems are not well developed for parallel machines. Cimmino's methods is a suitable technique for solving linear systems in many processors due to the fact that it is inherently parallel. Nevertheless, the original Cimmino's algorithm can be too slow to be considered as a practical and efficient method [3, 6, 7, 9].

In this work, we reformulate the saddle point problem as computing an orthogonal projection onto the intersection of many suitable subspaces. A natural choice to solve that formulation, in parallel architectures, is Cimmino's method. However, since it can be slow, we rewrite this problem as a convex quadratic

[*]Centro de Calculo Cientifico y Tecnologico, Facultad de Ciencias, Universidad Central de Venezuela, Ap. 47002, Caracas 1041-A, Venezuela.
[†]e-mail: luis.hernandez@ciens.ucv.ve

minimization problem, and inherently parallel Cimmino's versions of some optimization algorithms, such as the conjugate gradient method and the spectral gradient method (also known as the Barzilai-Borwein method) are proposed and adapted. Finally, some numerical examples with saddle point problems coming from KKT systems, Stokes problems and domain decomposition problems are presented.

2 Preliminaries

We are interested in solving the system of linear equations:

$$\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix}\begin{pmatrix} x \\ \lambda \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}, \tag{1}$$

where

$(H1)$ $B \in \Re^{m\times n}$ is a matrix with $rank(B) = m, m \leq n$,

$(H2)$ $A \in \Re^{n\times n}$ is a symmetric positive definite matrix.

Proposition 2.1 *Under the hypotheses (H1,H2) the system (1) has a unique solution* $(x, \lambda)^T$ *which satisfies:*

$$\begin{cases} f - Ax \perp \ker B, \\ x \in \ker B. \end{cases} \tag{2}$$

Proof
Under hypotheses (H1,H2) the system:

$$\begin{pmatrix} A & B^T \\ 0 & BA^{-1}B^T \end{pmatrix}\begin{pmatrix} x \\ \lambda \end{pmatrix} = \begin{pmatrix} f \\ BA^{-1}f \end{pmatrix}, \tag{3}$$

is equivalent to (1) and $BA^{-1}B^T$ is a positive definite matrix. Consequently the system (1) has a unique solution [19]. ∎

Proposition 2.2 *Under hypotheses (H1,H2), if* $(x, \lambda)^T$ *solves (1), then the vector* x *corresponds to the orthogonal projection of* $x_u = A^{-1}f$ *onto* $\ker B$*, in the scalar product* $\langle .,. \rangle_A$*.*

Proof
From (1) it is easy to check that the solution x verifies $f - Ax = B^T\lambda \perp \ker B$. Let V be the linear variety defined by:

$$V = \{x/f - Ax \perp \ker B\}. \tag{4}$$

Under the hypotheses $(H1, H2)$, the solution $(x, \lambda)^T$ of (1) is such that x is the only element in the intersection of V and $\ker B$,

$$\{x\} = V \cap \ker B. \tag{5}$$

Let $x_u = A^{-1}f$, the system (2) becomes

$$\begin{cases} \forall y \in \ker B, \langle A(x_u - x), y\rangle = 0, \\ x \in \ker B. \end{cases} \tag{6}$$

When A is a symmetric and positive definite matrix, the mapping $\{x, y\} \mapsto \langle Ax, y\rangle$ is a scalar product denoted by $\langle ., .\rangle_A$ and the system (1) is equivalent to:

$$\begin{cases} x_u = A^{-1}f \\ x_u - x \perp_A \ker B, \\ x \in \ker B. \end{cases} \tag{7}$$

It means that x is the orthogonal projection of $x_u = A^{-1}f$ onto $\ker B$, in the scalar product $\langle ., .\rangle_A$. The notation $x_u - x \perp_A \ker B$ means $\langle x_u - x, y\rangle_A = 0 \ , \forall y \in \ker B$. ∎

In this case we can write the linear variety as $V = \{x_u\} + (\ker B)^{\perp_A}$ [19].

