SOLUTION OF STRUCTURED SYSTEMS OF LINEAR EQUATIONS USING ELEMENT-BY-ELEMENT PRECONDITIONERS[1]

M.J. Daydé[2], J.Y. L'Excellent[2], and N.I.M. Gould[3]

Abstract

We consider the solution of the n by n system of linear equations, $\boldsymbol{Ax} = \boldsymbol{b}$, where $\boldsymbol{A}$ is both sparse and structured so that it may be expressed as

$$\boldsymbol{A} = \sum_{i=1}^{p} \boldsymbol{A}_i.$$

Sparse structured linear systems arise in many applications. The elementary matrices $\boldsymbol{A}_i$ are of low rank, and are usually sparse so that their nonzero entries may be represented as a small, dense block.

We assume that $\boldsymbol{A}$ is a large and normally positive definite symmetric matrix. The solution technique considered is the conjugate gradient method using a range of Element-By-Element (EBE) preconditioners that were introduced by [9] and [12] and have been successfully applied in a number of applications in engineering and physics (see, for example, [10], and [5]).

1 Introduction

We consider the solution of the n by n system of linear equations, $\boldsymbol{Ax} = \boldsymbol{b}$, where $\boldsymbol{A}$ is both sparse and structured so that it may be expressed as

$$\boldsymbol{A} = \sum_{i=1}^{p} \boldsymbol{A}_i. \tag{1.1}$$

The elementary matrices $\boldsymbol{A}_i$ are of low rank, and are usually sparse so that their nonzero entries may be represented as a small, dense block. We are especially interested by the linear systems that come from the solution of large scale nonlinear optimization problems using the all-pervasive property of partial separability, first studied by [6]. The function f is said to be *partially separable* if $f(\boldsymbol{x}) = \sum_{i=1}^{p} f_i(\boldsymbol{x})$, where each *element* function f_i has a large invariant subspace. Crucially, the Hessian matrix of such a function is of the form of (1.1).

[1] Travels were funded, in part, by the ALLIANCE program from the British Council

[2] ENSEEIHT-IRIT, 2 rue Camichel, 31071 Toulouse CEDEX, France

[3] Central Computing Department, RAL, OXON OX11 0QX, England

We shall assume that $\boldsymbol{A}$ is a large and normally positive definite symmetric matrix. The solution technique considered is the conjugate gradient method using the Element-By-Element (EBE) preconditioners that were introduced by [9] and [12] and have been successfully applied in a number of applications in engineering and physics (see, for example, [10], and [5]). A detailed analysis of this technique is given by [14]. The following element-by-element preconditioners are considered : the Element matrix Factorization (EMF) of [7] based on a Cholesky factorization of each element; the Finite Element preconditioner (FEP) of [11]; the one-pass (EBE) and two-pass (EBE2) element-by-element preconditioners of [9] and [12], initially described and used in the context of finite element techniques for partial differential equations; the "Gauss Seidel" EBE preconditioner (GS EBE); the overlapping block (OVBC) element-by-element preconditioner; and the incomplete Cholesky (IC) preconditioner.

These preconditioners have some important features. They can be computed element-wise and most of them do not require assembly. Furthermore, they allow efficient parallelization of the matrix-vector and of the solution step involved in the preconditioned gradient iteration.

2 Finite Element preconditioners

In this section, we review the range of element-by-element preconditioners that we have investigated. Most of them were originally proposed for solving the linear systems that arise from finite-element solution of partial differential equations. We note that specific error analyses are possible for particular classes of model differential equations, but a more general analysis is most likely impossible. Thus, all of the preconditioners should be viewed as heuristics which aim to approximate $\boldsymbol{A}$ at low cost.

As we are assuming that the element matrix $\boldsymbol{A}_i$ has non-zeros in just n_i rows, we may write

$$\boldsymbol{A}_i = \boldsymbol{C}_i^T \boldsymbol{A}_i^e \boldsymbol{C}_i, \tag{2.1}$$

where the rows of the n_i by n *connectivity* matrix, $\boldsymbol{C}_i$, are simply the rows of the n by n identity matrix corresponding to the variables used in the element, and $\boldsymbol{A}_i^e$ is a dense n_i by n_i symmetric matrix.

