Approximate Inverse Based Multigrid Solution of Large Sparse Linear Systems

A thesis submitted to

Lakehead University

in partial fulfillment of the requirements

for the degree of

Master of Science

by

Rabindra Nath Banerjee Fdez.-Bordas

1988

ISBN 0-315-44789-3

Acknowledgements

I want to thank my supervisor, Professor M.W. Benson, for his advice, encouragement and generous support during the preparation of this thesis. I thank him for his friendship and patient dedication through many stimulating discussions.

I am also indebted to Professor D.L. Black and Ms. J. Rives for their generous help in the translation of a Russian reference.

I extend my deepest appreciation to the Sarbadhikari and Chahal families; this thesis would not have been completed without their love and hospitality.

Finally, I dedicate this work to my parents and Susana, for their faithful support and encouragement during these years of uncertainty.

Abstract

In this thesis we study the approximate inverse based multigrid algorithm FAPIN for the solution of large sparse linear systems of equations.

This algorithm, which is closely related to the well known multigrid V-cycle, has proven successful in the numerical solution of several second order boundary value problems. Here we are mainly concerned with its application to fourth order problems. In particular, we demonstrate good multigrid performance with discrete problems arising from the beam equation and the biharmonic (plate) equation. The work presented also represents new experience with FAPIN using cubic B-spline, bicubic B-spline and piecewise bicubic Hermite basis functions. We recast a convergence proof in matrix notation for the nonsingular case.

Central to our development are the concepts of an approximate inverse and an approximate pseudo-inverse of a matrix. In particular, we use least squares approximate inverses (and related approximate pseudo-inverses) found by solving a Frobenius matrix norm minimization problem. These approximate inverses are used in the multigrid smoothers of our FAPIN algorithms.

Table of Contents

Introduction 1

Chapter I : Theoretical Background 4

1. Linear Stationary Iterative Methods 5

2. Approximate Inverses and Approximate Pseudo-inverses 8

3. LSQ-approximate Inverses and LSQ-approximate Pseudo-inverses 13

4. The Fast Approximate Inverse (FAPIN) 18

Chapter II : Numerical Experiments 32

1. Model problems 33

2. Experimental Measures 39

3. One-dimensional Model Problems 46

4. Two-dimensional Second Order Model Problems 50

5. Two-dimensional Fourth Order Model Problems 58

6. Conclusions and Suggestions for Further Study 63

Bibliography 65

Appendix I : Calculation of Element Matrices 68

1. Cubic Splines 69

2. Piecewise Cubic Hermites 72

3. Rescaling of the Piecewise Cubic Hermites : Scaling 1 75

4. Rescaling of the Piecewise Cubic Hermites : Scaling 2 76

Appendix II : Imposing Essential Boundary Conditions 79

1. A Second Order Problem 80

2. A Fourth Order Problem 83

3. Periodic Boundary Conditions 85

Appendix III : Interpolation and Collection Operators 88

1. Periodic Cubic Splines 89

2. Non-Periodic Cubic Splines 91

3. Piecewise Cubic Hermites 100

4. Piecewise Linear Elements 105

5. Kronecker Product Formulation of Interpolation Operators 107

Appendix IV : The High Level Environment HL 110

1. Description 111

2. Some Sparse Operators 112

3. Sample C-Source Code for an HL Operator 114

4. Sample HL Code 117

Appendix V : Proof of Theorem 3.1 of Chapter I 119

Introduction

The approximate inverse based multigrid algorithm FAPIN ('Fast Approximate Inverse' Frederickson [1975] for nonsingular problems, 'Fast Approximate Pseudo-Inverse' Frederickson & Benson [1986] for singular problems), has been used for the solution of linear systems arising from the finite element discretization of two-point second order eigenvalue problems (Chew [1977]), triangular finite element discretizations of second order eigenvalue and boundary value problems (abbreviated BVP's) (Liong [1977]) and and finite element discretizations of two-dimensional second order BVP's (Baumgardner & Frederickson [1985], Frederickson & Benson [1986] and Benson & Frederickson [1987]).

In this thesis we apply FAPIN to the solution of certain large sparse linear systems constructed from the finite element discretization of some one and two dimensional second and fourth order BVP's with both periodic and non-periodic boundary conditions. To this end, we consider finite element bases of piecewise linear, piecewise cubic Hermite and cubic spline functions. We also consider some two dimensional counterparts of these bases.

The logical units in this thesis are chapters and appendices, both of which are further subdivided into sections.

In Chapter I we introduce most of our notation and establish the necessary theoretical background. In Section 1, we consider linear stationary methods and relate them, in Section 2, to the concepts of an approximate inverse and an approximate pseudo-inverse of a matrix. Section 3 deals with the Least Squares (LSQ) approximate inverse (Benson [1973]) and the LSQ-approximate pseudo-inverse (Frederickson & Benson [1986])of a given matrix. These are used, together with other approximate inverses and approximate pseudo-inverses derived from them, to build the required multigrid smoothers. Though not directly applicable to our work, we also state in this section a property of Kronecker products of LSQ-approximate inverses. This result finds application when solving spline approximation problems in a rectangular domain by means of an LSQ-approximate inverse based linear stationary method. Finally, in Section 4 we describe the algorithm FAPIN and recast in matrix notation a convergence proof for the nonsingular case.

Chapter II is devoted to our numerical experiments. In Sections 1 and 2 we define our model problems and the experimental measures used for their analysis. Sections 3 to 5 present our experimental results, while in Section 6 we give our conclusions and some suggestions for further study.

The theoretical and practical considerations necessary to set up our experiments are contained in the appendices. The first of them contains the calculation of the element matrices required for the construction of our model problems. Appendix II describes how some essential boundary conditions may be imposed on a B-spline discretization of two-point BVP's. In Appendix III we calculate the intergrid transfer operators required for the implementation of the algorithm FAPIN for the different finite element bases considered. We also show that, when the two-dimensional bases have a Kronecker product structure, only the one-dimensional operators need be calculated and the two-dimensional operators can be generated from Kronecker products of these one-dimensional operators. Appendix IV briefly describes the high level environment HL (see Benson [1987]) used to implement our numerical experiments. Finally, in Appendix V we give the proof of the Kronecker product property of the LSQ-approximate inverse mentioned above.

References to different material in this thesis will be done according to the following conventions : We use the standard symbol § to denote section. Further, theorem (definition, equation) k in section j of Chapter i (where i is a Roman numeral) will be referred as T$j.k$ (D$j.k$, $(j.k)$) within Chapter i and as Ch.i.T$j.k$ (Ch.i.D$j.k$, Ch.i.$(j.k)$) in other chapters or appendices. Similarly, material in the appendices will be referenced following the same conventions with the prefix ‘Ch.’ changed to ‘Ap.’ .

CHAPTER I

Theoretical Background

In this chapter we introduce the necessary theoretical background for the definition and analysis of the multigrid algorithm FAPIN.

In Section 1 we recall some results concerning linear stationary iterative methods for the solution of nonsingular systems of equations. These ideas are related, in Section 2, to the concepts of an approximate inverse and an approximate pseudo-inverse of a matrix. The iterative solution of singular systems of equations is also considered within this framework. Section 3 describes the LSQ-approximate inverse and the LSQ-approximate pseudo-inverse of a matrix, together with some of their properties. Finally, in Section 4 we define the algorithm FAPIN. We study its properties and recast a convergence proof in terms of matrices and energy norms.

1. Linear Stationary Iterative Methods

In this section we present some theoretical results concerning linear stationary iterative methods for solving linear systems of equations. We also set up some of the notation we will use in this thesis and create the framework within which we will analyze the concepts of an approximate inverse and an approximate pseudo-inverse (or 'generalized inverse' as in Ben-Israel & Greville [1974]) of a given matrix. These concepts are of central importance to this thesis. We begin with some notation.

We let $\mathbf{R}$, $\mathbf{Z}$ and $\mathbf{Z}^+$, denote the set of real, integer and non-negative integer numbers, respectively. We let $\mathbf{M}_{n\times m}(\mathbf{R})$ denote the set of n by m real matrices, understanding $n = m$ when only one subscript is given. Furthermore we denote by $\rho(A)$ the spectral radius of the matrix A. $N(A)$, and $R(A)$ will denote the null space and the range space of A, respectively.

Next we state some well known definitions and theorems, referring the reader to the indicated references for the proofs.

In what follows, we restrict our attention to the real case, though the definitions given apply also to problems based on the complex field. Suppose we are given a linear system of equations

$$Au = f \quad , A\in\mathbf{M}_n(\mathbf{R}) \quad , u, f\in\mathbf{R}^n \quad . \tag{1.1}$$

All iterative methods employed in this thesis for the solution of (1.1) are of the form :

$$u^{(i+1)} = Gu^{(i)} + k, \; i = 0, 1, \cdots \quad , \tag{1.2}$$

for some $G \in \mathbf{M}_n$ and some $k \in \mathbf{R}^n$, and $u^{(0)} \in \mathbf{R}^n$ an arbitrary initial approximation. In the terminology of Young [1971] these are called linear stationary iterative methods of degree one. Since these are the only iterative methods considered here we will call them, for simplicity, just linear stationary methods.

If the sequence defined by (1.2) converges to some vector u^*, then, in the limit $u^* = Gu^* + k$, or equivalently : $(I - G)u^* = k$. In other words, u^* is a solution to the linear system $(I - G)u = k$. Hence, if we want the method (1.2) to converge to a solution of (1.1), some relationship between the solutions of the system

$$(I - G)u = k \tag{1.3}$$

and (1.1) must exist. This observation motivates the following definition :

Definition 1.1 (Young [1971, p.64] : *The method (1.2) is said to be*

(i) *Consistent with (1.1) if every solution of (1.1) is a solution of (1.3).*

(ii) *Reciprocally consistent with (1.1) if every solution of (1.3) is a solution of (1.1).*

(iii) *Completely consistent with (1.1) if* (i) *and* (ii) *hold simultaneously*

When condition D1.1.(i) holds then, if applying (1.2), we arrive to a solution $\overline{u}$ of (1.1), then all further iterations remain the same. Moreover, definition D1.1.(ii) requires that, if the sequence defined by (1.2) converges to a limit u^*, then this limit is a solution of (1.1).

Next we examine under what conditions D1.1 are satisfied. We give these in the form of a theorem referring the reader to Young [1971, p.68] for the proof.

Theorem 1.1 : *If the system (1.1) has a solution then the method (1.2) is :*

(i) *Consistent with (1.1) if and only if there exists* $Z \in \mathbf{M}_n(\mathbf{R})$ *such that :*

$$G = I - ZA, \text{ and } k = Zf \tag{1.4}$$

(ii) *Reciprocally consistent with (1.1) if and only if there exists* $Q \in \mathbf{M}_n(\mathbf{R})$ *such that :*

$$A = Q(I - G), \text{ and } f = Qk$$

(iii) *Completely consistent with (1.1) if and only if condition (i) holds with a nonsingular Z.*

In the next section of this chapter we will see that equation (1.4) in T1.1.(i) is closely related to the concept of an approximate inverse (see Froberg [1969, p.94] and Benson & Frederickson [1982, p.128]) of the matrix A, since if we let $Z = A^{-1}$, then we would have : $G = 0$ and $k = A^{-1}f$. Therefore it is reasonable to expect that the closer (in a sense to be made precise later) Z is to A^{-1}, when this is defined, the better the iterative method (1.2), based on G, will perform.

Now we turn our attention to the convergence properties of the method (1.2). We first introduce some notation and then give a convergence theorem. Both are adapted from Young [1971, p.77].

Definition 1.2 : *We call the method (1.2) convergent if the sequence defined by (1.2) converges for all $u^{(0)}$ and it does so to a limit that is independent of $u^{(0)}$.*

We can now state the fundamental convergence result for linear stationary methods (see Young [1971, p.77]) :

Theorem 1.2 : *The iterative method (1.2) is Convergent if and only if $\rho(G)<1$.*

The previous results allow us to summarize the conditions under which (1.2) can be used to iteratively solve (1.1) when A is nonsingular :

Conditions 1.1 : *For A nonsingular :*

(i) $\rho(G)<1$.

(ii) $G = I - ZA \quad ,k = Zf$

Observe that from condition 1.1.(i) we can immediately conclude (see Atkinson [1978, Theorem 7.10]) that $I - G = ZA$ is nonsingular and therefore Z must be nonsingular.

Conditions 1.1 imply complete consistency of (1.2) with (1.1) and convergence of (1.2). Note also that, under these conditions, the method (1.2) satisfies :

$$e^{(n)} = G^n e^{(0)} \quad , \tag{1.5}$$

where the error $e^{(i)}$ in the i-th iteration is defined by $e^{(i)} = u^{(i)} - \overline{u}$, where $\{u^{(i)}\}$, $i = 0, 1, \cdots$ is the sequence defined by (1.2) for some initial vector $u^{(0)}$, and $\overline{u}$ is the solution of (1.1). Also, when conditions 1.1 hold, we have $e^{(n)} \longrightarrow 0$ as $n \longrightarrow \infty$.

So far we have restricted ourselves to the case A nonsingular. In the singular case, we need additional conditions on the matrix G to guarantee that the method (1.2) will yield a solution of (1.1) (when this exists). These will be provided in the next section, where we concentrate on the concepts of an approximate inverse and an approximate pseudo-inverse of a given matrix A. We build linear stationary methods by letting the matrix Z in T1.1 be an approximate inverse or an approximate pseudo-inverse of A when A is nonsingular or singular, respectively.

2. Approximate Inverses and Approximate Pseudo-inverses

In this section we introduce the concepts of an approximate inverse of a given nonsingular matrix and that of an approximate pseudo-inverse of a given singular matrix. Their use for the construction of linear stationary methods, which is central to this thesis, is also examined. Though these concepts can be defined in more general settings (see, for example, Ben-Israel & Greville [1974, pp. 306-357] and Frederickson & Benson [1986], where a Hilbert Space framework is adopted), the algebraic point of view will suffice for our purposes.

Assume we are given a linear system of equations

$$Au = f \quad , A \in \mathbf{M}_n(\mathbf{R}),\ u,f \in \mathbf{R}^n \tag{2.1}$$

to be solved numerically by means of a linear stationary method. Assume further that A is nonsingular. Recalling the results of §1, we would like to find a matrix Z such that conditions 1.1 are satisfied and thus have a completely consistent convergent method. If we can build a Z with the property that $\| I - ZA \|_\alpha = \epsilon < 1$, for some compatible matrix α-norm ($\| \,.\, \|_\alpha$ is compatible *iff* $\| Ax \|_\alpha \leq \| A \|_\alpha \| x \|_\alpha$, $\forall$ $A \in \mathbf{M}_n(\mathbf{R})$, $x \in \mathbf{R}^n$; see Atkinson [1978, p.415]), this would be accomplished, since condition 1.1.i would be automatically satisfied ($\rho(A) \leq \| A \|_\alpha$ for any compatible matrix α-norm). These observations motivate the following definition :

Definition 2.1 (Benson & Frederickson [1982, p.128]) :

Given a nonsingular matrix $A \in \mathbf{M}_n(\mathbf{R})$, *we say that* $Z \in \mathbf{M}_n(\mathbf{R})$ *is an* ϵ-*approximate* *inverse of A if for $\epsilon < 1$*

$$\| I - ZA \|_{\alpha} \leq \epsilon \ ,$$

where $\| \ . \|_{\alpha}$ is any compatible matrix norm.