3 Saddle point resolution by Cimmino's method

Let us partition the matrix $B \in \Re^{m\times n}$ into r row blocks:

$$B^T = [B_1^T, B_2^T, \dots B_r^T].$$

Then,

$$\ker B = \bigcap_{i=1}^{r} \ker B_i.$$

From proposition (2.2), it follows that the solution x for the saddle point problem (1) is the A-orthogonal projection of $x_u = A^{-1}f$ onto the subspace $\ker B = \bigcap_{i=1}^{r} \ker B_i$. Hence, this problem can be viewed as a Best Approximation Problem (BAP) in the scalar product $\langle ., .\rangle_A$ and can be solved by Cimmino's method [3, 9]:

$$x_{k+1} = \frac{1}{r}\sum_{i=1}^{r} P_{M_i} x_k. \tag{8}$$

For the saddle point problem (1), the operator P_{M_i} becomes the A-orthogonal projection onto M_i. As each $\ker B_i$ is also a subspace, then we can calculate the A-projection of any vector $y \in \Re^n$ onto $\ker B_i$, $P_{M_i}y$, by computing x from the smaller saddle point problem:

$$\begin{pmatrix} A & B_i^T \\ B_i & 0 \end{pmatrix} \begin{pmatrix} x \\ \lambda \end{pmatrix} = \begin{pmatrix} Ay \\ 0 \end{pmatrix}, \tag{9}$$

These smaller saddle point problems can be solved by classical techniques. The Schur complement matrix $B_i A^{-1} B_i^T$ associated to (9) has dimensions $m_i \times m_i$, where m_i is the number of rows of block B_i. Consequently, the condensed system

$B_iA^{-1}B_i^T\lambda_i = B_iy$ is much easier to solve. [4, 18, 19]. Cimmino's method is highly parallelizable. For this case, at each iteration, the A-projection onto each subspace $\ker B_i$ can be performed independently for $i = 1, \ldots, r$. However, it is known that Cimmino's methods can be too slow. Therefore, in this work we propose to use some accelerated versions of this method that will be discussed in the following sections.

4 Acceleration of Cimmino's method

In a more general setting, let $M_i,\ i = 1, \ldots, r$, be closed subspaces of a Hilbert space H. Let P_{M_i} be the orthogonal projection onto M_i. We consider the following function to minimize:

$$f(x) = \frac{1}{2}\sum_{i=1}^{r} \|x - P_{M_i}x\|_A^2 = \frac{1}{2}\sum_{i=1}^{r} \|(I - P_{M_i})x\|_A^2. \tag{10}$$

The function f is a non-negative quadratic function called *proximity function* [5]. If a vector $x \in \cap_{i=1}^{r} M_i$, then $f(x) = 0$ and x minimizes f.
For using minimization techniques, we need the gradient of f,

$$\nabla f(x) = \sum_{i=1}^{r} x - P_{M_i}x, \tag{11}$$

and the Hessian,

$$\nabla^2 f(x) = \sum_{i=1}^{r} I - P_{M_i}. \tag{12}$$

It is easy to check that in this case the Hessian operator is a constant positive semidefinite matrix. In fact,

$$\langle x, \nabla^2 f x\rangle_A = \sum_{i=1}^{r} \|x - P_{M_i}x\|_A^2 \geq 0.$$

Additionally $\langle x, \nabla^2 f x\rangle_A = 0$ if and only if $x \in \cap_{i=1}^{r} M_i$. Consequently, we observe that the classical Cimmino's method (8) is the gradient method with a constant step length :

$$x_{k+1} = x_k - \frac{1}{r}\nabla f(x).$$

In general, the gradient method with a constant step length is slower than the classical steepest descent method (or Cauchy method) which is famous for being too slow. That explains the well-known slowness of Cimmino's method. Acceleration of Cimmino's method can be performed by using another efficient optimization techniques for minimizing the convex quadratic $f(x)$ [1, 24]. In this work, we compare the classical Cimmino's method with accelerated versions based on the classical conjugate gradient method and the Barzilai-Borwein method [2, 10, 16, 21, 22].