2.1 Element Matrix Factorization

First, we assume that the elementary matrices $\boldsymbol{A}_i^e$ are positive definite. The basic idea (see [7]) for forming the preconditioner is to factorize each elementary matrix into

$$\boldsymbol{A}_i^e = \boldsymbol{L}_i^e \boldsymbol{L}_i^{eT}, \tag{2.2}$$

where $\boldsymbol{L}_i^e$ is a lower triangular matrix. The preconditioner is then

$$\boldsymbol{P}_{EMF} = \left(\sum_{1=1}^{p} \boldsymbol{L}_i \right) \left(\sum_{1=1}^{p} \boldsymbol{L}_i \right)^T, \tag{2.3}$$

where $\boldsymbol{L}_i = \boldsymbol{C}_i^T \boldsymbol{L}_i^e$. Clearly $\sum_{1=1}^p \boldsymbol{L}_i$ is also lower triangular and thus (2.3) is easy to invert. More generally, letting $\boldsymbol{L}_i = \bar{\boldsymbol{L}}_i + \boldsymbol{D}_i$, where $\bar{\boldsymbol{L}}_i$ is the strictly lower triangular part of $\boldsymbol{L}_i$ and $\boldsymbol{D}_i = \Delta(\boldsymbol{L}_i)$, we might choose

$$\boldsymbol{P}_{EMF}(\theta) = \left((1+\theta)^{-1} \sum_{1=1}^{p} \bar{\boldsymbol{L}}_i + (1+\theta) \sum_{1=1}^{p} \boldsymbol{D}_i \right) \left((1+\theta)^{-1} \sum_{1=1}^{p} \bar{\boldsymbol{L}}_i + (1+\theta) \sum_{1=1}^{p} \boldsymbol{D}_i \right)^T, \tag{2.4}$$

where θ is a non-negative parameter. In our experiments for simplicity we choose $\theta = 0$, but other choices have been suggested by [7] for finite-element applications.

2.2 Finite Element preconditioner

If $\boldsymbol{A}_i$ is positive semi-definite, [11] and [15] have suggested modifying the previous preconditioner so that

$$\boldsymbol{A}_i = (\boldsymbol{D}_i + \bar{\boldsymbol{L}}_i) \boldsymbol{D}_i^+ (\boldsymbol{D}_i + \bar{\boldsymbol{L}}_i^T), \tag{2.5}$$

where $\boldsymbol{D}_i^+$ is the pseudo-inverse of $\boldsymbol{D}_i$. Then, if $\sum_{i=1}^p \boldsymbol{D}_i$ has positive diagonal entries, the finite element preconditioner (FEP) is

$$\boldsymbol{P}_{FEP} = \left(\sum_{i=1}^{p} \boldsymbol{D}_i + \sum_{i=1}^{p} \bar{\boldsymbol{L}}_i \right) \left(\sum_{i=1}^{p} \boldsymbol{D}_i \right)^{-1} \left(\sum_{i=1}^{p} \boldsymbol{D}_i + \sum_{i=1}^{p} \bar{\boldsymbol{L}}_i^T \right). \tag{2.6}$$

2.3 The EBE preconditioner

Element-By-Element (EBE) preconditioners were introduced by [9] and [12] and have been successfully applied in a number of applications in engineering and physics (see, for example, [10], and [5]).

Assuming that $\boldsymbol{A}$ is positive definite, we may rewrite $\boldsymbol{A}$ as $\boldsymbol{A} = \sum_{i=1}^p \boldsymbol{M}_i + \sum_{i=1}^p (\boldsymbol{A}_i - \boldsymbol{M}_i) = \boldsymbol{M} + \sum_{i=1}^p (\boldsymbol{A}_i - \boldsymbol{M}_i)$, where $\boldsymbol{M}_i = diag(\boldsymbol{A}_i)$ and $\boldsymbol{M} = \sum_{i=1}^p \boldsymbol{M}_i$. Now, let $\boldsymbol{M} = \boldsymbol{L}_M \boldsymbol{L}_M^T$ be the Cholesky factorization of $\boldsymbol{M}$ — of course $\boldsymbol{L}_M$ is simply a diagonal matrix. Then,