Observe that by taking norms in equation (1.5), we have $\| e^{(n)} \|_{\alpha} \leq \epsilon^{n} \| e^{0} \|_{\alpha}$. Thus, the closer Z is to A^{-1} in the α-norm, the better we might expect the iterative method (1.2) to perform asymptotically (see Varga [1970]). We point out that the common iterative methods (Jacobi, Gauss-Seidel, SOR) can be studied within the framework of approximate inverses (see Benson [1973] for an extensive account), which is also closely related to that of Regular Splittings of a matrix (see §3 of both Varga [1970] and Young [1971]).

For the remainder of this section we will view a matrix in $\mathbf{M}_n(\mathbf{R})$ as a linear map of $\mathbf{R}^n$ into itself. Further, we consider that an inner product has been defined in $\mathbf{R}^n$, since we will need the notion of orthogonality to be defined. We also let $\| \ . \|$ denote the corresponding norm derived from the inner product defined in $\mathbf{R}^n$.

When A is singular we must work with a pseudo-inverse instead of an inverse. Among all the possible definitions of a pseudo-inverse (see Ben-Israel & Greville [1974]) we find the following convenient (we note that this is not the Moore-Penrose pseudo-inverse) :

Definition 2.2 : *Given a singular $A \in \mathbf{M}_n(\mathbf{R})$, we call Z a pseudo-inverse of A if :*

(i) $Z = ZAZ$

(ii) $N(A)^{\perp} = R(Z), R(A)^{\perp} = N(Z)$,

where $N^{\perp}(A)$ denotes the orthogonal complement of $N(A)$ with respect to the defined inner product in $\mathbf{R}^n$.

The above definition corresponds, in the notation of Ben-Israel & Greville [1974, p.61] to that of a {2}-inverse with prescribed range and null space. We also note that, if in addition to (i) and (ii) above, we require Z to satisfy the condition $A = AZA$, then Z

would be the Moore-Penrose pseudo-inverse (A^+) of A (see exercise 32 in Ben-Israel & Greville [1974, p.62]).

An analogous definition to D2.1, for the singular case, is adapted from Frederickson & Benson [1986] :

Definition 2.3 : *The matrix* $Z \in \mathbf{M}_n(\mathbf{R})$ *is an* ϵ*-approximate pseudo-inverse of* A *if for* $\epsilon < 1$:

(i) $\| (Z - ZAZ)v \| \leq \epsilon \| Zv \|, \forall v \in \mathbf{R}^n$ and

(ii) $N(A)^{\perp} = R(Z), R(A)^{\perp} = N(Z)$.

Observe that D2.3 reduces to D2.1 when A and Z are nonsingular. Moreover, if we rewrite D2.3.(i) as $\| (I - ZA)(Zv) \| \leq \epsilon \| Zv \|, \forall v \in \mathbf{R}^n$ we see that the orthogonality conditions D2.3.(ii) essentially require Z to be an approximate inverse of A, the latter considered as a linear mapping from the orthogonal complement of $N(A)$ onto $R(A)$. This null space behavior of our approximate pseudo-inverses will allow us to build linear stationary methods based on them. The precise convergence properties of these methods are given in the following theorem from Frederickson & Benson [1986]. We give the proof since it illustrates how the orthogonality conditions in D2.3 are essential for convergence.

Theorem 2.1 : *If* Z *is an* ϵ*-approximate pseudo-inverse of* A*, then the method (1.2), based on* Z*, converges at the geometric rate* ϵ*, for any initial* $u^{(0)} \in \mathbf{R}^n$ *to a vector* $\overline{u}$*, such that* $\| f - A\overline{u} \|$ *is minimal. Moreover, if* $u^{(0)} = 0$*, then* $\overline{u}$ *is the Moore-Penrose pseudo-inverse solution :* $\overline{u} = u^+ = A^+ f$.

Proof :

We first write (1.2) in the form of a residual iteration, that is :

$$\begin{cases} r^{(n)} & = f - Au^{(n)} \\ u^{(n+1)} & = u^{(n)} + Zr^{(n)} \end{cases} \tag{2.2}$$

where $r^{(n)}$ is the residual in the n-th iteration. Two consecutive iterations of (2.2) yield : $u^{(n+1)} - u^{(n)} = (Z^{'} - ZAZ)\,(f\ - Au^{(n-1)})$. Since Z is an ϵ-approximate pseudo-inverse, we obtain : $\| \ u^{(n+1)} - u^{(n)} \| \leq \epsilon \| \ Z(f\ - Au^{(n-1)}) \|$. From (2.2) we have :

$$\| \ Z(f\ - Au^{(n-1)}) \| = \| \ Zr^{(n-1)} \| = \| \ u^{(n)} - u^{(n-1)} \| \quad . \tag{2.3}$$

Thus :

$$\| \ u^{(n+1)} - u^{(n)} \| \leq \epsilon^n \ \| \ u^1 - u^0 \| \ .$$

Therefore $u^{(n)}$ converges geometrically to a certain element $\overline{u} \in \mathbf{R}^n$. If we take the limit $n \rightarrow \infty$ in (2.3) we have $\| \ Zr \ \| = \| \ \overline{u} \ - \overline{u} \ \| = 0$, where r is the residual for $\overline{u}$. Therefore, $r \in N(Z)$. Since, by hypothesis, $N(Z) = R(A)^{\perp}$, we finally have $r \perp R(A)$, and therefore r is minimal.

If $u^{(0)} = 0$ then, from (2.2) and the orthogonality properties of Z, we see that $u^{(n)} \in R(Z) = N^{\perp}(A)$, $\forall n \in \mathbf{Z}^{+}$. Thus the limit $\overline{u} \in R(Z)$. $\overline{u}$ is also of minimum norm, for if u' is another residual minimizer, we must have $A\overline{u} = Au'$, and then $(\overline{u} \ - u') \in \mathbf{N}(A)$. But then $\overline{u} \perp (\overline{u} \ - u')$ and therefore $\| \ \overline{u} \ \| < \| \ u' \ \|$. Thus $\overline{u} = u^{+} = A^{+}f \quad \square$.

Several observations follow from the previous theorem. First we note that the method (2.2) (or equivalently (1.2)) converges to the vector $\overline{u}$ closest to $u^{(0)}$ in the norm $\| \ . \|$ such that $\| \ f\ - A\overline{u} \ \|$ is minimal. Thus for a $\| \ . \|_2$ minimization, when $u^{(0)} = 0$, $\overline{u}$ is the usual least squares solution of minimum norm (see Stewart [1973] and Ben-Israel & Greville [1974]). Second, if (2.1) is solvable then $f \in R(A)$. Thus the minimum of $\| \ f\ - Au \ \|$ is 0 and $\overline{u}$ is an exact solution of (2.1). Hence, we have a convergent consistent method in the sense of Young [1971, p.68]. It is also worth noting that two consecutive iterations of (1.2), based on an approximate pseudo-inverse Z, would be equivalent to a single iteration of the same method but based on the matrix $Z\ '$ defined by : $Z\ ' = 2Z\ - ZAZ$.

This $Z\ '$ can be shown to be an ϵ^2-approximate pseudo-inverse, whenever Z is an ϵ-approximate pseudo-inverse (see Frederickson & Benson [1986]). This suggests the recursive definition $Z^{(n+1)} = 2Z^{(n)} - Z^{(n)}AZ^{(n)}$, $Z^{(0)} = Z$ with $n \in \mathbf{Z}^+$. It has been shown (Ben-Israel & Cohen [1966], Ben-Israel [1965, 1966] and Ben-Israel & Greville [1974]) that, in the limit $n \longrightarrow \infty$, $Z^{(n)}$ tends to the Moore-Penrose pseudo-inverse A^+, of A. However, as observed by Södeström and Stewart [1974] there are some important numerical considerations required for the use of this approach : The iteration must be started with a rather restricted class of matrices and, even for matrices that are moderately ill-conditioned the convergence will be slow. Also, for a $p \times q$ matrix, each iteration requires about $2pq^2$ floating-point multiplications which, in addition to the possible slow convergence, represents a serious drawback (see Södeström and Stewart [1974] for further details). We also note that, when dealing with large sparse linear systems, it is not feasible, to explicitly construct A^+.

We observe that, in the limit, applying (1.2) based on Z is equivalent to performing a single iteration with A^+ in its place. Doing this, the solution u' to the least squares problem is given by

$$u' = (I - A^+A)u^{(0)} + A^+f \tag{2.4}$$

Since (see Ben-Israel & Greville [1974, p.70]) $A^+A = P_{R(A^*)} = P_{N(A)^\perp}$, we have $I - A^+A = P_{N(A)}$, where P denotes the orthogonal projector onto the indicated subspace. Thus, when $u^{(0)} \perp N(A)$ we get the Moore-Penrose solution u^+. On the other hand, when $u^{(0)}$ has some null space component, we get a minimizing solution $u' = (P_{N(A)}u^{(0)}) + A^+f$ which is closest to $u^{(0)}$ (since the projection is orthogonal). This approach leads to an alternate proof of T2.1.

In the next chapter we will show how approximate inverses and approximate pseudo-inverses can be applied to the solution of large sparse linear systems through a multigrid algorithm. We will solve certain linear systems constructed from the discretization of boundary value problems with periodic and non-periodic boundary conditions. These results represent an extension of current experience in this area.

3. Least Squares-Approximate Inverses and Approximate Pseudo-Inverses

In this section we introduce the concepts of a Least Squares (LSQ)-approximate inverse and that of a Least Squares-approximate pseudo-inverse of a given matrix A. The idea of an LSQ-approximate inverse, along with several other approximate inverses, was first introduced by Benson [1973] and has been utilized in the construction of different iterative methods that either apply it directly or use it to build more sophisticated approximate inverses and approximate pseudo-inverses. Its applicability ranges from the solution of several different spline approximation problems and boundary value problems by means of linear stationary methods (Benson [1973], Benson & Frederickson [1982] and Benson, Krettman & Wright [1984]) and preconditioned conjugate gradient methods (Benson, Krettman & Wright [1982] and Benson & Frederickson [1986]), to the multigrid solution of boundary value problems in two dimensions (Baumgardner & Frederickson [1985]) and Frederickson & Benson [1986])

Assume we are given a linear system of equations :

$$Au = f \ , \ \ A \in \mathbf{M}_n(\mathbf{R}), \ \ u,f \in \mathbf{R}^n \tag{3.1}$$

where A is a sparse matrix (i.e. the ratio of the number of nonzero entries to the number of zero entries, in A is small). To solve (3.1) numerically, we build an approximate inverse $B \in \mathbf{M}_n(\mathbf{R})$ (or approximate pseudo-inverse, if A is singular). Of course, we could consider $B = A^{-1}$, but this is not practical in most cases of interest. Thus, we restrict the candidates B to those which have a particular sparsity pattern. For example, we can consider B to have the same sparsity pattern as A (that is : $b_{ij} \neq 0$ *iff* $a_{ij} \neq 0$, $i,j = 1, 2, \cdots, n$). We would like to have BA equal to I, but this gives rise to n overdetermined independent systems that cannot be solved exactly. We elect to solve them in a least squares sense. Defining the Frobenius norm of a matrix $G \in \mathbf{M}_{m \times n}$ as (see, for example, Stewart [1973, p.173])

$$\| G \|_F = (\sum_{i=1}^{m} \sum_{j=1}^{n} g^2{}_{ij})^{\frac{1}{2}}$$

the least squares solution of the independent overdetermined systems, amounts to the minimization of $\| I - BA \|_F$ over all matrices with a given sparsity pattern. We observe that the relation R defined between matrices P, $Q \in \mathbf{M}_n(\mathbf{R})$ by :

$$\begin{aligned} P \,\mathrm{R}\, Q \text{ } &\textit{iff} \text{ } (\textit{P has the same sparsity pattern as Q}) \\ &\textit{iff} \text{ } (q_{ij} \neq 0 \textit{ iff } p_{ij} \neq 0), \end{aligned}$$

is an equivalence relation. As usual, we denote by $[P]$ the class of $P \in \mathbf{M}_n(\mathbf{R})$ under the relation R. The above ideas lead to the following definition :

Definition 3.1 : *Let $A, P \in \mathbf{M}_n(\mathbf{R})$ and let $[P]$ denote the class of $P \in \mathbf{M}_n(\mathbf{R})$ with respect to the relation* R. *We call B an LSQ-approximate inverse of A if*

$$\| I - BA \|_F = \min_{M \in [P]} \| I - MA \|_F \ .$$

Benson & Frederickson [1982] give a similar definition in terms of a more specialized set of sparsity patterns.

Since B need not have the same sparsity pattern as A, we have some flexibility for approximating more difficult parts of A^{-1}. We also observe that the i-th row of B is determined from only a few of the rows of A. The precise form of the linear system for the i-th row is the following : Let B_i denote the row-vector corresponding to the i-th row of the matrix $B \in [P]$, for some $P \in \mathbf{M}_n(\mathbf{R})$. Let B_i have nonzero entries in positions $i_1, i_2, \ . \ . \ . \ , i_r$ and let β_i be the row-vector built from the nonzero entries in B_i, i.e. $\beta_i \equiv (b_{ii_1}, b_{ii_2}, \ . \ . \ . \ , b_{ii_r})$. Then β_i is required to satisfy :

$$\beta_i C_A = e_i, \tag{3.3}$$

where e_i is the i-th canonical basis row-vector of $\mathbf{R}^n$ and C_A is the following $(r \times n)$ block extracted from A according to the positions of the nonzero entries in the i-th row of B :

$$C_A = \begin{pmatrix} A_{i_1} \\ A_{i_2} \\ \cdots \\ A_{i_r} \end{pmatrix} = \begin{pmatrix} a_{i_1 1} & a_{i_1 2} & \cdots & a_{i_1 n} \\ a_{i_2 1} & a_{i_2 2} & \cdots & a_{i_2 n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{i_r 1} & a_{i_r 2} & \cdots & a_{i_r n} \end{pmatrix} \qquad (3.4)$$

Some columns of C_A may be identically zero. Observe that the rows forming the block C_A corresponding to the i-th row of B are those rows A_j of A such that :

$$(A_j \textit{ is in } C_A \textit{ iff } b_{ij} \neq 0). \qquad (3.5)$$

Of course, when actually calculating the approximate inverse, the possible null columns in C_A are discarded.

Certainly, D3.1 does not guarantee that $\rho(G) = \rho(I - BA) < 1$. Nevertheless, as $\rho(G) \leq \| G \|_F, \forall G \in \mathbf{M}_n(\mathbf{R})$, it is reasonable to expect that the minimization of $\| G \|_F$ will lead, in some cases, to an iteration matrix G with an spectral radius less than one. This has been the case for example in Benson & Frederickson [1982], Benson, Krettman & Wright [1984] and is the case for the results in Chapter II. Thus, in many cases of interest, the LSQ-approximate inverse of a nonsingular matrix is a nonsingular matrix that is a useful approximate inverse for the construction of completely consistent iterative methods (in the sense of D1.1) to numerically solve (3.1).

In some cases, matrices arising from the discretization of higher dimensional problems, are expressible as linear combinations of Kronecker products (also called tensor or cross products in the literature) of the corresponding matrices for the one dimensional case (e.g. certain spline approximation problems and separable partial differential equations in rectangular parallelepipeds). When building LSQ-approximate inverses B for these types of matrices, it is reasonable to choose B's that have a sparsity pattern which has also a Kronecker product structure. We will show later how, in some cases, considerable computational effort can be saved by taking advantage of this structure. Following Ben-Israel & Greville [1974, p.41], we define a Kronecker product :

Definition 3.5 : *The Kronecker Product* $C \otimes D$ *of the two matrices* $C = [c_{ij}] \in \mathbf{M}_{m\times n}(\mathbf{R})$, $D = [d_{ij}] \in \mathbf{M}_{p\times q}(\mathbf{R})$, *is the* $mp\times nq$ *matrix* $K = [k_{ij}] \in \mathbf{M}_{mp\times nq}(\mathbf{R})$, *expressed in partitioned form as :*

$$K \equiv C\otimes D = \begin{bmatrix} c_{11}D & c_{12}D & \cdots & c_{1n}D \\ c_{21}D & c_{22}D & \cdots & c_{2n}D \\ \cdots & \cdots & \cdots & \cdots \\ c_{m1}D & c_{m2}D & \cdots & c_{mn}D \end{bmatrix} .$$

Observe that the i,j-th entry in K is given by : $k_{ij} = c_{\alpha\beta} d_{\mu\nu}$, where $i = (\alpha - 1)p + \mu$ and $j = (\beta - 1)q + \nu$.