5 Relationship between Hessian spectra and geometry of the problem

In this section we show an interesting theoretical results that relates certain Rayleigh quotients of the Hessian matrix with the geometry of the problem. The convergence of methods designed for minimizing the quadratic function,

$$f(x) = \frac{1}{2}\sum_{i=1}^{r} \|x - P_{M_i}x\|_A^2 = \frac{1}{2}\sum_{i=1}^{r} \|(I - P_{M_i})x\|_A^2 \tag{13}$$

are usually related to the spectral properties of the Hessian matrix

$$\nabla^2 f = \sum_{i=1}^{r} I - P_{M_i}. \tag{14}$$

For any vector $x \in H$, the Rayleigh quotient $R(x)$ in the A-product is given by:

$$R(x) = \frac{\langle x, \nabla^2 f x\rangle_A}{\langle x, x\rangle_A}.$$

It is well-known that

$$\lambda_{min} \leq R(x) \leq \lambda_{max},$$

where λ_{min} and λ_{max} are the smallest and the largest eigenvalues of the Hessian $\nabla^2 f$, respectively. We denote by $P_{M_i^{\perp_A}} = I - P_{M_i}$ the A-orthogonal projection onto $\ker B_i^{\perp_A}$, which is an idempotent and auto-adjoint operator in the product $\langle ., .\rangle_A$ [14]. Therefore,

$$\begin{aligned}
\langle x, \nabla^2 f x\rangle_A &= \langle x, \sum_{i=1}^{r} x - P_{M_i}x\rangle_A \\
&= \sum_{i=1}^{r}\langle x, x - P_{M_i}x\rangle_A \\
&= \sum_{i=1}^{r}\langle x, P_{M_i^{\perp_A}}x\rangle_A.
\end{aligned}$$

and

$$\begin{aligned}
\sum_{i=1}^{r}\langle x, P_{M_i^{\perp_A}}x\rangle_A &= \sum_{i=1}^{r}\langle P_{M_i^{\perp_A}}x, P_{M_i^{\perp_A}}x\rangle_A \\
&= \sum_{i=1}^{r} \|P_{M_i^{\perp_A}}x\|_A^2.
\end{aligned}$$

Hence, for all $x \in H - \{0\}$, the Rayleigh quotient becomes,

$$R(x) = \sum_{i=1}^{r} \frac{\|P_{M_i^{\perp_A}}x\|_A^2}{\|x\|_A^2}.$$

When M is a closed subspace of a Hilbert space, we have for all $x \in H - \{0\}$,

$$\cos(span\{x\}, M_i) = \begin{cases} \|P_M(x)\|/\|x\| & \text{if } x \notin M \\ 0 & \text{if } x \in M \end{cases}$$

[11].
As a consequence, for all $x \in H - \{0\}$ we obtain,

$$R(x) = \sum_{i=1}^{r} \sin^2(span\{x\}, M_i), \tag{15}$$

where $\sin(span\{x\}, M_i)$ is the sine of the angle between $span\{x\}$ and M_i (in the $\langle ., . \rangle_A$ product).
Hence, we have,

$$s \leq R(x) \leq r,$$

where

$$s = \min_{i \neq j} \sin^2(M_j \ , M_i), \ \ i, j = 1, \dots, r. \tag{16}$$

Where $\sin(M_j \ , M_i)$ is the sine of the angle between subspaces M_j and M_i [11].

6 Numerical Experiments

We apply Cimmino's methods and its accelerated versions (Barzilai-Borwein-Cimmino and Conjugate-Gradient-Cimmino) to solve:

- A selection of saddle point problems from the CUTEr collection [17].
- A set of Stokes saddle point problems generated using IFISS incompressible flow software associated with the book by Elman et al. [13] (stokes_testproblems):
  - STOKES1: Channel domain with natural outflow boundary.
  - STOKES2: Flow over a backward facing step.
  - STOKES3: Lid driven cavity.
  - STOKES4: Colliding flow.
- Several domain decomposition saddle point problems discussed in [19] (DD1, DD2, DD3 and DD4).