$$\boldsymbol{A} = \boldsymbol{L}_M \left(\boldsymbol{I} + \sum_{i=1}^{p} \boldsymbol{L}_M^{-1} (\boldsymbol{A}_i - \boldsymbol{M}_i) \boldsymbol{L}_M^{-T} \right) \boldsymbol{L}_M^T = \boldsymbol{L}_M \left(\boldsymbol{I} + \sum_{i=1}^{p} \boldsymbol{E}_i \right) \boldsymbol{L}_M^T, \tag{2.7}$$

where $\boldsymbol{E}_i = \boldsymbol{L}_M^{-1} (\boldsymbol{A}_i - \boldsymbol{M}_i) \boldsymbol{L}_M^{-T}$. Using the approximation $\boldsymbol{I} + \sum_{i=1}^p \boldsymbol{E}_i \approx \prod_{i=1}^n (\boldsymbol{I} + \boldsymbol{E}_i)$, we obtain :

$$\boldsymbol{A} \approx \boldsymbol{L}_M \prod_{i=1}^{p} (\boldsymbol{I} + \boldsymbol{E}_i) \boldsymbol{L}_M^T. \tag{2.8}$$

The EBE preconditioner is given by :

$$P_{EBE} = \boldsymbol{L}_M \left(\prod_{i=1}^{p} \boldsymbol{L}_i \right) \left(\prod_{i=1}^{p} \boldsymbol{D}_i \right) \left(\prod_{i=p}^{1} \boldsymbol{L}_i^T \right) \boldsymbol{L}_M^T \tag{2.9}$$

where the $\boldsymbol{L}_i$ and $\boldsymbol{D}_i$ factors come from the $\boldsymbol{LDL}^T$ factorization of the matrices $\boldsymbol{I} + \boldsymbol{E}_i$.

Clearly, the efficiency of the EBE preconditioner depends on the the partitioning of the initial matrix and on the magnitude of the off-diagonal elements of the elementary matrices. As the decomposition of $\boldsymbol{A}$ is, in general, not unique, different decompositions may significantly affect the performance of the preconditioner.

In order to solve efficiently the system of equations $\boldsymbol{P}_{EBE}\boldsymbol{x} = \boldsymbol{y}$, we exploit the decomposition (2.9). We are free to order the elements in any way we choose and may thus encourage parallelism by consecutively ordering non-overlapping elements so that we can perform groups of forward and backsolve in parallel.

Remark : Combining (2.7) and the relationship

$$\boldsymbol{I} + \sum_{i=1}^{p} \boldsymbol{E}_i \approx \prod_{i=1}^{p} (\boldsymbol{I} + \tfrac{1}{2}\boldsymbol{E}_i) \prod_{i=p}^{1} (\boldsymbol{I} + \tfrac{1}{2}\boldsymbol{E}_i), \tag{2.10}$$

we obtain the two-pass element-by-element preconditioner (EBE2)

$$\boldsymbol{P}_{EBE2} = \boldsymbol{L}_M \left(\prod_{i=1}^{p} (\boldsymbol{I} + \tfrac{1}{2}\boldsymbol{E}_i) \right) \left(\prod_{i=p}^{1} (\boldsymbol{I} + \tfrac{1}{2}\boldsymbol{E}_i) \right) \boldsymbol{L}_M^T. \tag{2.11}$$

The main problem with the approximation (2.10) is that the terms $\frac{1}{4}{\boldsymbol{E}_i}^2$, which result when expanding the product (2.11), are non-zero even if there is little overlap between distinct element Hessians $\boldsymbol{E}_i$ and $\boldsymbol{E}_j$ $(i \neq j)$. Furthermore, as a solve using EBE2 is roughly twice as expensive as one with EBE, in practice EBE2 is less efficient than EBE.