Some properties of this product are (see Jacobson [1953]) :

$$(C\otimes D)(P\otimes Q) = CP\otimes DQ \ , \tag{3.6}$$

$$(C\otimes D)^t = C^t \otimes D^t \ , \tag{3.7}$$

where precedence of ordinary matrix multiplication over Kronecker product is to be understood.

If the matrix to be approximately inverted has the form $A = A' \otimes A''$ and we are looking for an LSQ-approximate inverse $B \in [R\otimes S]$, for some $R \in \mathbf{M}_m(\mathbf{R})$, $S \in \mathbf{M}_n(\mathbf{R})$, much work can be saved by applying theorem 3.1 below. Though we cannot use it for our experiments since they arise from differential equation problems, this theorem finds application in the numerical solution of linear systems arising, for example, from cubic spline approximation problems.

Theorem 3.1 : *Let* $A = A' \otimes A''$, $A' \in \mathbf{M}_m(\mathbf{R})$, $A'' \in \mathbf{M}_n(\mathbf{R})$. Let $B\,' \in [R]$, $B\,'' \in [S]$, be LSQ-approximate inverses of A', A'', respectively. Then, the LSQ-approximate inverse $B \in [R\otimes S]$ of A satisfies :

$$B = B\,' \otimes B\,'' \ . \tag{3.8}$$

Since this result is not essential to our developments, we refer the reader to Ap.V below for a proof.

When A in (3.1) is singular, we have to modify D3.1 to have the desired null space behavior for an approximate pseudo-inverse (see D2.2). First, we introduce two definitions from Ben-Israel & Greville [1974, p.82] :

Definition 3.2 : *Let $A \in \mathbf{M}_n(\mathbf{R})$ be the matrix representation of a linear transformation of R^n. Let $S \subseteq \mathbf{R}^n$ be a subspace. We denote by $A_{[S]}$ the restriction of A to the subspace S, defined as the linear transformation*

$$\begin{aligned} &A_{[S]} : S \longrightarrow A(S) \\ &A_{[S]}x = Ax, \quad \forall x \in S. \end{aligned}$$

We remark that the notation $A_{[S]}$ should not be confused with the notation $[A]$ for the equivalence class of A under the relation R.

The counterpart of the above definition is :

Definition 3.3 : *Let A be a linear transformation from $S \subseteq \mathbf{R}^n$ to $\mathbf{R}^n$. Then we define $ext(A)$, the extension of A to all of $\mathbf{R}^n$, by*

$$ext(A) = \begin{cases} Ax & \text{if } x \in S \\ 0 & \text{if } x \in S^{\perp} \end{cases}$$

In light of D3.2 and D3.3 we see that, if $A \in \mathbf{M}_n(\mathbf{R})$, then $ext(A_{[S]}) = AP_S$, where P_S is the orthogonal projector onto S.

Consider the linear system (3.1) with A singular. Since the map $A_{[N^{\perp}(A)]}$ is an isomorphism of $N^{\perp}(A)$ onto $R(A)$, the map $A^{-1}{}_{[N^{\perp}(A)]}$ is well defined. Let B be a linear map of $\mathbf{R}^n$, such that $B_{[R(A)]} = A^{-1}{}_{[N^{\perp}(A)]}$. Observe that then the corresponding extensions $B_{[R(A)]}P_{R(A)}$, $A_{[N^{\perp}(A)]}P_{N^{\perp}(A)}$, satisfy :

$$\begin{aligned} &ext(B_{[R(A)]})ext(A_{[N^{\perp}(A)]})ext(B_{[R(A)]}) \\ &\quad = B_{[R(A)]}P_{R(A)}A_{[N^{\perp}(A)]}P_{N^{\perp}(A)}B_{[R(A)]}P_{R(A)} \\ &\quad = B_{[R(A)]}P_{R(A)}A_{[N^{\perp}(A)]}B_{[R(A)]}P_{R(A)} \\ &\quad = A^{-1}{}_{[N^{\perp}(A)]}P_{R(A)}A_{[N^{\perp}(A)]}A^{-1}{}_{[N^{\perp}(A)]}P_{R(A)} \\ &\quad = A^{-1}{}_{[N^{\perp}(A)]}P_{R(A)} \\ &\quad = B_{[R(A)]}P_{R(A)} \\ &\quad = ext(B_{[R(A)]}). \end{aligned}$$

Thus, $ext(B_{|R(A)|})$ is a pseudo-inverse (in the sense of D2.2) of $ext(A_{|N^\perp(A)|})$, since the orthogonality conditions are satisfied by definition. Observe also that, in this case, $ext(A_{|N^\perp(A)|})$ is precisely A. Therefore $ext(B_{|R(A)|})$ is a pseudo-inverse of A.

Thus, for a singular A, if we can find an approximate inverse of $A_{|N^\perp(A)|}$, with the null space behavior taken care of by suitable projectors, we can construct an approximate pseudo-inverse of A, in the sense of D2.2. We thus have (see Frederickson & Benson [1986]) :

Definition 3.4 : *The LSQ-approximate pseudo-inverse of a matrix* $A \in \mathbf{M}_n(\mathbf{R})$ *is defined as the matrix*

$$Z = P_{N^\perp(A)} B \; P_{R(A)} \tag{3.3}$$

where B *minimizes the Frobenius norm* $\| I - BA \|_F$, *subject to the constraint* $B \in [Q]$, *for some* $Q \in \mathbf{M}_n(\mathbf{R})$.

Note that, as for D3.1, there is no guarantee that B will be a good approximate inverse of $A_{|N^\perp(A)|}$. Nevertheless, good experimental results have been obtained in several different situations (see references at the beginning of the section) and, as we show in the next chapter, D3.4 can be used successfully for the numerical solution of several linear systems constructed from the discretization of second and fourth order boundary value problems with periodic boundary conditions. Also, for D3.4 to be of any practical use, $N(A)$ must be known. For the singular problems in this thesis, this is not a difficulty and, moreover, the application of the required orthogonal projectors is inexpensive.

4. The Fast Approximate Inverse (FAPIN)

In this section we present the algorithm FAPIN (Fast Approximate Inverse), introduced by Frederickson [1975], along with a convergence proof. This algorithm can be used to solve certain large sparse linear systems of equations. Given a system $Au = f$, FAPIN effectively builds an approximate inverse (see §2) to A in a multigrid fashion. This approximate inverse is never built explicitly. FAPIN, in its original version for non-singular problems has proven successful in the numerical solution of some second order boundary value and eigenvalue problems in one and two dimensions (see Chew [1977] and Liong [1977]). Moreover, an extension of FAPIN to singular problems (in which case the acronym FAPIN stands for Fast Approximate Pseudo-Inverse), which we describe later in this section, has given good results when applied to the solution of some second order boundary value problems in two dimensions (see Baumgardner & Frederickson [1985], Frederickson & Benson [1986] and Benson & Frederickson [1987]).

Given the wide variety of multigrid algorithms that appear in the literature, we do not intend to give here a general definition that would embrace all of them. Rather, we use a more restricted one, suitable to our needs, and refer the reader to the bibliography for other points of view (see, for example, Hackbusch & Trottenberg [1981] and references therein).

We will develop our ideas in an algebraic framework, though we will relate them to variational formulations of multigrid algorithms whenever possible (see, for example, McCormick & Ruge [1982], Douglas [1984], Maitre & Musy [1984] , Bank & Douglas [1985] and McCormick [1984, 1985]).

Assume we are given a linear system of equations

$$Au = f \ , \tag{4.1}$$

where $A \in \mathbf{M}_n(\mathbf{R})$, is a symmetric and positive-definite matrix. From now on we will interpret matrices as the expression, in terms of the corresponding canonical bases, of linear maps defined between spaces of the form $\mathbf{R}^s$, for some $s \in \mathbf{Z}^+$, endowed with the inner product : $<x,y> = x^t y$. We want to solve (4.1) iteratively by means of a linear stationary method (see §1). To this end we want to build an approximate inverse (see D2.1) F to the matrix A such that the iterative process :

$$\begin{cases} r^{(n)} & = f - Au^{(n)} \\ u^{(n+1)} & = u^{(n)} + Fr^{(n)} \end{cases} \tag{4.2}$$

converges to the solution $\overline{u}$ of (4.1). If $\rho(I - FA) < 1$ we will obtain a completely consistent convergent iterative method as defined in D1.1 and D1.3. To build F, we define a sequence of nested Euclidean spaces

$$\mathbf{X}_0 \subset \mathbf{X}_1 \subset \cdots \subset \mathbf{X}_k \subset \cdots \subset \mathbf{X}_M \equiv \mathbf{R}^n ,$$

and refer to $\mathbf{X}_k$ as 'grid k'. Between the above spaces we define the full rank maps :

$$P_k : \mathbf{X}_k \rightarrow \mathbf{X}_{k-1}$$
$$Q_{k-1} : \mathbf{X}_{k-1} \rightarrow \mathbf{X}_k ,$$

such that :

$$P_k = (Q_{k-1})^t , \ 1 \leq k \leq M. \tag{4.3}$$

P_k is a restriction operator, usually called 'collection', while Q_{k-1}, usually called 'interpolation', is the matrix representation of the containment ($x \in \mathbf{X}_{k-1}$ *implies* $x \in \mathbf{X}_k$) in the chosen bases for the above spaces. Furthermore, letting A_M be the matrix A of (4.1), we define the maps

$$A_{k-1} : \mathbf{X}_{k-1} \rightarrow \mathbf{X}_{k-1}$$

by :

$$A_{k-1} = P_k A_k Q_{k-1} , \ 1 \leq k \leq M. \tag{4.4}$$

We now prove an important property of the matrices A_k defined above :

Lemma 4.1 : *If A_k is symmetric and positive definite then so is A_{k-1}.*

Proof :

First we prove symmetry. Since A_k is symmetric by assumption :

$$\begin{aligned} {A_{k-1}}^t &= (P_k A_k Q_{k-1})^t \\ &= {Q_{k-1}}^t {A_k}^t {P_k}^t \\ &= P_k A_k Q_{k-1}, \ \textit{by}\ (4.3) \\ &= A_{k-1} . \end{aligned}$$

Positive definiteness is shown in a similar fashion. Since A_k is positive definite by assumption :

$$\begin{aligned} &\forall\, x \in \mathbf{X}_{k-1}\,,\ x \neq 0, \\ x^t A_{k-1} x &= x^t (P_k A_k Q_{k-1})^t x \\ &= x^t Q_{k-1}{}^t A_k{}^t P_k{}^t x \\ &= (Q_{k-1} x)^t A_k (Q_{k-1} x) \\ &= w^t A_k w > 0,\ \textit{since}\ w = Q_{k-1} x \text{ and } Q_{k-1}\ \textit{is full rank} \quad \square \end{aligned}$$

Since the A_k, $0 \leq k \leq M$ are symmetric and positive definite, we can define new inner products, in the usual way, on the corresponding spaces $\mathbf{X}_k$:

$$<x,y>_{A_k} \equiv <A_k x,y>,\ x,y \in \mathbf{X}_k, \tag{4.5}$$

and the corresponding norms :

$$\| \, x \, \|_{A_k} \equiv (<x,x>_{A_k})^{1/2}. \tag{4.6}$$

We note that when the spaces $\mathbf{X}_k$ are the coefficient spaces corresponding to a nested sequence of finite element spaces $\mathbf{H}_k$, the operator Q_{k-1} is just the interpolation operator resulting from writing a function in $\mathbf{H}_{k-1}$, as a linear combination of basis functions for $\mathbf{H}_k$. This fact is used in Ap.III to calculate the interpolation and collection operators required for our numerical experiments. Furthermore in this case, if we define $P_k \equiv Q_{k-1}{}^t$ and we let A_k, $0 \leq k \leq M$ be the assembled element matrices derived from the Ritz-Galerkin formulation of a given boundary value problem in the space $\mathbf{H}_k$, then (4.4) follows automatically (see Nicolaides [1977, p.896]). Also, $<.,.>_{A_k}$ is just the energy (associated with the corresponding energy functional) in the finite element space $\mathbf{H}_k$. Thus we will use the notation 'energy inner product' to denote (4.5). Consequently, we will speak of 'orthogonality in the energy inner product' or 'A_k-orthogonality' and denote it by $\perp_{A_k}$. We will also call the norm in (4.6) the 'energy norm ' or 'A_k-norm'.

With the above notation, we now observe that any $x \in \mathbf{X}_k$ has a unique representation :

$$x = x' + x'',\ x' \in N(P_k A_k),\ x'' \in N^{\perp_{A_k}}(P_k A_k)\ ,$$

and where $N^{\perp_{A_k}}(P_k A_k)$ denotes the A_k-orthogonal complement of the null space of $P_k A_k$ in $\mathbf{X}_k$. As will be shown later, this decomposition will prove essential in understanding the convergence of the algorithm FAPIN. We now give some properties of the previously defined operators :

Lemma 4.2 : *Let A_k, P_k be as above. Then :*

$$N(P_k A_k) = A_k^{-1}(N(P_k)) \ .$$

Proof :

The case $x = 0$ is trivial. Thus assume $x \neq 0$. Then :

$$\begin{aligned} x \in N(P_k A_k) \ &\mathit{iff} \ (P_k A_k)x = 0 \\ &\mathit{iff} \ P_k(A_k x) = 0 \\ &\mathit{iff} \ P_k y = 0, \ y = A_k x \neq 0 \\ &\mathit{iff} \ y \in N(P_k) \\ &\mathit{iff} \ x \in A_k^{-1}(N(P_k)) \quad \square. \end{aligned}$$

Lemma 4.3 : *Let A_k, P_k and Q_{k-1} be as above. Then :*

$$N^{\perp_{A_k}}(P_k A_k) = R(Q_{k-1}) \ ,$$

where $R(Q_{k-1})$ denotes the range of Q_{k-1}.

Proof :

Assume $x \neq 0$. Then :

$$\begin{aligned} x \in N^{\perp_{A_k}}(P_k A_k) \ &\mathit{iff} \ <A_k x, w> = 0, \ \forall \ w \in N(P_k A_k) = A_k^{-1}(N(P_k)) \\ &\mathit{iff} \ <A_k x, A_k^{-1} y> = 0, \ \forall \ y \in N(P_k) \\ &\mathit{iff} \ <x, A_k^t A_k^{-1} y> = 0, \ \forall \ y \in N(P_k) \\ &\mathit{iff} \ <x, y> = 0, \ \forall \ y \in N(P_k) \\ &\mathit{iff} \ x \in N^{\perp}(P_k) = R(Q_{k-1}) \ , \ by \ (4.3) \quad \square. \end{aligned}$$

Lemma 4.4 : *Let Q_{k-1} be as above, and let $x \in \mathbf{X}_{k-1}$. Then :*

$$\| \ Q_{k-1} x \ \|_{A_k} = \| \ x \ \|_{A_{k-1}} \ .$$

Proof :

$$\| \ Q_{k-1} x \ \|_{A_k}^{\ 2} = <A_k \, Q_{k-1} x, Q_{k-1} x>$$

$$= <Q_{k-1}{}^t A_k Q_{k-1} x , x>$$
$$= <A_{k-1} x , x> \text{ , } by \text{ (4.3) and (4.4)}$$
$$= \| \, x \, \|_{A_{k-1}}{}^2 \quad \square .$$

Next we define the concept of a nested ϵ-approximate inverse Z_k to a given non-singular matrix A_k. This is an adaptation of the somewhat more general definition given in Frederickson & Benson [1986]. It tailors the idea of an ϵ-approximate inverse (see D2.1) to a multigrid environment.