All our experiments were run on a Pentium IV using MATLAB 7.6. We compare accelerated Cimmino's versions versus CG-Uzawa and CG-AOP method. CG-Uzawa is the conjugate gradient method over the condensed system (also known as the Schur complement system) $BA^{-1}B^T\lambda = BA^{-1}F$ [4]. CG-AOP method is a preconditioned version of CG-Uzawa [19]. The preconditioner used by this method is $((B^{\dagger})^T AB^{\dagger})$, where $B^{\dagger}$ represents the pseudo-inverse of B.

For all sets of problems, the matrix $A \in \Re^{n\times n}$ is symmetric and positive definite, and $B \in \Re^{m\times n}$ is a full row rank matrix. For the CUTEr problems, and also for Stokes problems we set the block $(2,2)$ of the saddle point matrix to be the zero matrix, for obtaining a system like (1).

All experiments were obtained by a one row block partition of B $(r = m)$. This case presents an important advantage because the Schur complement matrix of system (9), (needed for computing $P_{M_i},\ i = 1,\dots r$) has dimensions 1×1. The Schur complement matrix is computed by solving linear systems $Aw_i = B_i^T,\ i = 1,\dots,r$ (that can be obtained in parallel). These are the only linear systems solved in the algorithm. It is not necessary to solve another linear system with the matrix A for computing P_{M_i}.

6.1 Results

Table 1 shows the convergence behavior of the different accelerations of Cimmino's method. We can see that the accelerated versions are clearly better than the classical Cimmino's method, in number of iterations and cpu-time. For all problems, the conjugate gradient version of Cimmino's method have the best performance for all considered accelerations. Figures 1,2, and 3 show the evolution of the norm of the gradient by iteration in the accelerated versions for a CUTEr problem, a Stokes problem, and a domain decomposition problem. We can observe the typical behavior of these fast methods when minimizing a convex quadratic function.

Table 2 compares the conjugate gradient version of Cimmino's versus the conjugate gradient over the Schur complement system (CG-Uzawa and CG-AOP). We can observe that in some problems CG-Cimmino's can be better than the Schur complement conjugate-gradient method in a simple-processor machine. Quotient $time/m$ in table 2 shows the potential speedup of CG-Cimmino's for parallel machines. Figures 4,5, and 6 show the evolution of the norm of the gradient by iteration for a CUTEr problem, a Stokes problem and a domain decomposition problem.

7 Final remarks

Our preliminary tests, in sequential machines, show an outstanding behavior of Cimmino's method when it is accelerated by low-cost optimization techniques (e.g., conjugate gradient method and Barzilai-Borwein method). Taking into account that Cimmino's method and its accelerations are inherently parallel algorithms, we hope a much better performance when they are implemented in parallel machines. In order to improve further the performance of these techniques, preconditioning strategies should also be included.

Acknowledgments.
The author would like to thank Marcos Raydan and René Escalante for interesting discussions and constructive suggestions on this topic and for a careful

Problem			Cimmino's BB		Cimmino's CG		Cimmino's	
Name	m	n	iter	time	iter	time	iter	time
AUG2DCQP	1600	3280	339	211.51	117	163.93	*	*
CVXQP1S	50	100	12297	49.17	306	0.72	*	*
CVXQP2S	25	100	1138	2.67	58	0.08	*	*
CVXQP3S	75	100	9489	55.94	1025	5.80	*	*
DUALC1	215	223	10	0.38	4	0.19	4230	132.48
DUALC5	278	285	15	0.77	5	0.34	*	*
DUALC8	503	510	10	1.52	6	0.84	*	*
GOULDQP2S	349	659	39	1.86	23	1.25	*	*
KSIP	1001	1021	60	53.95	15	13.13	*	*
MOSARQP1	700	3200	8	11.28	5	9.83	*	*
MOSARQP2	600	1500	8	2.42	5	1.95	*	*
PRIMAL1	85	410	57	1.45	20	0.50	*	*
PRIMAL2	96	745	45	1.77	20	0.77	*	*
PRIMAL3	111	856	51	4.98	26	2.28	*	*
PRIMAL4	75	1564	43	2.88	21	1.38	*	*
PRIMALC1	9	239	6	0.05	4	0.02	*	*
PRIMALC2	7	238	6	0.06	3	0.03	63	0.13
QGROW15	300	645	65	3.63	41	2.27	*	*
QGROW22	440	946	71	6.92	45	4.67	*	*
QSCFXM3	990	1800	3229	837.88	294	83.30	*	*
STOKES1	256	578	57	2.31	30	1.63	*	*
STOKES2	704	1538	74	18.61	68	17.39	*	*
STOKES3	256	578	51	1.92	33	1.28	*	*
STOKES4	256	578	75	2.73	29	1.14	*	*
DD1	80	1600	163	3.48	64	1.45	501	10.36
DD2	85	2975	360	11.13	83	3.28	*	*
DD3	70	2850	225	6.72	33	2.30	675	20.14
DD4	140	4900	231	22.11	29	9.86	693	61.31