2.4 The GS EBE preconditioner

The GS (Gauss-Seidel) EBE preconditioner is based on the same decomposition as the EBE preconditioner. But instead of using a Crout factorization, we instead use the decomposition

$$\boldsymbol{E}_i = \boldsymbol{L}_i + \boldsymbol{L}_i^T \tag{2.12}$$

where $\boldsymbol{L}_i$ is a strictly lower triangular matrix. The preconditioner is then

$$\boldsymbol{P}_{GS} = \boldsymbol{L}_M \prod_{i=1}^{p} (\boldsymbol{I} + \boldsymbol{L}_i) \prod_{i=p}^{1} (\boldsymbol{I} + \boldsymbol{L}_i^T) \boldsymbol{L}_M^T \tag{2.13}$$

The advantage of this preconditioner is obviously that it is very easy to construct. If $\boldsymbol{A}$ is pre-scaled to have unit diagonals, the preconditioner does not need to be explicitly stored. The principal drawback is that it is not exact, even when there is no overlap between element Hessians, because of the terms $\boldsymbol{L}_i\boldsymbol{L}_i^T$ which arise when approximating $\boldsymbol{I} + \sum_{i=1}^p \boldsymbol{E}_i$ by $\prod_{i=1}^p(\boldsymbol{I} + \boldsymbol{L}_i)\prod_{i=p}^1(\boldsymbol{I} + \boldsymbol{L}_i^T)$.

2.5 Overlapping block element-by-element preconditioner

Let $\boldsymbol{P}_i$, $i = 1, n$ be the assembled matrix whose sparsity is identical to $\boldsymbol{A}_i$, but whose entries are those of $\boldsymbol{A}$. $\boldsymbol{P}_i$ is positive definite and thus has a Cholesky decomposition $\boldsymbol{P}_i = \boldsymbol{L}_i\boldsymbol{L}_i^T$. The overlapping block preconditioner is

$$\boldsymbol{P}_{OVBC}^{-1} = \boldsymbol{D}^{\frac{1}{2}} \sum_{i=1}^{p} (\boldsymbol{L}_i{\boldsymbol{L}_i}^T)^{-1} \boldsymbol{D}^{\frac{1}{2}}$$

where $\boldsymbol{D}$ is a diagonal matrix defined by

$$\boldsymbol{D}_{ij} = \delta_{ij}.\boldsymbol{N}_i,$$

where $\boldsymbol{N}_i$ is the number of elements sharing the variable i. The construction of this preconditioner may be costly as the elements have to be assembled. However, See [8] indicate that the preconditioner is effective, and the solution step is parallel.

2.6 Incomplete Cholesky element-by-element preconditioner

We have designed an incomplete Cholesky preconditioner acting on elementary matrices. It is similar to a standard incomplete Cholesky without fill-in. The rank 1 updates are written directly into the elements and the preconditioner is assembled column-by-column. To allow fill-in in the factorization, merging some elements (c.f. Section 5) can be useful. A simple ordering – very close to minimum degree – is chosen : the variables belonging to a small number of elements are eliminated first. For each pivot, the pivot column is assembled, and the rank one update is matched into the elements containing at least one variable from the pivot column. As we might encounter small or negative pivots, the diagonals may need to be boosted during the factorization to prevent large growth. The advantage of this preconditioner is to keep the elemental structure of the matrix while computing the incomplete factorization. However our implementation is not optimal and it is often not competitive with the traditional incomplete Cholesky preconditioner where the matrix is assembled.

3 Experiments with Element-by-Element preconditioners

A number of the preconditioners described in Section 2 are known to work well in practice when applied to classes of problems arising from partial differential equations. In this section, we aim to investigate whether these preconditioners are effective in the more general context of systems which arise from partially separable optimization applications. Although our experiments are still preliminary, they do lead to interesting conclusions and are helpful in deciding future directions of research.

We report in Table 3.1 the results of applying some of the element-by-element preconditioners to optimization problems issued from the CUTE collection (see [1]) or to problems issued from the Harwell-Boeing collection(see [2]). Further experiments are reported in [4].