Definition 4.1 : *We call the set of maps* $Z_l : \mathbf{X}_l \rightarrow \mathbf{X}_l$, $0 \le l \le k$ *a sequence of nested* ϵ*-approximate inverses of the set of maps* $A_k : \mathbf{X}_k \rightarrow \mathbf{X}_k$, $0 \le l \le k$ *if there exists* $\epsilon < 1$, *independent of* l, *such that :*

$$\| \, (I - Z_l A_l) x \, \|_{A_l}{}^2 \le \epsilon^2 \| \, x' \, \|_{A_l}{}^2 + \| \, x'' \, \|_{A_l}{}^2 \, , \ \forall \, x \in \mathbf{X}_l \ , \ \ 0 \le l \le k \ ,$$

where $x = x' + x''$, $x' \in N(P_l A_l)$ and $x'' \in N^{\perp_{A_l}}(P_l A_l)$. *When* $l = 0$ *we define* $x'' = 0$ *and* $x' = x$.

We are now prepared to define the algorithm FAPIN. We adapt the definition given in Frederickson & Benson [1986] to our case. We will avoid additional subscripts by letting the symbol $\longleftarrow$ denote replacement.

Definition 4.2 : *With the above definitions and notation, the algorithm* FAPIN (Fast Approximate Inverse) *is defined by the following pseudo-code :*

```
FAPIN ($k$,$r_k$) {
    if ($k = 0$) then
        $w_0 \longleftarrow Z_0 r_0$
    else {
    (1) restrict residual to next coarser grid
        $r_{k-1} \longleftarrow P_k r_k$
    (2) apply FAPIN recursively to this new residual problem
```

```
        $w_{k-1} \leftarrow$ FAPIN $(k, r_k)$

    (3) interpolate result to finer grid

        $w_k \leftarrow Q_{k-1} w_{k-1}$

    (4) perform one iteration of (4.2) on finer grid with $Z_k$

        $s_k \leftarrow r_k - A_k w_k$

        $w_k \leftarrow w_k + Z_k s_k$

    }

    return $w_k$

}
```

Steps (1) through (3) in the 'else' part of FAPIN are usually called the 'coarse grid correction', while step (4) is usually regarded as a 'relaxation' or 'smoothing' step.

We can now interpret D4.1. Observe that coarse grid correction terms, for grid k, lie in $R(Q_{k-1})$. But, since by Lemma 4.3 we have $R(Q_{k-1}) = N^{\perp_{A_k}}(P_k A_k)$, we are requiring Z_k to act as an ϵ-approximate inverse (see D2.1) on the subspace $N(P_k A_k) = \mathbf{R}^{\perp_{A_k}}(Q_{k-1})$ of $\mathbf{X}_k$, that is, on the set of vectors of $\mathbf{X}_k$ which are not affected by the coarse grid correction. This is easily seen by letting $x \in N(P_k A_k)$ in D4.1.

We note that $R(Q_{k-1})$ and $N(P_k A_k) = R^{\perp_{A_k}}(Q_{k-1})$ are, respectively, the subspaces of $\mathbf{X}_k$ of 'smooth' and 'oscillatory' vectors defined in McCormick & Ruge [1982], McCormick [1984] and McCormick [1985].The terminology stems from the 'modal analysis' framework for the study of multigrid algorithms, introduced by Brandt [1977]. Moreover, we observe the close relationship of FAPIN to the well known V-cycle algorithm, as pointed out in Lemma 4.1 of McCormick [1984].

To interpret FAPIN we first examine a two-grid algorithm, that is, $k = 0, 1$. Assume we want to solve the system

$$A_1 u = f \tag{4.7}$$

using the iterative process :

$$\begin{cases} r_1 & \longleftarrow f - A_1 u \\ w_1 & \longleftarrow F_1 r_1 \\ u & \longleftarrow u + w_1 \end{cases} \tag{4.8}$$

where F_1 is an ϵ-approximate inverse to A_1. Observe that if the solution to the residual problem

$$A_1 w_1 = r_1 \tag{4.9}$$

was known, we could immediately solve (4.7). The two-grid FAPIN algorithm seeks the iterative solution of (4.9), by using as an initial guess the coarse grid correction. This correction is the result of interpolating an approximate solution of the problem

$$A_0 w_0 = r_0 \equiv P_1 r_1 \ . \tag{4.10}$$

Observe that if $\overline{w}_0$ is the solution of (4.10) then $\overline{w}_1 \equiv Q_0 \overline{w}_0$ is the solution of $P_1 A_1 w_1 = P_1 r_1$ in $\mathbf{X}_{k-1}$, since $P_1 A_1 Q_0 \overline{w}_0 = P_1 r_1$ would be satisfied. Thus (4.10) is the formulation of (4.9) restricted to $\mathbf{X}_{k-1}$ and hence an approximate solution to (4.10) is an approximate solution to (4.9) on $R(Q_{k-1})$. The approximate solution of (4.10) is obtained by applying one relaxation step with a linear stationary method based on Z_0, an ϵ-approximate inverse of A_0 (when $k = 0$, the concepts of a sequence of nested approximate inverses and an approximate inverse are equivalent), and starting with the vector $w_0 = 0$, that is :

$$\begin{cases} \rho & \longleftarrow r_0 - A_0 w_0 \\ \delta & \longleftarrow Z_0 \rho \\ w_0 & \longleftarrow w_0 + \delta \end{cases}$$

Observe however that the above relaxation step is equivalent to the replacement $w_0 \longleftarrow Z_0 r_0$, since the initial w_0 is zero.

Once the initial guess $w_1 \longleftarrow Q_0 w_0$ (i.e. the coarse grid correction) has been calculated, FAPIN proceeds to apply one relaxation step witn a Z_1, a member of a sequence of nested ϵ-approximate inverses, to 'smooth' the residual r_1 :

$$\begin{cases} s_1 & \leftarrow & r_1 - A_1 w_1 \\ w_1 & \leftarrow & w_1 + Z_1 s_1 \end{cases}$$

We now can write the above process in an algorithmic form :

$$\begin{cases} r_1 & \leftarrow & f \; - A_1 u \\ w_1 & \leftarrow & \text{FAPIN}(1, r_1) \\ u & \leftarrow & u + w_1 \end{cases}$$

where we have defined FAPIN $(1, r_1)$ as the following sequence of operations :

(1) $r_0 \leftarrow P_1 r_1$

(2) $w_0 \leftarrow Z_0 r_0$

(3) $w_1 \leftarrow Q_0 w_0$

(4) $s_1 \leftarrow r_1 - A_1 w_1$

(5) $w_1 \leftarrow w_1 + Z_1 s_1$

In view of the above it is now clear that the step involving F_1 in (4.8) can be expressed as :

$$\begin{aligned} F_1 r_1 &= Q_0 Z_0 P_1 r_1 + Z_1 (r_1 - (A_1 Q_0 Z_0 P_1 r_1)) \\ &= Z_1 r_1 + (Q_0 Z_0 P_1 - Z_1 A_1 Q_0 Z_0 P_1) r_1 \\ &= (Z_1 + (I - Z_1 A_1) Q_0 Z_0 P_1) r_1 . \end{aligned}$$

Thus we may define :

$$F_1 \equiv Z_1 + (I - Z_1 A_1) Q_0 Z_0 P_1 \, . \tag{4.11}$$

The above definition generalizes easily to the case of several grids. We need only realize that the job done by Z_0 on grid $k = 0$ to solve (4.8) on grid $k = 1$ would be done by F_{k-1} to solve the corresponding residual problem on grid k. This leads to the following recursive definition of FAPIN for the nonsingular case (the more general singular case in described in Frederickson & Benson [1986]) :

Definition 4.3 : *With the above definitions and notation, a Fast Approximate Inverse (FAPIN) $F_k : \mathbf{X}_k \rightarrow \mathbf{X}_k$ to a given $A_k : \mathbf{X}_k \rightarrow \mathbf{X}_k$ is the linear operator (matrix) constructed from a sequence of nested ϵ-approximate inverses Z_k and defined recursively by :*

$$F_0 = Z_0$$
$$F_k = Z_k + (I - Z_kA_k)Q_{k-1}F_{k-1}P_k \ , \ 1 \le k \le M \ .$$

We now prove the convergence of the algorithm FAPIN. That is, we prove that under certain conditions, the operator F_k defined in D4.3 is an ϵ-approximate inverse of A_k with $\epsilon < 1$, thus yielding a convergent iterative method (see T1.2.(ii)).

Theorem 4.1 : *If for $\epsilon < 1$, Z_l is a sequence of nested ϵ-approximate inverses for $0 \le l \le k$, then F_k, as defined in* D4.3 *is an ϵ-approximate inverse with the same ϵ.*

Proof :

We must show :

$$\| (I - F_kA_k)x \|_{A_k} \le \epsilon \| x \|_{A_k} \ , \ \forall\, x \in \mathbf{X}_k \ .$$

We prove the theorem by induction on k.

(i) $k = 0$

Since by definition $F_0 = Z_0$ and, by hypothesis, Z_0 is a member of a sequence of nested ϵ-approximate inverses, we have (in the notation of D4.1) :

$$\begin{aligned} \| (I - F_0A_0)x \|_{A_0}^{\ 2} &= \| (I - Z_0A_0)x \|_{A_0}^{\ 2} \ , \ \forall\, x \in \mathbf{X}_0 \\ &\le \epsilon^2 \| x' \|_{A_0}^{\ 2} + \| x'' \|_{A_0}^{\ 2} \ , \\ &\le \epsilon^2 \| x' \|_{A_0}^{\ 2} \ , \ \textit{since } x'' = 0 \textit{ when } k = 0 \end{aligned}$$

(ii) Assume the theorem holds for F_{k-1}, that is :

$$\| (I - F_{k-1}A_{k-1})x \|_{A_{k-1}} \le \epsilon \| x \|_{A_{k-1}} \ , \ \forall\, x \in \mathbf{X}_{k-1} \ .$$

Then :

$$\begin{aligned} (I - F_kA_k)x &= \Big(I - (Z_k + (I - Z_kA_k)Q_{k-1}F_{k-1}P_k)A_k\Big)x \\ &= \Big((I - Z_kA_k)(I - Q_{k-1}F_{k-1}P_kA_k)\Big)x \\ &= \Big((I - Z_kA_k)(I - Q_{k-1}F_{k-1}P_kA_k)\Big)(x' + x'') \ , \end{aligned}$$

where $x' \in N(P_kA_k)$ and $x'' \in N^{\perp_{A_k}}(P_kA_k)$. But, now, we have :

$$(I - F_kA_k)x' = \Big((I - Z_kA_k)(I - Q_{k-1}F_{k-1}P_kA_k)\Big)x'$$

$$= (I - Z_k A_k) x' \ , \ \ since \ x' \in N(P_k A_k)$$

By Lemma 4.3, we have $x'' \in \mathbf{R}(Q_{k-1})$. Thus, there exists $y \in \mathbf{X}_{k-1}$ such that : $x'' = Q_{k-1} y$. Hence :

$$\begin{aligned}(I - F_k A_k) x'' &= \Big((I - Z_k A_k)(I - Q_{k-1} F_{k-1} P_k A_k) \Big) x'' \\ &= \Big((I - Z_k A_k)(I - Q_{k-1} F_{k-1} P_k A_k) \Big) Q_{k-1} y \\ &= (I - Z_k A_k)(Q_{k-1} y - Q_{k-1} F_{k-1} A_{k-1} y) \\ &= (I - Z_k A_k) Q_{k-1} (I - F_{k-1} A_{k-1}) y \\ &= (I - Z_k A_k) \psi \ ,\end{aligned}$$

where we have defined : $\psi \equiv Q_{k-1}(I - F_{k-1} A_{k-1}) y$.

Therefore :

$$\begin{aligned}\| (I - F_k A_k) x \|_{A_k}{}^2 &= \| (I - F_k A_k)(x' + x'') \|_{A_k}{}^2 \\ &= \| (I - Z_k A_k)(x' + \psi) \|_{A_k}{}^2 \ .\end{aligned}$$

Observe that :

$$\begin{aligned}\| \psi \|_{A_k} &= \| Q_{k-1}(I - F_{k-1} A_{k-1}) y \|_{A_k} \\ &= \| (I - F_{k-1} A_{k-1}) y \|_{A_{k-1}} \ , \ by \ Lemma \ 4.4 \\ &\leq \epsilon \| y \|_{A_{k-1}} \ , \ by \ the \ induction \ hypothesis \\ &= \epsilon \| Q_{k-1} y \|_{A_k} \ , \ by \ Lemma \ 4.4 \\ &= \epsilon \| x'' \|_{A_k} \ , \ by \ definition \ of \ y \end{aligned} \qquad (4.12)$$

Note that, since $\psi \in R(Q_{k-1})$, by Lemma 4.3 we have that $\psi \in N^{\perp_{A_k}}(P_k A_k)$ and thus x' and ψ are A_k-orthogonal. Thus :

$$\begin{aligned}\| (I - F_k A_k) x \|_{A_k}{}^2 &= \| (I - F_k A_k)(x' + x'') \|_{A_k}{}^2 \\ &= \| (I - Z_k A_k)(x' + \psi) \|_{A_k}{}^2 \ ,\end{aligned}$$

Since Z_k is by assumption a member of a sequence of nested ϵ-approximate inverses, we finally have :

$$\begin{aligned}\| (I - Z_k A_k)(x' + \psi) \|_{A_k}{}^2 &\leq \epsilon^2 \| x' \|_{A_k}{}^2 + \| \psi \|_{A_k}{}^2 \\ &\leq \epsilon^2 \| x' \|_{A_k}{}^2 + \epsilon^2 \| x'' \|_{A_k}{}^2 \ , \ \ by \, (4.12)\end{aligned}$$

$$= \epsilon^2 \| (x' + x'') \|_{A_k}^{\ 2} \ , \ \textit{by } A_k\textit{-orthogonality}$$
$$= \epsilon^2 \| x \|_{A_k}^{\ 2} \ , \ \textit{by definition of } x' \text{ and } x'' \qquad \square$$

We note that since ϵ is independent of k the rate of convergence of FAPIN does not depend on the number of grids considered. Thus, the rate of convergence is also independent of the number of unknowns in the finest grid. This fact will provide us with a test of multigrid behavior for our numerical experiments.

We now show that FAPIN also yields a completely consistent method (see T1.1(iii)). Let $G_k \equiv I - F_k A_k$ be the iteration matrix corresponding to F_k defined in D4.3. Then, there exists $\epsilon < 1$ such that (by T4.1) :

$$\| G_k x \|_{A_k} \leq \epsilon \| x \|_{A_k} \ , \ \forall x \in \mathbf{X}_k$$

Thus :

$$\frac{\| G_k x \|_{A_k}}{\| x \|_{A_k}} \leq \epsilon < 1 \ , \ \forall\, x \in \mathbf{X}_k \ , \ x \neq 0$$

and hence $\| G_k \|_{A_k} < 1$, which implies $\rho(G_k) < 1$. Therefore $I - G_k = F_k A_k$ is nonsingular (see Atkinson[1978, Theorem 7.10]) and F_k must be nonsingular.