Table 1: Comparison between different accelerated Cimmino's methods

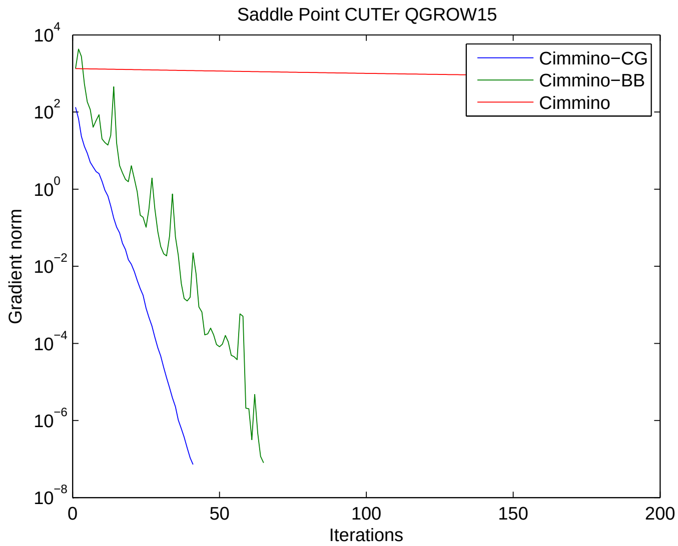

Figure 1: Cimmino's acceleration for a CUTEr problem

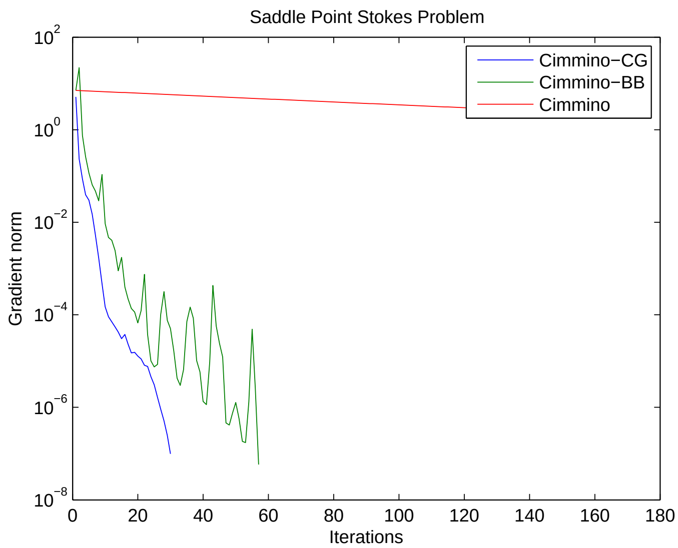

Figure 2: Cimmino's acceleration for a Stokes channel domain problem

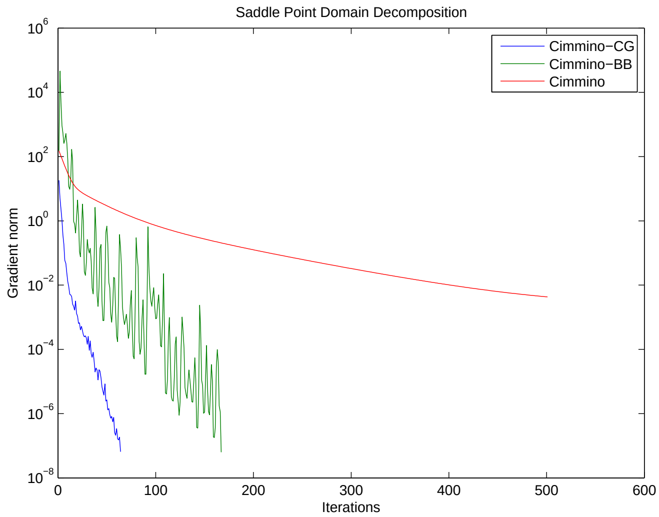

Figure 3: Cimmino's acceleration for a domain decomposition saddle point, $m = 80$ and $n = 1600$

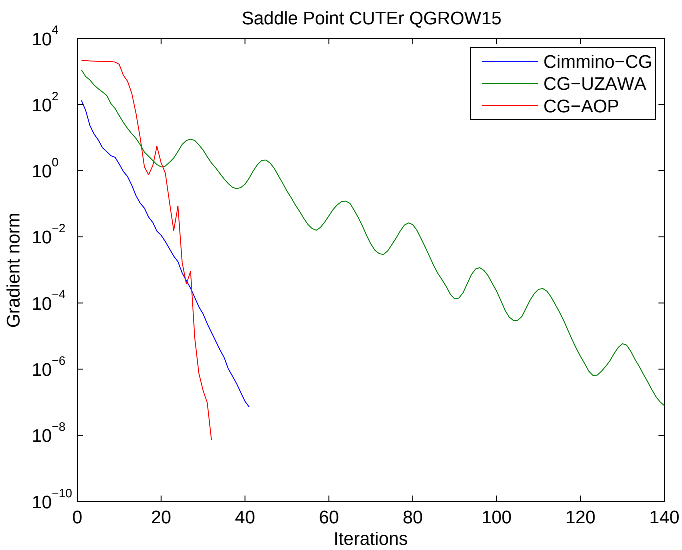

Figure 4: CG-Cimmino's vs. conjugate gradient over condensed system for a CUTEr problem

Problem			Cimmino's CG			CG-AOP		CG-UZAWA	
Name	m	n	iter	time	time/m	iter	time	iter	time
AUG2DCQP	1600	3280	117	163.93	0.1025	2	1.19	122	5.63
CVXQP1S	50	100	306	0.72	0.0144	44	0.16	152	0.19
CVXQP2S	25	100	58	0.08	0.0031	36	0.14	46	0.08
CVXQP3S	75	100	1025	5.80	0.0773	47	0.19	411	0.45
DUALC1	215	223	4	0.19	0.0009	21	0.83	267	1.36
DUALC5	278	285	5	0.34	0.0012	13	0.75	229	1.84
DUALC8	503	510	6	0.84	0.0017	11	2.14	503	16.13