Problem name	Precondi-tioner	Time to calculate the preconditioner	Number of iterations	Time for convergence
NOBNDTOR	NONE	0	68	0.28
	DIAG	0	68	0.28
	EBE	0.03	35	0.58
	EBE2	0.02	33	1.05
	GSEBE	0.01	67	0.61
	OVBC	0.06	45	0.79
	EMF	0.03	36	0.23
	FEP	0.02	34	0.22
	IC	0.05	23	0.41
CEGB2802	NONE	0.0	121	7.39
	DIAG	0.01	35	2.19
	EBE	0.88	12	1.85
	EBE2	0.82	11	2.65
	GSEBE	0.14	17	2.54
	OVBC	2.21	16	2.40
	EMF	2.11	56	10.9
	FEP	1.86	32	6.30
	IC	8.43	9	4.54

Table 3.1: Comparison of different preconditioners on two test problems : NOBNDTOR, 562 variables, 480 elements and CEGB2802, 2694 variables, 108 elements, times reported for a SUN Sparc 10

3.1 Conclusions on the use of element-by-element preconditioners

The results of the previous sections are reinforced by [4] and indicate that element-by-element preconditioners are effective, in terms of the numbers of iterations required and the clustering of eigenvalues of the preconditioned Hessian, particularly if the overlap between blocks is small. EBE seems to be the best of our block preconditioners and it does not require any assembly of the matrix. EMF and FEP do not require an assembly of the matrix either but the resulting triangular incomplete factors need to be partially assembled, which can make each solve rather costly.

A disadvantage of EBE2 and GS EBE is that the terms $\frac{1}{4}\boldsymbol{E}_i^2$ and $\boldsymbol{L}_i\boldsymbol{L}_i^T$ may give rise to poor approximations even when there is little overlap between elements. If there is significant overlap, the efficiency of EBE2 and GS EBE is close to that observed for EBE. As GS EBE is the easiest preconditioner to construct, it may be beneficial to use GS EBE when we do not need much accuracy in the solution of our system, as, for example, is common in the early stages of optimization calculations. However in more general cases, we prefer EBE.

In our experiments, which are not all represented here, except for ill-conditioned problems, EBE is not significantly more efficient than diagonal preconditioning. We believe that this is for two reasons. The first is because of the structure of the elements. When there is lower overlap, EBE appears much more efficient than diagonal preconditioning. Amalgamating elements may reduce the number of iterations by decreasing the degree of overlap in the new partition. The second is that vectorization here is not as efficient as it could be. If we knew *a priori* that all blocs have the same size, it would be possible to vectorize the solve more efficiently, as was reported in previous experiments by, for example, [5].

4 Generalizing the EBE preconditioner

Let $\boldsymbol{A}$ satisfy (1.1). In Section 2.3, we considered the classical EBE preconditioner

$$\boldsymbol{P}_{EBE} = \boldsymbol{L}_M \left(\prod_{i=1}^{p} \boldsymbol{L}_i \right) \left(\prod_{i=1}^{p} \boldsymbol{D}_i \right) \left(\prod_{i=p}^{1} \boldsymbol{L}_i^T \right) \boldsymbol{L}_M^T, \tag{4.1}$$

where $\boldsymbol{M} \overset{\text{def}}{=} \Delta(\boldsymbol{A}) = \boldsymbol{L}_M \boldsymbol{L}_M^T$, $\boldsymbol{M}_i = \Delta(\boldsymbol{A}_i)$, $\boldsymbol{E}_i \overset{\text{def}}{=} \boldsymbol{L}_M^{-1}(\boldsymbol{A}_i - \boldsymbol{M}_i)\boldsymbol{L}_M^{-T}$ and the Winget decomposition, $\boldsymbol{W}_i$, had a factorization

$$\boldsymbol{W}_i \overset{\text{def}}{=} \boldsymbol{I} + \boldsymbol{E}_i = \boldsymbol{L}_i \boldsymbol{D}_i \boldsymbol{L}_i^T. \tag{4.2}$$

In this section, we consider a generalization of $\boldsymbol{M}$ with the intention of including more than simply the diagonal of $\boldsymbol{A}$.

Let $\boldsymbol{M}_i^e$ be a symmetric n_i by n_i matrix, for each $1 \le i \le p$,

such that the composite matrix,

$$\boldsymbol{M} = \sum_{i=1}^{p} \boldsymbol{M}_i, \tag{4.3}$$

is positive definite, and where

$$\boldsymbol{M}_i = \boldsymbol{C}_i^T \boldsymbol{M}_i^e \boldsymbol{C}_i. \tag{4.4}$$

Then we observe that the descriptions of the EBE preconditioner given in Section 2.3 *do not depend* on $\boldsymbol{M}$ being the diagonal of $\boldsymbol{A}$. We may thus think of using (4.1) with a non-diagonal $\boldsymbol{M}$. However, a restriction on the choice of $\boldsymbol{M}$ has to be made, since we want to have the same structure for the matrices $\boldsymbol{A}_i$ and $\boldsymbol{E}_i$. A general matrix $\boldsymbol{M}$ — indeed even a band matrix — may introduce fill-in, complicate the required data structures and increase storage and computational overheads. For simplicity in this section, we only consider matrices $\boldsymbol{M}$ for which no fill-in occurs.