Thus we have shown that conditions 1.1 of §1 are satisfied and therefore FAPIN yields a completely consistent convergent iterative method.

We note that when the spaces $\mathbf{X}_k$ are finite element spaces, the algorithm FAPIN described above is equivalent to the '/-cycle' ('slash cycle') described in McCormick[1984, p.260] and in McCormick[1985, p.635]. Furthermore, the definition of a sequence of nested ϵ-approximate inverses is then equivalent to equation (4.7) of Theorem 4.1 in McCormick[1984] and to the 'smoothing property' defined in Lemma 2.2 of McCormick[1985], which in our notation reads :

$$\| (I - Z_k A_k) e \|_{A_k}^{\ 2} \leq \alpha \| \ Te \|_{A_k}^{\ 2} + \| \ Se \|_{A_k}^{\ 2} \ ,$$

where S and T are the A_k-orthogonal projectors from X_k onto $R(Q_{k-1})$ and $R^{\perp_{A_k}}(Q_{k-1})$, respectively. Moreover, we remark that the rates of convergence $\alpha^{1/2}$ that are proven in the above references (see Theorem 4.1 in McCormick[1984] and Lemma 2.3 in McCormick [1985]) coincide with that implied for FAPIN by T4.1 proved above.

The works by McCormick cited above prove that a Richardson smoothing step (that is, with an approximate inverse of the form cI_k , $c \in \mathbf{R}$, I_k the identity in $\mathbf{X}_k$) satisfies, under certain conditions, the smoothing property. Thus cI_k constitutes, in our notation, a sequence of nested approximate inverses. This suggests trying FAPIN algorithms based on different approximate inverses. In this thesis we consider the case of LSQ-approximate inverses (see D3.1) and those resulting from the application of two successive relaxation steps with an LSQ-approximate inverse (see description in §2). As shown in the next chapter, good experimental results were obtained with FAPIN based on these LSQ and LSQ based approximate inverses for the solution of certain linear systems constructed from finite element discretizations of one and two dimensional second and fourth order boundary value problems. Of course, this does not allow us to conclude that these sets of approximate inverses are in fact sequences of nested approximate inverses, since this is a sufficient but not a necessary condition for the convergence of FAPIN.

Next we briefly describe the extension of the algorithm FAPIN for the solution of linear systems of the form (4.1), where A is singular. We refer the reader to the work by Frederickson & Benson [1986] for further details.

When A in (4.1) is singular, a linear stationary method must be based on an approximate pseudo-inverse of A (see D2.3). Thus, in this case, FAPIN will yield a fast approximate pseudo-inverse. The definition of a sequence of nested approximate inverses $Z_l, 0 \leq l \leq k$ (defined in D4.1) must now be extended to account for the null space behavior of the problem to be solved. Thus, following Frederickson & Benson [1986], we require for each $Z_l, 0 \leq l \leq k$:

$$R(Z_l) \perp N(A_l) \ , \ \ N(Z_l) \perp R(A_l) \ .$$

This defines the required sequence of nested ϵ-approximate pseudo-inverses.

The algorithm FAPIN for singular problems is just that described in D4.2 with the addition of a 'step (0)' to the 'else' part, of the form :

```
(0) project residual onto the range of $A_k$
    $r_k \leftarrow \Pi_k r_k$ ,
```

where Z_k is now a sequence of nested ϵ-approximate pseudo-inverses.

Frederickson & Benson [1986] prove the convergence of the above version of FAPIN by showing that it actually yields an ϵ-approximate pseudo-inverse. Since FAPIN is an ϵ-approximate pseudo-inverse, T2.1 implies that, when the initial guess for the iterative solution of (4.1) with A singular is $u^{(0)} = 0$, then the above FAPIN converges to the Moore-Penrose solution $(A^+ f)$ of (4.1), were A^+ is the Moore-Penrose pseudo-inverse of A.

In Chapter Two of this thesis we show how this last version of FAPIN yields good experimental results when applied to the solution of linear systems constructed from periodic cubic spline and biperiodic bicubic spline discretizations of some second and fourth order periodic boundary value problems in one and two dimensions. The approximate pseudo-inverses used in our experiments were LSQ-approximate pseudo-inverses (see D3.4) and those resulting from the application of two successive relaxation steps based on LSQ-approximate pseudo-inverses (see description in §2). Once again, this convergence does not allow us to conclude that these sets of approximate pseudo-inverses are in fact sequences of nested approximate pseudo-inverses, since Frederickson and Benson [1986] only prove that this is a sufficient but not necessary condition for convergence.

CHAPTER II

Numerical Experiments

In this chapter we present several numerical experiments consisting of the application of the algorithm FAPIN to the solution of large sparse linear systems of equations. The results presented here represent new experience with thi multigrid algorithm. The matrices for these systems arise from the finite element discretizations of several different boundary value problems, both with periodic and non-periodic boundary conditions. The right hand sides are built in such a manner that the exact solutions to the systems are known in each case. We are primarily concerned with fourth order problems, both in one and two dimensions; however, we present some second order problems for the purpose of comparison.

The experiments were all carried out on a Sun-3/180 and were designed with the idea of numerically identifying the general trends in the behavior of the algorithm FAPIN, rather than building production code. For this purpose, the high level environment HL, developed by Benson [1987], proved to be well suited (we briefly describe HL in Ap.IV). Because of this, our measures of efficiency will not consider computing time, but rather number of arithmetic operations. All computations are in double precision.

The chapter is organized as follows : Section 1 describes in detail the model problems used in the different experiments. In Section 2 we describe the different experimental measures used to analyse the results in this thesis. Section 3 presents the results for our one-dimensional experiments. Sections 4 and 5 contain the results for the two-dimensional second and fourth order problems, respectively. Finally, in section 6 we present our final conclusions and some suggestions for further study.

1. Model Problems

In this section we describe the model problems to be considered in this thesis. We divide them into two categories :

(1) Those constructed from the discretization of two-point boundary value problems in the interval $I \equiv [0,\pi]$,

(2) Those constructed from the discretization of boundary value problems in the domain $\Omega \cup \Gamma \equiv [0,\pi] \times [0,\pi] \subset \mathbf{R}^2$.

Thus, we will use the notation 'one-dimensional' and 'two dimensional' problems to reference the above categories. Despite this notation we remark that our model problems are not strictly boundary value problems, but rather, linear systems of equations whose matrices arise from the discretization of the former. Within each of these categories we make a further distinction according to the boundary conditions imposed on the given boundary value problem (BVP). These are :

(1) Periodic boundary conditions (PBP),

(2) Non-Periodic boundary conditions (NPBP).

The general process we use to build our model problems is the following : Given a BVP (in a one or two-dimensional domain) we perform a Ritz-Galerkin finite element discretization (see Strang & Fix [1973]). This leads to a linear system of equations of the form : $Au = g$, $A \in \mathbf{M}_n(\mathbf{R})$, $g \in \mathbf{R}^n$ for some $n \in \mathbf{Z}^+$. We then choose a vector $\overline{u} \in \mathbf{R}^n$ and build the linear system $Au = (A\overline{u})$ (Clearly $\overline{u}$ is the desired solution to the system). Since we are interested in testing the iterative algorithm FAPIN on these systems, we also fix an initial guess $u^{(0)}$ to the solution $\overline{u}$. Thus a model problem will have the general form :

$$\begin{cases} Au = (A\overline{u}) \\ \overline{u},\ u^{(0)}\ \textit{fixed} \end{cases}$$

In the periodic case, however, there are some special characteristics to be considered when we build the corresponding model problems. In these cases, the matrices arising from the discretization of the corresponding BVP's are singular and the general process described above has to be modified to provide enough generality. In every periodic case, considered here, we have $N(A) = \{ \mathbf{1} \}$, where $\mathbf{1}$ is the vector with all components equal to one (see Table 1.1). Thus we choose a certain vector v and remove its component in $N(A)$. This will be our $\overline{u}$ (i.e. the solution to the linear system). Now, since A is symmetric, $N(A) = R^{\perp}(A)$ and thus, if we considered the vector $f = A\overline{u}$ as our right hand side, f would have no components in $N(A)$. Hence, to have full generality, we add to f the vector $\mathbf{1}$ in $N(A)$ and consider $f + \mathbf{1} = (A\overline{u}) + \mathbf{1}$ as the right hand side. The particular choices of solution vectors $\overline{u}$ and initial guesses $u^{(0)}$ that we make and the notation we follow to reference them are shown in Table 1.1.

Table 1.1 : Vectors to build Solutions and Initial Guesses

Notation	Vector
rand	$u_i \in \mathbf{R}$ where $0 \leq u_i \leq 1$ and u_i is random
1	$u_i = 1, \forall i$
0	$u_i = 0, \forall i$

The above vectors were always combined into pairs of solution ($\overline{u}$) and initial guess ($u^{(0)}$) vectors for the corresponding model problem, according to the boundary conditions of the BVP from which it derived. These pairs are the following :

(i) Non-Periodic Boundary Conditions (NPBC)

(1) $\overline{u} = rand,\ u^{(0)} = \mathbf{0}$

(2) $\overline{u} = \mathbf{0},\ u^{(0)} = \mathbf{1}$

(ii) Periodic Boundary Conditions (PBC)

(P) $\overline{u} = rand - s,\ u^{(0)} = \mathbf{0}$, where s is the orthogonal projection of *rand* onto $N(A)$ with respect to the usual inner product in $\mathbf{R}^n$.

We will refer to those non-periodic model problems which are based on the choices (i).1 and (i).2 above as model problems of 'type .1' and 'type .2', respectively. The choice of random vectors for our 'type .1', tries to avoid unnaturally smooth vectors, thus providing us with an extreme situation. On the other hand, the choice (i).2 is the model problem proposed by Young [1971, p.3 & 132], thus making our 'type .2' model problems actual finite element discretizations of constant coefficient homogeneous BVP's (in contrast to our 'type .1' model problems, which do not represent a discrete boundary value problem).

Next we give details for all the model problems considered in this thesis. For brevity we do this in the form of tables. Tables 1.2 and 1.3 contain the model problems arising from BVP's with NPBC and PBC, respectively. We recall that I denotes the interval $[0,\pi]$, while $\Omega \cup \Gamma$ denotes the domain $I \times I \subset \mathbf{R}^2$. Further, we label the segments of Γ by Γ_1, Γ_2, Γ_3 and Γ_4, considered counterclockwise and starting with the axis $y = 0$. We also let $\mathbf{x}$ denote an arbitrary point of $\Omega \cup \Gamma$ and $u_n(\mathbf{x})$ the directional derivative of the function $u(\mathbf{x})$ in the direction of the outward normal to Γ. We note that the solutions $u(\mathbf{x})$ to the non-periodic boundary value problems of Table 1.2 are singular at the corners of the domain Ω *union'gm* (see Stephan [1979] and Strang & Fix [1973, p.263]).

Our finite element bases where built on the elementary basis functions given by Strang & Fix [1973]. For the one-dimensional problems we considered bases of piecewise linear ('roof') functions, cubic B-splines and piecewise cubic Hermite functions. In the two-dimensional cases we considered bases of piecewise bilinear functions, bicubic B-splines and piecewise bicubic Hermite functions. Since the domain $\Omega \cup \Gamma$ is a square, we built these bases as Kronecker products of the corresponding one-dimensional ones. This Kronecker product construction, and the fact that the BVP's considered are separable (see Tables 1.2 & 1.3), implies a Kronecker product decomposition of the corresponding finite element assembled matrices (see, for example, Kaufman & Warner [1984]). We take advantage of this fact to reduce the work involved in setting up the corresponding two-dimensional model problems.

The linear basis functions were used as given in Strang & Fix [1973, p.27], while the B-splines were those given in Strang & Fix [1973, p.61] normalized by a factor of $\frac{6}{4}$ so that they would have a maximum value of 1 in their support. The Hermite basis functions were first tried as given in Strang & Fix [1973, p.56], but the algorithm FAPIN diverged on some preliminary one-dimensional second order experiments. As observed by Greenbaum [1984], the scaling of the basis functions is an important factor in the performance of a multigrid algorithm for the solution of a finite element discretization of a BVP. Following this idea, two other scalings of the Hermite basis were considered. They are described in Ap.I.§3&4 These new scalings produced good convergence results.

The notation used to label the model problems consists of an alphanumeric string composed of three segments of information separated by the character '.' . The first segment is a mnemonic for the physical interpretation of the BVP from which the model problem derives : ST=string, BM=beam, MB=membrane and PT=plate. The second string refers to the basis used for the discretization : B=B-splines, L=linear, H1=Hermite-scaling 1, H2=Hermite-scaling 2. The third and last segment labels the choice of the pair solution-initial guess (i.e. $(\overline{u},\, u^{(0)})$) for that model problem : 1='type .1', 2='type .2', P=periodic (see Tables 1.2 & 1.3).

The construction of our two-dimensional model problems was accomplished through a Kronecker product formulation. Since this method has been widely used in the literature, we do not elaborate extensively on it but rather refer the reader to the bibliography for further details (see Schultz [1969] and Douglas & Dupont [1971] for extensive theoretical expositions. See also Bank [1978], Margenov [1981], Kaufman & Warner [1984]). The essential idea is that, for separable BVP's in rectangular parallelepipeds, the assembled element matrix can be built from linear combinations of Kronecker products of those assembled element matrices corresponding to one-dimensional problems. The latter matrices must contain the proper boundary conditions. For example, it can be shown that the assembled element matrix A for the biharmonic operator Δ^2, can be expressed as :

$$A = B \otimes M + M \otimes B + 2S \otimes S,$$

where B, S and M are the assembled bending, stiffness and mass matrices for the one-dimensional case, respectively.

The data structure used to store our operators was a form of sparse storage analogous to that described by Rivara [1984]. Our form of storage compares in efficiency to the latter when the number of non-zero entries per row of the matrix to be stored does not vary widely among rows. This is the case for the operators involved in multigrid algorithms for the solution of discretized BVP's. See Ap.IV.§2 for a full description of our sparse storage.

The following sections present the results of the application of the algorithm FAPIN to the model problems described in this section. We will show how this algorithm seems well suited for the solution of a wide variety of linear systems constructed from the finite element discretization of constant coefficient boundary value problems.