GOULDQP2S	349	659	23	1.25	0.0036	51	4.59	29	2.23
KSIP	1001	1021	15	13.13	0.0013	10	11.77	6	0.06
MOSARQP1	700	3200	5	9.83	0.0140	65	308.67	*	*
MOSARQP2	600	1500	5	1.95	0.0033	102	79.09	4	2.56
PRIMAL1	85	410	20	0.50	0.0059	42	4.91	21	0.03
PRIMAL2	96	745	20	0.77	0.0080	31	6.89	21	0.09
PRIMAL3	111	856	26	2.28	0.0206	20	11.72	27	0.14
PRIMAL4	75	1564	21	1.38	0.0183	15	8.09	22	0.35
PRIMALC1	9	239	4	0.02	0.0017	10	0.17	3	0.02
PRIMALC2	7	238	3	0.03	0.0045	8	0.08	3	0.02
QGROW15	300	645	41	2.27	0.0076	32	6.44	140	8.69
QGROW22	440	946	45	4.67	0.0106	55	22.09	170	27.28
QSCFXM3	990	1800	294	83.30	0.0841	101	107.52	*	*
STOKES1	256	578	30	1.63	0.0063	10	0.44	32	0.11
STOKES2	704	1538	68	17.39	0.0270	13	2.11	93	0.58
STOKES3	256	578	33	1.28	0.0050	9	0.39	34	0.11
STOKES4	256	578	29	1.14	0.0045	9	0.38	29	0.09
DD1	80	1600	64	1.45	0.0181	21	0.34	69	0.42
DD2	85	2975	83	3.28	0.0386	23	0.75	112	1.31
DD3	70	2850	72	2.30	0.0329	21	0.56	225	1.13
DD4	140	4900	85	9.86	0.0704	23	1.27	89	1.78