To date, the generalized EBE preconditioner has only been tested on matrices with regular patterns : block diagonal matrices where the overlap appears only between two consecutive elements. $\boldsymbol{M}$ is defined to be the overlapping parts of the elements. Table 4.2 gives a comparison between EBE and GEN EBE for a problem for which the logarithms of the eigenvalues in each element are randomly chosen between -1 and 1. There are 50 elements, each of dimension 10. We allow the overlap to vary and report on the effect of this in the table. For these matrices, we observe that GEN EBE appears to be more effective as the overlap increases, but unfortunately is still less efficient than EBE.

		Preconditioner			
Overlap	Order	None	Diagonal	EBE	GEN EBE
0	500	102/1.7	116/2.0	1/0.20	1/0.20
1	451	111/1.8	100/1.6	19/1.3	19/2.6
2	402	102/1.7	86/1.4	20/1.3	18/2.5
3	353	96/1.5	70/1.2	20/1.2	17/2.3
4	304	84/1.3	59/0.97	17/0.99	15/2.1
5	255	59/0.96	50/0.83	18/0.99	12/1.7

Table 4.2: Comparison of EBE and GEN EBE on a well conditioned problem (number of iterations / time to converge)

5 Importance of element amalgamation

Clearly, the efficiency EBE preconditioners depends on the the partitioning of the initial matrix and on the magnitude of the off-diagonal elements of the elementary matrices. As the decomposition of $\boldsymbol{A}$ is, in general, not unique, different decompositions may significantly affect the performance of the preconditioner. In order to improve both the computational performance and the numerical convergence of the method, we amalgamate two elements into a group if their overlap is large. The main difficulty is to define a suitable heuristic to control this amalgamation process. Two strategies were considered : **amal1** is an amalgamation heuristic that aims at minimizing the time spent in matrix-vector products; **amal2** is another amalgamation heuristic well suited for the EBE preconditioner since it tries to minimize the time spent both in the matrix-vector products and in the forward-backward solution. In Table 5.3 we report the number of iterations, the time and the speed-up per iteration, and the solution time (in seconds), of a diagonal preconditioner and the EBE preconditioner on the Alliant FX/80 using our two amalgamation heuristics. The matrix **NET3** is of order 512 and has 538 elements, while the second one, **CBRATU3D**, is of order 4394 and has 4394 elements. Also these results demonstrate the very good potential of element-by-element preconditioners for parallelization by considering the speed-ups achieved.

Problem	Preconditioner		Iterations		Time and speed-up per iteration		Time for solution	
			seq	par	seq	8 procs	seq	8 procs
NET3	Diagonal	initial	1561		0.04	3.72	57.3	15.4
		amal1	1595		0.01	2.77	23.8	2.9
	EBE	initial	628	696	0.14	4.90	88.1	19.9
		amal2	140	137	0.04	3.35	5.18	1.5
CBRATU3D	Diagonal	initial	53		0.59	4.50	31.9	7.1
		amal1	53		0.50	4.57	27.3	6.0
	EBE	initial	24	25	1.95	5.74	48.8	8.8
		amal2	19	19	1.28	4.94	25.7	5.2

Table 5.3: Performance summary of the diagonal and the EBE preconditioners on the Alliant FX/80.

6 Conclusion

We have shown that element-by-element preconditioners may be extremely effective for sparse structured systems of linear equations which arise in partial differential equations and partially separable nonlinear optimization applications. Furthermore, they seem to offer great possibilities of vectorization/parallelization on multiprocessor architectures. We believe that further study, on real problems with less regular matrix structures, will inevitably lead to a better understanding of the classes of problems for which each of the preconditioners we have considered is particularly appropriate.