Table 1.2 : Model Problems for Non-Periodic Boundary Conditions

One-Dimensional

Notation	Boundary Value Problem to build the Matrix	Finite Element Basis	Discrete Solution and Initial Guess
ST.B.1	$-u''(x) = g(x),\ x \in I$ $u(0) = 0$ $u'(\pi) = 0$	Cubic B-splines	$\bar{u} = rand,\ u^{(0)} = 0$
ST.B.2		Cubic B-splines	$\bar{u} = 0,\ u^{(0)} = 1$
BM.B.1	$u''''(x) = g(x),\ x \in I$ $u(0) = u'(0) = 0$ $u''(\pi) = u'''(\pi) = 0$	Cubic B-splines	$\bar{u} = rand,\ u^{(0)} = 0$
BM.B.2		Cubic B-splines	$\bar{u} = 0,\ u^{(0)} = 1$

Two-Dimensional

Notation	Boundary Value Problem to build the Matrix	Finite Element Basis	Discrete Solution and Initial Guess
MB.L.1	$-\Delta u(\mathrm{x}) = g(\mathrm{x}),\ \mathrm{x} \in \Gamma$ $u(\mathrm{x}) = 0,\ \mathrm{x} \in \Gamma_1 \cup \Gamma_4$ $u_n(\mathrm{x}) = 0,\ \mathrm{x} \in \Gamma_2 \cup \Gamma_3$	Piecewise Bilinear functions	$\bar{u} = rand,\ u^{(0)} = 0$
MB.L.2			$\bar{u} = 0,\ u^{(0)} = 1$
MB.H1.1		Piecewise Bicubic Hermite Scaling 1	$\bar{u} = rand,\ u^{(0)} = 0$
MB.H1.2			$\bar{u} = 0,\ u^{(0)} = 1$
MB.H2.1		Piecewise Bicubic Hermite Scaling 2	$\bar{u} = rand,\ u^{(0)} = 0$
MB.H2.2			$\bar{u} = 0,\ u^{(0)} = 1$
PT.H1.1	$\Delta^2 u(\mathrm{x}) = g(\mathrm{x}),\ \mathrm{x} \in \Gamma$ $u(\mathrm{x}) = u_n(\mathrm{x}) = 0,\ \mathrm{x} \in \Gamma_1 \cup \Gamma_4$ $u_{nn}(\mathrm{x}) = u_{nnn}(\mathrm{x}) = 0,\ \mathrm{x} \in \Gamma_2 \cup \Gamma_3$	Piecewise Bicubic Hermite Scaling 1	$\bar{u} = rand,\ u^{(0)} = 0$
PT.H1.2			$\bar{u} = 0,\ u^{(0)} = 1$
PT.H2.1		Piecewise Bicubic Hermite Scaling 2	$\bar{u} = rand,\ u^{(0)} = 0$
PT.H2.2			$\bar{u} = 0,\ u^{(0)} = 1$

Table 1.3 : Model Problems for Periodic Boundary Conditions

One-Dimensional			
Notation	Boundary Value Problem to build the Matrix	Finite Element Basis	Discrete Solution and Initial Guess
BM.B.P	$u'''' = g(x)$, $x \in I$ PBC	Periodic Cubic B-splines	$\bar{u} = rand - s$, $u^{(0)} = 0$
Two-Dimensional			
Notation	Boundary Value Problem to build the Matrix	Finite Element Basis	Discrete Solution and Initial Guess
MB.B.P	$-\Delta u(\mathrm{x}) = g(\mathrm{x})$, $\mathrm{x} \in \Gamma$ PBC	Biperiodic Bicubic B-splines	$\bar{u} = rand - s$, $u^{(0)} = 0$
PT.B.P	$\Delta^2 u(\mathrm{x}) = g(\mathrm{x})$, $\mathrm{x} \in \Gamma$ PBC	Biperiodic Bicubic B-splines	$\bar{u} = rand - s$, $u^{(0)} = 0$

2. Experimental Measures

In this section we define the experimental measures used to analyse the performance of the algorithm FAPIN on the model problems of this thesis.

All our model are problems built from finite element discretizations of boundary value problems. In every experiment, the number of elements into which we divide the domain is an integer power of 2. Thus, we use the exponent of this power as a label for the size of the corresponding linear system. We will denote a generic exponent by k. Of course, the actual number of equations and unknowns will depend on the particular choice of finite element basis that is made and on the boundary conditions imposed. Some representative examples of these numbers are shown in the second column of Table 2.1, where $\nu(k)$ represents the number of equations and unknowns for a given value of k.

To measure the efficiency of the algorithm FAPIN, when applied to the solution of our model problems, we considered several different measures. Those which measure its convergence properties are given by the following definitions :

Definition 2.1 : *The performance in the α-norm, in the i-th iteration is the quotient*

$$P(i)_{\|\,.\,\|_\alpha} \equiv \frac{\| r^{(i)} \|_\alpha}{\| r^{(0)} \|_\alpha}, \qquad . \tag{2.1}$$

where $r^{(i)}$ is the residual vector in the i-th iteration.

Definition 2.2 : *We let N_α denote the smallest positive integer such that* :

$$\frac{\| e^{(N_\alpha)} \|_\alpha}{\| e^{(0)} \|_\alpha} \leq 10^{-5}, \tag{2.2}$$

where $e^{(i)}$ denotes the error vector in the i-th iteration.

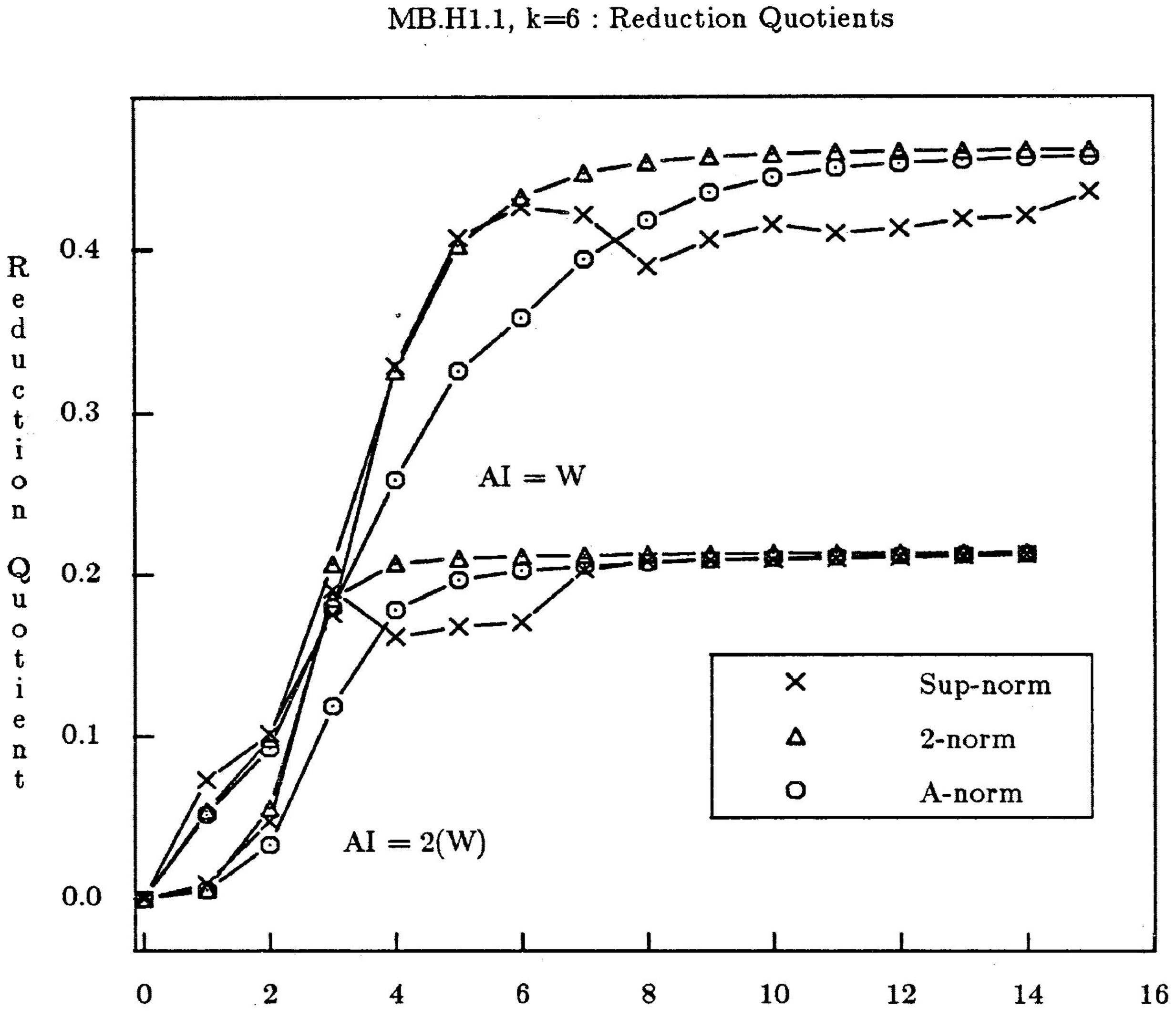
Definition 2.3 : *The reduction quotient in the α-norm in the i-th iteration is the ratio*

$$\frac{\| r^{(i)} \|_\alpha}{\| r^{(i-1)} \|_\alpha}. \tag{2.3}$$

In particular, we denote by Q_α the reduction quotient in the α-norm in the N_α-th iteration, i.e.

$$Q_\alpha \equiv \frac{\| r^{(N_\alpha)} \|_\alpha}{\| r^{(N_\alpha - 1)} \|_\alpha} \, . \tag{2.4}$$

Observe that the quantities defined above give a measure of the convergence properties of the method, but do not make any reference to the work involved (work estimates are provided below). A reasonable estimate of the asymptotic behavior of our methods can be obtained from the reduction quotient. The typical behavior is shown in figure 2.1, where we represent the reduction quotient in the infinity (uniform), 2 and A-norms (i.e. the energy norm of Ch.I.(4.6)) versus the iteration number for the model problem MB.H1.1, with finest grid corresponding to $k = 6$. Two cases are presented, corresponding to FAPIN algorithms based on the two different approximate inverses (AI's) W and 2(W) (see below for description of this notation). The cutoff value 10^{-5} in definition 2.2 should provide us with an N_α close enough to the asymptotic regime, at least in the smoother A (energy) and 2 norms.



Iteration Number
Fig. 2.1

To include the work to obtain a solution in our experimental measures, we define in (2.15) a new quantity which we call experimental effort (η). This measure will include both a work measure and a convergence measure and its value will be independent of the number of unknowns in the problem if the algorithm being studied exhibits multigrid behavior.

We define the complexity (C) of the algorithm FAPIN when applied to a model problem, as the number of arithmetic operations performed in one iteration on the corresponding model problem. We neglect the work due to additions as compared to that in multiplications. We also neglect the work involved in setting up the experiments.

Following Varga [1974, p.62], we define the following quantities :

Definition 2.4 : *Let $M \in \mathbf{M}_n(\mathbf{R})$. If for some $m \in \mathbf{Z}^+$, $\| M^m \|_2 < 1$, then :*

$$R(M^m) \equiv -\ln\left(\| M^m \|_2{}^{\frac{1}{m}}\right) \tag{2.5}$$

is the average rate of convergence for m iterations of the matrix M.

Definition 2.5 : *We let $\sigma(m)$ denote the average reduction factor per iteration, where*

$$\sigma(m) \equiv \left(\frac{\| e^{(m)} \|_2}{\| e^{(0)} \|_2}\right)^{\frac{1}{m}}. \tag{2.6}$$

Observe that from the previous definitions we have (see Varga [1974]) :

$$\sigma(m) \leq \| M^m \|_2^{\frac{1}{m}} = e^{-R(M^m)}. \tag{2.7}$$

If we define $\mu(m) \equiv (R(M^m))^{-1}$ then from (2.7) :

$$\sigma(m)^{\mu(m)} \leq \frac{1}{e}. \tag{2.8}$$

Thus $\mu(m)$ is a measure of the number of iterations needed to reduce the 2-norm of the initial error by a factor of e. But from (2.8) we have :

$$\mu(m) \leq -\frac{1}{\ln(\sigma(m))} \tag{2.10}$$

and thus the quantity $-\dfrac{1}{\ln(\sigma(m))}$ is an upper bound to the estimate $\mu(m)$.

In this way we arrive at our definition of effort :

Definition 2.6 : *We define the effort in the application of m iterations of an iterative algorithm to the solution of a linear system of $\nu(k)$ equations as :*

$$E \equiv -\frac{C}{\nu(k)\ln\sigma(m)}, \tag{2.11}$$

where C and $\nu(k)$ are the corresponding complexity and number of unknowns, respectively.

We note that, asymptotically, this definition is equivalent to that given by Benson & Frederickson [1982, p.130]. When estimating the complexities we neglect boundary effects and the smaller band-widths of the operators in the coarser grids. Thus we let b_A, b_Z, b_C and b_I denote the maximum of the number of non-zero entries per row in the corresponding matrix representation of the differential operator, the approximate inverse, the collection operator and the interpolation operator, respectively, for the finest grid. Also, let k_m be the index of the finest grid considered (i.e. 2^{k_m} and 2^{2k_m} is the number of elements in the finest grid, in the one and two-dimensional case, respectively). Then, recalling the form of the algorithm FAPIN (cf. Ch.I.§5) and noting that $\nu(k-1) \approx \frac{\nu(k)}{2^d}$, where d is the dimension of the model problem (cf. §2.1), the number of multiplications in a given grid k has the following breakdown :

$$\begin{array}{lll} \textit{Collect} & \longrightarrow & b_C\nu(k-1) \approx b_C\,\dfrac{\nu(k)}{2^d} \\ \textit{Apply } A & \longrightarrow & b_A\;\nu(k) \\ \textit{Apply } Z & \longrightarrow & b_Z\;\nu(k) \\ \textit{Interpolate} & \longrightarrow & b_I\;\nu(k+1) \approx b_I\;2^d\nu(k) \end{array} \tag{2.12}$$

Thus, If we let $\beta = \frac{b_C}{2^d} + b_A + b_Z + 2^d b_I$, from (2.12), an estimate of the complexity (actually, an upper bound) is given by :

$$C(k_m) \approx \left(\sum_{k=0}^{k_m} \beta\nu(k)\right) - 2^d b_I \nu(k_m) \ , \tag{2.13}$$

where the extra interpolation term $2^d b_I\nu(k_m)$ counted in the summation when $k = k_m$ is removed by subtraction. We are also neglecting the effect of adding a term for a collection and an application of A in the coarsest grid. Performing the operations expressed in (2.13) we obtain the desired complexities. These are shown in Table 2.1, which contains all the distinct cases of this thesis.

Table 2.1 : Number of Equations and Complexities

Problem	$\nu(k)$	$C(k_m) \approx$
ST.B.1	2^k+2	$(\beta-b_I)2^{k_m+1}+2\beta k_m+\beta-4b_I$
BM.B.1	2^k+1	$(\beta-b_I)2^{k_m+1}+\beta k_m+\beta-2b_I$
BM.B.P	2^k	$(\beta-b_I)2^{k_m+1}-\beta$
MB.L.1	$(2^k)^2$	$(\frac{\beta}{3}-b_I)4^{k_m+1}-\frac{1}{3}\beta$
MB.B.P		
PT.B.P		
MB.H1.1	$(2(2^k+1)-1)^2$	$(\frac{4}{3}\beta-4b_I)4^{k_m+1}+(8\beta-16b_I)2^{k_m}+2k_m-\frac{13}{3}\beta-4b_I$
PT.H1.1	$(2(2^k+1)-2)^2$	$(\frac{4}{3}\beta-4b_I)4^{k_m+1}-\frac{4}{3}\beta$

From Table 2.1 we observe that, in every case, the approximate number of multiplications per unknown is :

$$\frac{C(k_m)}{\nu(k_m)} \approx c_1\beta - c_2b_I\ . \tag{2.14}$$

Equation (2.14) is the expression we will use to estimate the effort and serves, together with (2.11), as the basis for the definition of our last experimental measure :

Definition 2.7 : *We let η denote the experimental effort defined by :*

$$\eta \equiv -\frac{c_1\beta - c_2b_I}{\ln\sigma(N_2)} \tag{2.15}$$

where N_2 is defined in (2.2).

Representative examples of the estimated number of multiplications per unknown, that cover all the cases that appear in this thesis, are given in Table 2.2. Observe that, since the complexity for a given problem depends on the sparsity pattern of the approximate inverse used, these are indicated in the table under the entry 'AI', together with the constants β and b_I used to evaluate the right member of (2.14).