Table 2: Comparison between Cimmino's conjugate gradient and the conjugate gradient method applied to the Schur complement system

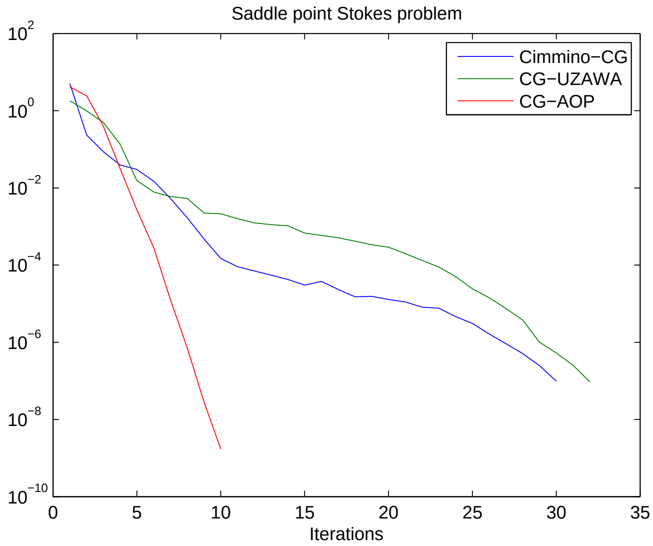

Figure 5: CG-Cimmino's vs. conjugate gradient over condensed system for a Stokes channel domain problem

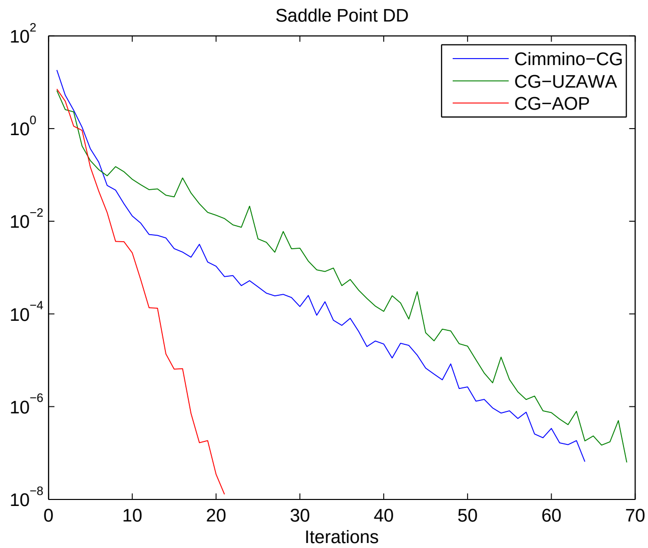

Figure 6: CG-Cimmino's vs. conjugate gradient for a domain decomposition saddle point, $m = 80$ and $n = 1600$

reading of this document. He also thanks Alessandra Fariñas and Roberto Morillo for programming assistance.

References

[1] M. Arioli and I.S. Duff and D. Ruiz and M. Sadkane, *Block Lanczos Techniques for Accelerating the Block Cimmino Method*, SIAM Journal on Scientific Computing, 16, pp. 1478–1511, 1995.

[2] J. Barzilai and J. M. Borwein, *Two point step size gradient methods*, IMA Journal of Numerical Analysis, 8, pp. 141—148, 1988.

[3] M. Benzi, *Gianfranco Cimmino's contributions to numerical mathematics*, Atti del Seminario di Analisi Matematica, Dipartimento di Matematica dell'Universita di Bologna. Volume Speciale: Ciclo di Conferenze in Memoria di Gianfranco Cimmino, Marzo-Aprile 2004, Tecnoprint, Bologna (2005), pp. 87–109.