We have demonstrated that element preprocessing is crucially important. It is clear that this preprocessing should be applied to amalgamate elements prior to the formation of the preconditioner, and so that a suitable colouring is found for subsequent parallel execution. This is a difficult task as many criteria need to be taken into account. This preprocessing may well be quite costly, but we expect there to be longer-term payoffs. At a certain level of amalgamation, it may well be important to exploit the sparsity within the amalgamated elements.

These element-by-element preconditioners will shortly be publically available under anonymous ftp within a package called *PAREBE*. We aim to implement these preconditioning techniques in the LANCELOT package for large scale optimization problems, profiting of the preprocessing to optimize other parts of the algorithm such as the matrix-vector products and linear solvers.

References

[1] I. Bongartz and A. R. Conn and N. I. M. Gould and Ph. L. Toint. CUTE: Constrained and Unconstrained Testing Environment Technical report TR/PA/93/10, CERFACS, Toulouse, France, 1993.

[2] I. S. Duff, R. G. Grimes, and J. G. Lewis. Users' guide for the harwell-boeing sparse matrix collection. Technical Report TR/PA/92/86, CERFACS, Toulouse, France, 1992.

[3] M. J. Daydé, J. Y. L'Excellent, and N. I. M. Gould. PAREBE: Parallel Element-By-Element preconditioners for the conjugate gradient algorithm. Technical report, ENSEEIHT, Toulouse, France, to appear.

[4] M. J. Daydé, J. Y. L'Excellent, and N. I. M. Gould. On the use of element-by-element preconditioners to solve large scale partially separable optimization problems. Technical report RT/APO/94/4, ENSEEIHT, Toulouse, France, 1994.

[5] J. Erhel, A. Traynard, and M. Vidrascu. An element-by-element preconditioned conjugate gradient method implemented on a vector computer. *Parallel Computing*, 17:1051–1065, 1991.

[6] A. Griewank and Ph. L. Toint. On the unconstrained optimization of partially separable functions. In M. J. D. Powell, editor, *Nonlinear Optimization 1981*, pages 301–312, London and New York, 1982. Academic Press.

[7] I. Gustafsson and G. Lindskog. A preconditioning technique based on element matrix factorization. *Computational Methods in Applied Mechanics*, 55:201–220, 1986.

[8] L. J. Hayes and P. Devloo An element-by-element block iterative method for large nonlinear problems In W. K. Liu and T. Belytschko and K. C. Park editors, *International Conference on Innovative Methods for Nonlinear Problems*, pages 51-62, Swansea, 1984. Pineridge Press Ltd.

[Hughes, 1987] T. J. R. Hughes. *The Finite-Element Method: Linear Static and Dynamic Finite-Element Analysis.* Prentice Hall, Englewood Cliffs, New Jersey, USA, 1987.

[9] T. J. R Hughes, I. Levit, and J. Windget. An element-by-element solution algorithm for problems of structural and solid mechanics. *Computational Methods in Applied Mechanics and Engineering*, 36:241–254, 1983.

[10] T. J. R. Hughes, R. M. Ferencz, and J. O. Hallquits. Large-scale vectorized implicit calculations in solid mechanics on a CRAY X-MP/48 utilizing EBE preconditioned conjugate gradients. *Computational Methods in Applied Mechanics and Engineering*, 61:215–248, 1987.

[11] E. F. Kaasschieter. A general finite element preconditioning for the conjugate gradient method. *BIT*, 1990.

[12] M. Ortiz, P. M. Pinsky, and R. L. Taylor. Unconditionally stable element-by-element algorithms for dynamic problems. *Computational Methods in Applied Mechanics and Engineering*, 36:223–239, 1983.

[13] A. Ramage and A. J. Wathen On preconditioning for finite element equations on irregular grids *SIAM J. Matrix Ananl. Appl.*, 15:909-921, 1994.

[14] A. J. Wathen. An analysis of some element-by-element techniques. *Computational Methods in Applied Mechanics and Engineering*, 74:271–287, 1989.

[15] A. J. Wathen. Singular element preconditioning for the finite element method In R. Beauwens and P. de. Groen, editors, *Iterative methods in linear algebra*, pages 531–540, Elsevier/North Holland, Amsterdam, 1992.