The sparsity patterns of the LSQ-approximate inverses which the lables I,F,W, etc. represent are explained more extensively in the results sections below. In brief, ID means 'idem', that is, the same sparsity pattern as the corresponding discrete differential operator. F and W stand for 'fill' and 'wide', respectively. F fills the zero entries contained in the stripes (bands in the one dimensional case) of the corresponding discrete differential operator. W applies only to our two-dimensional second order non-periodic model problems. It consists of a Kronecker product $P{\otimes}P$, where P is a nine-diagonal matrix of the appropriate size, with nine by nine blocks of non-zero entries at the left hand upper corner and the right hand lower corner.

The notation 2(P), where P can be ID, F, or W, indicates that the algorithm uses two smoothing steps with the correponding LSQ-approximate inverse of the indicated pattern. We recall that two iterations of a linear stationary iterative method with an approximate inverse Z are equivalent to a single iteration with $Z' = 2Z - ZAZ$ (see Ch.I.§2). We also recall that Z' is an approximate inverse whenever Z is (in particular, when Z is an LSQ-approximate inverse); thus the use of 2(P) is justified.

Table 2.2 : Estimated Multiplications per Unknown

Problem	AI	β	b_I	$(c_1\beta - c_2b_I)$
ST.B.1	ID	$\frac{45}{2}$	3	39
BM.B.1	ID			
BM.B.P	ID			
MB.L.1	ID	$\frac{145}{4}$	4	$\frac{97}{3}$
MB.H1.1	F	$\frac{525}{4}$	16	444
	2(F)	$\frac{769}{4}$	16	$\frac{2308}{3}$
	W	$\frac{705}{4}$	16	684
	2(W)	$\frac{1129}{4}$	16	$\frac{3748}{3}$
PT.H1.1	F	$\frac{525}{4}$	16	444
	2(F)	$\frac{769}{4}$	16	$\frac{2308}{3}$
MB.B.1	ID	$\frac{561}{4}$	9	151
	2(ID)	$\frac{1737}{4}$	9	534

3. One-Dimensional Model Problems

In this section we present the experimental results for our one-dimensional model problems. These are given in the form of tables. In every case, k_m indicates the index of the finest grid used in the corresponding experiment. Since we are dealing with one dimensional problems, it also indicates the power of 2 that gives the number of finite elements in the finest grid considered. Comments on the results are given at the end of the section.

The results for the non-periodic model problems ST.B.1, ST.B.2, BM.B.1 and BM.B.2 are given in Tables 3.1.a, 3.1.b, 3.2.a and 3.2.b, respectively. For these problems, the LSQ-approximate inverses considered always had the sparsity pattern 'fill' (denoted by F). Some sparsity patterns for the discrete differential operators for the BM.B.1 and BM.B.2 model problems are shown, for the grids $k = 3,2,1$, in Figure 3.1. The corresponding LSQ-approximate inverse patterns are obtained by 'filling in' the null entries contained in the bands. The coarsest grid used in these experiments corresponded to the index $k = 0$. We recall that the number of unknowns for the ST.B.1 (and ST.B.2) and BM.B.1 (and BM.B.2) experiments are $2^{k_m} + 2$ and $2^{k_m} + 1$, respectively (see Table 2.1).

```
****-----        ****-        ***
***-*----        ***-*        ***
****-*---        ****-        ***
*-***-*--        *-***
-*-***-*-        -*-**
--*-***-*
---*-***-
----*-***
-----*-**
```

Fig. 3.1 : Sparsity Patterns for BM.B.1&2

The results for the periodic model problem BM.B.P are given in Table 3.3 . For this problem, the LSQ-approximate pseudo-inverse patterns were ID (i.e. that of the corresponding differential operator), except in the coarsest grid $k = 2$, where the pattern of a four by four tridiagonal circulant matrix was used (since in this case the corresponding discrete operator is a singular full matrix, and thus the LSQ-approximate inverse of this pattern corresponds to the exact inverse, which is not defined). Some examples of these patterns are shown in Figure 3.2, for the grid indices $k = 4$, 3 and 2.

```
**-*---------*-*      **-*-*-*      **-*
***-*---------*-      ***-*-*-      ***-
-***-*---------*      -***-*-*      -***
*-***-*---------      *-***-*-      *-**
-*-***-*--------      -*-***-*
--*-***-*-------      *-*-***-
---*-***-*------      -*-*-***
----*-***-*-----      *-*-*-**
-----*-***-*----
------*-***-*---
-------*-***-*--
--------*-***-*-
---------*-***-*
*---------*-***-
-*---------*-***
*-*---------*-**
```

Fig. 3.2 : LSQ Patterns for BM.B.P

We recall that the matrix representations of the differential operators for BM.B.P in the different grids are all circulant matrices based on the molecule (see Ap.II.(3.7)) :

$$b \equiv \frac{1}{8h^3}\left\{3,\ 0,\ -27,\ 48,\ -27,\ 0,\ 3\right\},$$

where h is the mesh-size. Therefore, the corresponding LSQ-approximate pseudo-inverses for $k \geq 4$ will be circulant matrices based on a molecule of the form :

$$l \equiv 8h^3\left\{l_1,\ 0,\ l_2,\ l_3,\ l_4,\ 0,\ l_5\right\}.$$

Thus, since the mesh size (h) dependency in the above molecules reduces to a multiplicative factor, only three approximate inverses, with no h dependency, need be calculated (i.e. $k = 2,3,4$). Hence, for BM.B.P, we did not store the operators in the usual sparse form (see Ap.IV.§2), but rather used the corresponding molecules. The HL package (see Ap.IV) provides a special purpose operator that calculates the vector resulting from the application of a given molecule to a given vector in a circulant manner. The collection and interpolation operators were handled by a block extension of the same operator in HL. We recall that the number of unknowns for the BM.B.P experiments is 2^{k_m} (see Table 2.1).

Table 3.1.a : Results for ST.B.1

k_m	$P(3)\|.\|_A$	$P(8)\|.\|_A$	$P(3)\|.\|_2$	$P(8)\|.\|_2$	N_∞	N_2	Q_∞	Q_2	η
3	3.79e-04	9.36e-09	4.15e-04	1.06e-08	5	5	8.96e-02	9.79e-02	18.1
4	1.00e-03	1.90e-08	1.01e-03	1.82e-08	5	5	1.31e-01	1.18e-01	17.4
5	1.06e-03	3.58e-08	1.09e-03	3.77e-08	5	5	2.39e-01	1.39e-01	19.2
6	9.47e-04	3.45e-08	9.71e-04	3.72e-08	5	5	1.34e-01	1.29e-01	17
7	9.97e-04	3.80e-08	1.02e-03	3.93e-08	5	5	1.35e-01	1.30e-01	18.1
8	1.10e-03	5.00e-08	1.10e-03	4.91e-08	6	6	1.43e-01	1.36e-01	20.6
9	1.17e-03	5.92e-08	1.15e-03	5.69e-08	5	5	1.32e-01	1.36e-01	18.3
10	1.24e-03	6.34e-08	1.22e-03	6.07e-08	6	5	1.40e-01	1.36e-01	20.7

Table 3.1.b : Results for ST.B.2

k_m	$P(3)\|.\|_A$	$P(8)\|.\|_A$	$P(3)\|.\|_2$	$P(8)\|.\|_2$	N_∞	N_2	Q_∞	Q_2	η
3	3.47e-04	8.58e-09	3.62e-04	8.93e-09	5	4	2.13e-01	8.60e-02	15.8
4	7.27e-04	1.25e-08	6.33e-04	1.12e-08	4	4	8.76e-02	1.15e-01	15
5	7.38e-04	2.62e-08	6.35e-04	2.20e-08	4	4	9.34e-02	1.18e-01	14.2
6	7.40e-04	2.64e-08	6.37e-04	2.21e-08	4	4	9.33e-02	1.18e-01	13.8
7	7.40e-04	2.64e-08	6.37e-04	2.21e-08	4	4	9.33e-02	1.18e-01	13.5
8	7.41e-04	2.64e-08	6.37e-04	2.21e-08	4	4	9.33e-02	1.18e-01	13.4
9	7.41e-04	2.64e-08	6.37e-04	2.21e-08	4	4	9.33e-02	1.18e-01	13.4
10	7.41e-04	2.64e-08	6.37e-04	2.21e-08	4	4	9.33e-02	1.18e-01	14.1

Table 3.2.a : Results for BM.B.1

k_m	$P(3)\| . \|_A$	$P(8)\| . \|_A$	$P(3)\| . \|_2$	$P(8)\| . \|_2$	N_∞	N_2	Q_∞	Q_2	η
3	2.62e-04	3.77e-10	2.71e-04	4.25e-10	4	4	4.34e-02	5.60e-02	14.5
4	4.35e-04	3.02e-09	4.45e-04	2.97e-09	5	5	9.46e-02	9.58e-02	15.9
5	8.24e-04	2.32e-08	8.25e-04	2.30e-08	5	5	1.32e-01	1.23e-01	18.7
6	9.73e-04	3.26e-08	9.60e-04	3.18e-08	6	5	1.34e-01	1.27e-01	20
7	9.36e-04	3.00e-08	9.16e-04	2.89e-08	6	6	1.36e-01	1.27e-01	21.1
8	9.72e-04	3.19e-08	9.50e-04	3.07e-08	6	6	1.28e-01	1.28e-01	22.2
9	1.12e-03	3.83e-08	1.09e-03	3.67e-08	6	6	1.18e-01	1.29e-01	23.6
10	1.18e-03	4.12e-08	1.15e-03	3.95e-08	7	7	1.25e-01	1.30e-01	26.2

Table 3.2.b : Results for BM.B.2

k_m	$P(3)\| . \|_A$	$P(8)\| . \|_A$	$P(3)\| . \|_2$	$P(8)\| . \|_2$	N_∞	N_2	Q_∞	Q_2	η
3	9.72e-05	3.21e-10	1.25e-04	3.61e-10	4	4	1.34e-01	1.16e-01	15.3
4	3.04e-04	2.22e-09	3.33e-04	2.36e-09	5	4	8.23e-02	9.03e-02	15.9
5	3.53e-04	8.02e-09	3.83e-04	8.63e-09	5	4	8.56e-02	1.08e-01	15.6
6	3.54e-04	8.25e-09	3.83e-04	8.87e-09	4	4	1.15e-01	1.09e-01	15.6
7	3.54e-04	8.25e-09	3.83e-04	8.87e-09	4	4	1.16e-01	1.09e-01	14.7
8	3.54e-04	8.25e-09	3.83e-04	8.87e-09	4	4	1.16e-01	1.09e-01	14.5
9	3.54e-04	8.25e-09	3.83e-04	8.87e-09	4	4	1.16e-01	1.09e-01	14.3
10	3.54e-04	8.25e-09	3.83e-04	8.87e-09	4	4	1.16e-01	1.09e-01	13.8

Table 3.3 : Results for BM.B.P

k_m	$P(3)\| . \|_A$	$P(8)\| . \|_A$	$P(3)\| . \|_2$	$P(8)\| . \|_2$	N_∞	N_2	Q_∞	Q_2	η
3	2.32e-03	4.65e-07	3.02e-03	6.04e-07	6	6	1.81e-01	1.82e-01	22.2
4	2.60e-03	3.34e-06	3.38e-03	4.47e-06	6	6	2.56e-01	2.67e-01	25.2
5	2.18e-03	2.00e-06	2.68e-03	2.70e-06	6	6	2.25e-01	2.56e-01	24
6	1.78e-03	1.03e-06	2.05e-03	1.34e-06	6	6	2.17e-01	2.37e-01	23.8
7	2.39e-03	1.68e-06	2.86e-03	2.16e-06	7	7	2.46e-01	2.44e-01	27.3
8	2.75e-03	2.55e-06	3.38e-03	3.35e-06	7	7	2.54e-01	2.57e-01	28.1
9	2.71e-03	2.42e-06	3.28e-03	3.18e-06	8	8	2.48e-01	2.60e-01	30.8
10	3.05e-03	2.69e-06	3.68e-03	3.53e-06	8	8	2.51e-01	2.59e-01	30.6

The tables are consistent with the independence of the experimental measures on the size (i.e. k_m) and the type of the model problem solved (ST.B.1 vs. ST.B.2 and BM.B.1 vs. BM.B.2). This is the expected behavior of a multigrid algorithm.

4. Two-Dimensional Second Order Model Problems

In this section we present the experimental results for our two-dimensional second order problems. We commence with the non-periodic ones (Tables 4.1.a, 4.1.b, 4.2.a, 4.2.b, 4.3.a and 4.3.b) and summarize our results in Table 4.4. Finally we give the results for the periodic case in Table 4.5. Comments on each set of results are given after the corresponding set of tables.

Tables 4.1.a and 4.1.b show the results for MB.L.1 and MB.L.2, respectively. We recall that the finite element basis used in these cases consisted of bilinear 'pagoda' functions, built through Kronecker products of one-dimensional linear 'roof' functions. In every run, the coarsest grid was that corresponding to $k = 1$. The approximate inverses were, in every grid, LSQ-approximate inverses of the ID pattern (in this case, that of a Kronecker product of tridiagonal matrices). We recall that the number of unknowns for these experiments is $(2^{k_m})^2$ (see Table 2.1).

Table 4.1.a : Results for MB.L.1

k_m	$P(3)\|\ .\ \|_A$	$P(8)\|\ .\ \|_A$	$P(3)\|\ .\ \|_2$	$P(8)\|\ .\ \|_2$	N_∞	N_2	Q_∞	Q_2	η
3	1.62e-03	1.50e-07	1.53e-03	1.37e-07	5	5	1.45e-01	1.56e-01	16.9
4	1.46e-03	1.64e-07	1.35e-03	1.47e-07	5	5	1.68e-01	1.60e-01	17.2
5	1.30e-03	1.27e-07	1.22e-03	1.15e-07	6	6	1.64e-01	1.58e-01	18.2
6	1.20e-03	1.21e-07	1.14e-03	1.10e-07	6	6	1.69e-01	1.59e-01	18.3
7	1.23e-03	1.23e-07	1.16e-03	1.12e-07	6	6	1.54e-01	1.59e-01	18.3

Table 4.1.b : Results for MB.L.2

k_m	$P(3)\|\,.\,\|_A$	$P(8)\|\,.\,\|_A$	$P(3)\|\,.\,\|_2$	$P(8)\|\,.\,\|_2$	N_∞	N_2	Q_∞	Q_2	η
3	3.40e-03	3.29e-07	2.67e-03	2.47e-07	5	5	1.63e-01	1.59e-01	16.2
4	3.78e-03	5.38e-07	3.01e-03	4.01e-07	6	5	1.55e-01	1.70e-01	17.3
5	3.96e-03	7.15e-07	3.21e-03	5.34e-07	6	6	1.58e-01	1.76e-01	18
6	4.06e-03	8.22e-07	3.32e-03	6.15e-07	6	6	1.59e-01	1.81e-01	18.4
7	4.11e-03	8.77e-07	3.37e-03	6.56e-07	6	6	1.59e-01	1.82e-01	18.6

From the above tables we observe that the algorithm FAPIN shows a clear multigrid behavior, since the experimental results are essentially independent of the grid size.