[4] M. Benzi and G. Golub and J. Liesen, *Numerical Solution of Saddle Point Problems*. Acta Numerica (2005) 14: 1–137.

[5] Byrne, Charles, *A unified treatment of some iterative algorithms in signal processing and image reconstruction*. Inverse Problems 20 (2004) 103—120.

[6] Y. Censor, *Computational acceleration of projection algorithms for the linear best approximation problem*. Linear Algebra and its Applications 416 (2006) 111—123.

[7] Y. Censor, D. Gordon and R. Gordon, *Component averaging: An efficient iterative parrallel algorithm for large and sparse unstructured problems*, Parallel computing, 27(2001), 777–808.

[8] Y. Censor, *Computational acceleration of projection algorithms for the linear best approximation problem*, Linear Algebra and its Applications 416 (2006) 111–123.

[9] G. Cimmino, *Calcolo approssimato per le soluzioni dei sistemi di equazioni lineari*, La Ricerca Scientifica, II, 9 (1938), pp. 326–333.

[10] Y. H. Dai and L. Z. Liao, *R-linear convergence of the Barzilai and Borwein gradient method*, IMA Journal of Numerical Analysis 22, pp. 1–10, 2002.

[11] F. Deutsch, *Best Approximation in Inner Product Spaces*, Springer-Verlag, 2000.

[12] S. Dolar and A. Wathen, *Approximate factorization constraint preconditionners for saddle-point matrices*, SIAM J. Sci Comput., Vol 27, Number 5, pp. 1555–1572, 2006.

[13] H. Elman, D. Silvester and A. Wathen, *Finite Elements and Fast Iterative Solvers (with applications in incompressible fluid dynamics*, Oxford University Press, 2005.

[14] R. Escalante and M. Raydan, *Alternating Projection Methods*, Book in preparation.

[15] R. Fletcher, *Practical methods of Optimization*, John Wiley, 1987.

[16] R. Fletcher, *On the Barzilai-Borwein method, Department of Mathematics*, University of Dundee, NA/207, Dundee, Scotland, 2001.

[17] N.I.M. Gould, D. Orban and PH. L. Toint, *CUTEr (and SifDec) a Constrained and Unconstrained Testing Enviroment, Revisited*, Tech. Report TR/PA/01/04, CERFACS, Toulouse, France, 2001.

[18] Yi-Qing Hu and Yu-Hong Dai, *Inexact Barzilai-Borwein method for saddle point problems*, Numerical Linear Algebra with Applications, Volume 14, Issue 4, Pages 299 – 317.

[19] L.M. Hernández-Ramos, *Alternating oblique projections for coupled linear systems.* Numerical Algorithms (2005) 38: 285—303.

[20] L.M. Hernández-Ramos. *Un opérateur de projection pour les systèmes linéaires couplés.* Thèse du Diplôme d'Etudes Doctorales, Laboratoire Jacques-Louis Lions, l'Université "Pierre et Marie Curie" (Paris 6), Paris, 2000.

[21] B. Molina and M. Raydan, *Preconditioned Barzilai-Borwein method for the numerical solution of partial differential equations*, Numerical Algorithms 13, pp. 45-–60, 1996.

[22] M. Raydan, *On the Barzilai and Borwein choice of steplength for the gradient method*, IMA Journal of Numerical Analysis 13, pp. 321—326, 1993.

[23] M. Rozloznik, *Saddle point problems, iterative solution and preconditioning: a short overview.* Habilitation resume, Technical University of Liberec, Department of Modelling of Processes, Faculty of Mechatronics and Interdisciplinary Studies, Liberec, Czech Republic, April 2004.

[24] Qingzhi, Yang and Jinling, Zhao, *Some Remarks on the Convex Feasibility Problem and Best Approximation Problem*, Numerical Mathematics: Theory, Methods and Applications, Numer. Math. Theor. Meth. Appl., Vol. 1, No. 1, pp. 78-91 (2008).