Tables 4.2.a and 4.2.b present the results for MB.H1.1 and MB.H1.2, respectively. The finite element basis used in these cases consisted of piecewise bicubic Hermite functions built through Kronecker products of one-dimensional piecewise cubic Hermite functions, scaled according to scaling 1, described in Ap.I.§3. The coarsest grid was always that corresponding to $k = 0$. For these problems we tried four different approximate inverses (F, 2(F), W and 2(W)), based on LSQ-approximate inverses of two different sparsity patterns : F and W. One-dimensional counterparts (recall the Kronecker product structure of the 2D operators) of the patterns are shown in Figure 4.1 for the case $k = 3$.

```
***--------------      *********--------
*****------------      *********--------
*****------------      *********--------
-******----------      *********--------
-******----------      *********--------
---******--------      -*********-------
---******--------      --*********------
-----******------      ---*********-----
-----******------      ----*********----
-------******----      -----*********---
-------******----      ------*********--
---------******--      -------*********-
---------******--      --------*********
-----------******      --------*********
-----------******      --------*********
-------------****      --------*********
-------------****      --------*********
```

Fig. 4.1 : F and W LSQ Patterns for 1D Hermite Bases

The above approximate inverses were used in all grids except those corresponding to $k = 1{,}0$, for which the LSQ-approximate inverse of pattern ID was used (in this case, an exact inverse was calculated). In Figure 4.2 we depict the sparsity patterns for the one dimensional counterparts of the discrete differential operators, for the grids $k = 3{,}2{,}1{,}0$. We recall that the umber of unknowns in these experiments is $(2(2^{k_m}+1)—1)^2$ (see Table 2.1).

```
***--------------      ***------      ***--      ***
**-**------------      **-**----      **-**      ***
*-***------------      *-***----      *-***      ***
-***-**----------      -***-**--      -****
-**-***----------      -**-***--      -****
---***-**--------      ---***-**
---**-***--------      ---**-***
-----***-**------      -----****
-----**-***------      -----****
-------***-**----
-------**-***----
---------***-**--
---------**-***--
-----------***-**
-----------**-***
-------------****
-------------****
```

Fig. 4.2 : Differential Operator Patterns for 1D Hermite Bases

Table 4.2.a : Results for MB.H1.1

AI	k_m	$P(3)\|\cdot\|_A$	$P(8)\|\cdot\|_A$	$P(3)\|\cdot\|_2$	$P(8)\|\cdot\|_2$	N_∞	N_2	Q_∞	Q_2	η
F	3	3.54e-03	1.99e-04	3.86e-03	2.30e-04	>15	>15	6.02e-01	6.06e-01	841
	4	3.63e-03	9.07e-05	3.73e-03	1.16e-04	>15	>15	6.13e-01	6.22e-01	895
	5	2.84e-03	4.11e-05	2.93e-03	6.21e-05	>15	>15	6.15e-01	6.30e-01	915
	6	3.70e-03	2.27e-05	3.75e-03	4.40e-05	>15	>15	5.95e-01	6.27e-01	882
2(F)	3	5.97e-04	3.17e-06	6.84e-04	3.67e-06	10	9	3.56e-01	3.59e-01	710
	4	2.70e-04	1.59e-06	3.38e-04	2.13e-06	11	10	3.70e-01	3.80e-01	762
	5	1.24e-04	6.97e-07	1.78e-04	1.25e-06	11	10	3.72e-01	3.91e-01	779
	6	7.05e-05	3.82e-07	1.22e-04	9.91e-07	10	10	3.66e-01	3.99e-01	787
W	3	7.16e-04	6.15e-06	9.01e-04	8.34e-06	10	9	3.68e-01	3.79e-01	628
	4	7.64e-04	5.40e-06	1.00e-03	1.06e-05	12	11	4.21e-01	4.38e-01	762
	5	7.24e-04	4.31e-06	9.77e-04	1.14e-05	12	11	4.19e-01	4.54e-01	798
	6	8.65e-04	4.29e-06	1.11e-03	1.27e-05	12	11	4.12e-01	4.59e-01	812
2(W)	3	4.36e-05	2.26e-09	5.96e-05	3.09e-09	5	5	1.37e-01	1.41e-01	578
	4	2.98e-05	6.32e-09	5.73e-05	1.51e-08	6	6	1.77e-01	1.92e-01	702
	5	2.16e-05	5.98e-09	5.72e-05	2.13e-08	6	6	1.75e-01	2.06e-01	734
	6	2.09e-05	6.27e-09	6.18e-05	2.55e-08	6	6	1.70e-01	2.11e-01	747

Table 4.2.b : Results for MB.H1.2

AI	k_m	$P(3)\|\cdot\|_A$	$P(8)\|\cdot\|_A$	$P(3)\|\cdot\|_2$	$P(8)\|\cdot\|_2$	N_∞	N_2	Q_∞	Q_2	η
F	3	1.11e-02	7.84e-04	9.47e-03	6.72e-04	>15	>15	6.05e-01	6.08e-01	858
	4	7.80e-03	5.46e-04	6.72e-03	4.73e-04	>15	>15	6.14e-01	6.23e-01	907
	5	5.60e-03	3.82e-04	5.15e-03	3.45e-04	>15	>15	6.14e-01	6.30e-01	927
	6	4.18e-03	2.65e-04	4.48e-03	2.70e-04	>15	>15	5.99e-01	6.27e-01	894
2(F)	3	2.28e-03	1.34e-05	1.94e-03	1.16e-05	11	9	3.65e-01	3.65e-01	730
	4	1.59e-03	9.88e-06	1.35e-03	8.99e-06	11	10	3.73e-01	3.81e-01	774
	5	1.11e-03	6.94e-06	9.56e-04	7.20e-06	11	10	3.73e-01	3.91e-01	791
	6	7.69e-04	4.89e-06	7.20e-04	6.30e-06	11	10	3.73e-01	4.00e-01	799
W	3	2.83e-03	2.84e-05	2.92e-03	2.84e-05	11	9	3.69e-01	3.76e-01	643
	4	3.10e-03	3.43e-05	3.41e-03	4.60e-05	13	11	4.24e-01	4.39e-01	777
	5	3.30e-03	4.08e-05	3.94e-03	6.41e-05	13	11	4.24e-01	4.54e-01	811
	6	3.82e-03	5.08e-05	4.50e-03	7.95e-05	13	12	4.24e-01	4.60e-01	830
2(W)	3	2.02e-04	1.01e-08	2.05e-04	1.02e-08	6	5	1.35e-01	1.39e-01	590
	4	1.89e-04	4.11e-08	2.50e-04	6.71e-08	7	6	1.82e-01	1.93e-01	715
	5	2.00e-04	5.75e-08	3.20e-04	1.20e-07	7	6	1.82e-01	2.06e-01	746
	6	2.28e-04	7.97e-08	3.83e-04	1.61e-07	7	6	1.82e-01	2.11e-01	758

The tables above again show the multigrid behavior of FAPIN. We note, however, that since the values of N_∞ and N_2 were not available, for the problems using F, the presented values of Q_∞, Q_2 and η correspond to the last iteration performed (the 15-th) and thus the given values of η are underestimated. We also observe that, for the choices of approximate inverses that yield good convergence (2(F), W and 2(W)), the experimental efforts (η) are all very similar. Thus the 'preferred' (at least for a non-parallel implementation) algorithm should probably be that based on 2(F), because of its comparatively smaller storage requirements. The following results show how a proper scaling of the basis can provide us with a significantly better algorithm.

In Tables 4.3.a and 4.3.b we give the results for MB.H2.1 and MB.H2.2, respectively. These problems differ from MB.H1.1 and MB.H1.2 in that the basis functions were scaled according to scaling 2, described in Ap.I.§4. The 'type .1' problems also differ from the previous ones in their corresponding right hand sides, since these are built by application of the discrete differential operator for the finest grid. The number of unknowns in these experiments are the same as for MB.H1.1 and MB.H1.2.

Table 4.3.a : Results for MB.H2.1

AI	k_m	$P(3)\|\,.\,\|_A$	$P(8)\|\,.\,\|_A$	$P(3)\|\,.\,\|_2$	$P(8)\|\,.\,\|_2$	N_∞	N_2	Q_∞	Q_2	η
F	3	4.52e-03	2.17e-05	7.03e-03	3.36e-05	10	9	3.56e-01	3.46e-01	398
	4	3.71e-03	1.84e-05	6.73e-03	3.33e-05	10	9	3.63e-01	3.48e-01	393
	5	3.71e-03	1.81e-05	6.97e-03	3.52e-05	10	9	3.61e-01	3.49e-01	391
	6	3.75e-03	1.75e-05	7.10e-03	3.57e-05	10	8	3.61e-01	3.49e-01	385
2(F)	3	1.85e-04	4.96e-09	2.87e-04	7.34e-09	5	5	1.28e-01	1.20e-01	348
	4	1.51e-04	4.17e-09	2.77e-04	7.26e-09	5	5	1.32e-01	1.21e-01	343
	5	1.48e-04	4.05e-09	2.90e-04	7.73e-09	5	5	1.31e-01	1.22e-01	341
	6	1.43e-04	3.90e-09	2.94e-04	7.87e-09	5	4	1.31e-01	1.22e-01	334

Table 4.3.b : Results for MB.H2.2

AI	k_m	$P(3) \| . \|_A$	$P(8) \| . \|_A$	$P(3) \| . \|_2$	$P(8) \| . \|_2$	N_∞	N_2	Q_∞	Q_2	η
F	3	8.32e-03	4.45e-05	1.06e-02	5.26e-05	11	9	3.57e-01	3.47e-01	404
	4	7.60e-03	4.09e-05	1.07e-02	5.43e-05	11	9	3.60e-01	3.48e-01	399
	5	7.20e-03	3.83e-05	1.07e-02	5.52e-05	11	9	3.60e-01	3.49e-01	396
	6	6.98e-03	3.68e-05	1.08e-02	5.56e-05	11	9	3.60e-01	3.49e-01	395
2(F)	3	3.73e-04	1.03e-08	4.46e-04	1.15e-08	6	5	1.28e-01	1.21e-01	353
	4	3.35e-04	9.33e-09	4.51e-04	1.19e-08	6	5	1.30e-01	1.21e-01	348
	5	3.12e-04	8.59e-09	4.55e-04	1.21e-08	6	5	1.30e-01	1.22e-01	346
	6	3.00e-04	8.17e-09	4.58e-04	1.22e-08	6	5	1.30e-01	1.22e-01	344

The above tables again show the multigrid behavior of the algorithm FAPIN. We note however the improved performance when based on the newly scaled basis. Observe that the experimental effort in Tables 4.3.a and 4.3.b have been approximately reduced by half with respect to those in Tables 4.2.a and 4.2.b. A graphical representation of this behavior is depicted in Figure 4.3. In it we plot the logarithm of the uniform norm (i.e. $\| . \|_\infty$) of the error versus the iteration number for the model problems MB.H1.2 and MB.H2.2 ($k_m = 6$) for the different approximate inverses used. We recall that MB.H1.2 and MB.H2.2 represent two different discretizations (differing in the scaling of the finite element basis chosen) of the same boundary value problem.

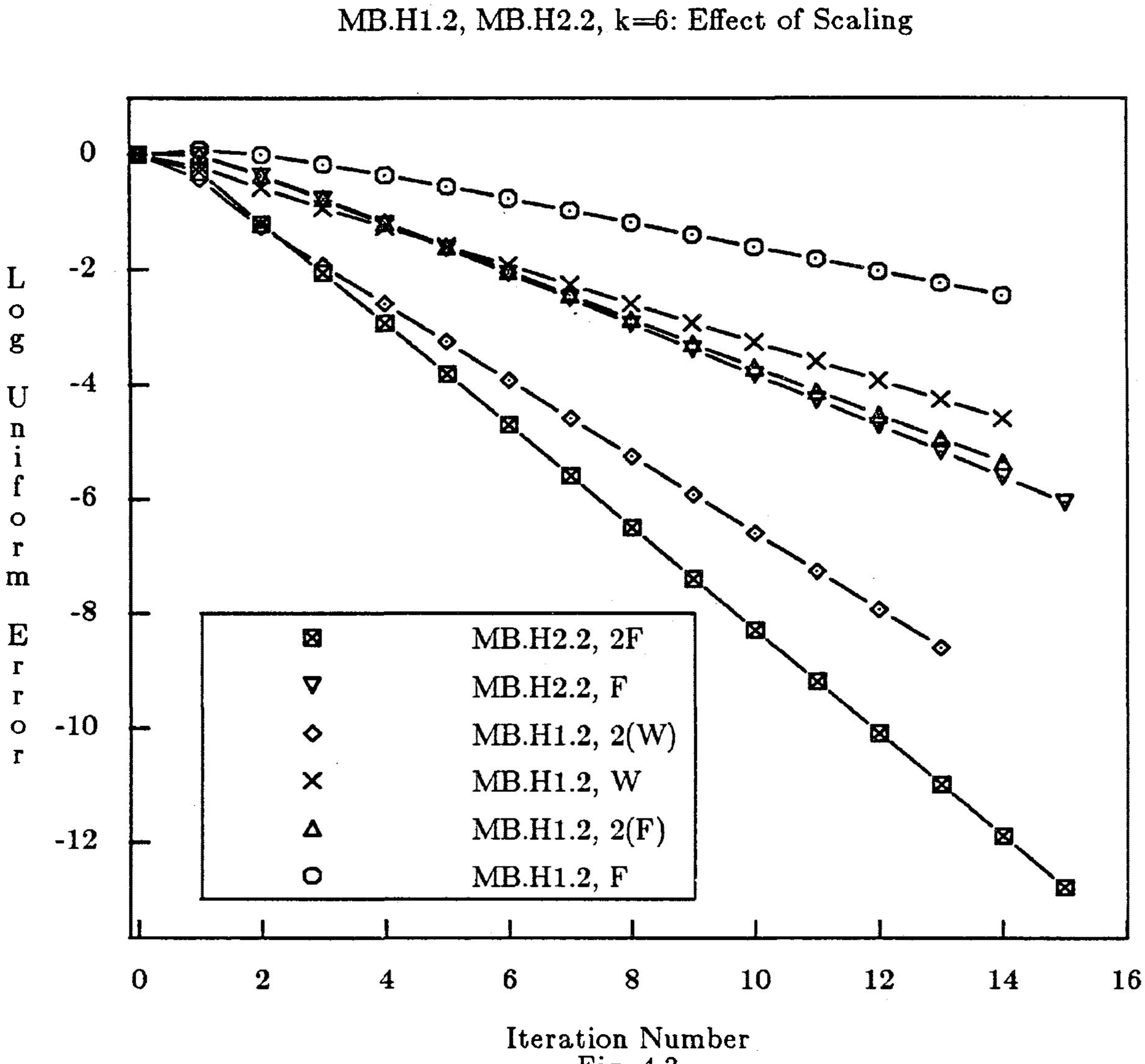


Fig. 4.3

The results given thus far correspond to the non-periodic cases and are summarized in Table 4.4, where the experimental measures are all given for the indicated model problem which has a finest grid corresponding to $k_m = 6$.

Table 4.4 : Non-Periodic Membrane Experiments Summary

Problem	AI	$P(8)\|\ .\ \|_A$	$P(8)\|\ .\ \|_2$	$P(8)\|\ .\ \|_\infty$	N_A	N_2	N_∞	Q_A	Q_2	Q_∞	η
MB.L.1	ID	1.21e-07	1.10e-07	1.94e-07	6	6	6	1.60e-01	1.59e-01	1.69e-01	18.3
MB.L.2	ID	8.22e-07	6.15e-07	2.34e-07	6	6	6	1.84e-01	1.81e-01	1.59e-01	18.4
MB.H1.1	F	2.27e-05	4.40e-05	3.84e-04	>15	>15	>15	6.06e-01	6.27e-01	5.95e-01	882
	2(F)	3.82e-07	9.91e-07	6.05e-06	10	10	10	3.76e-01	3.99e-01	3.66e-01	787
	W	4.29e-06	1.27e-05	1.22e-05	12	11	12	4.52e-01	4.59e-01	4.12e-01	812
	2(W)	6.27e-09	2.55e-08	1.60e-08	6	6	6	2.02e-01	2.11e-01	1.70e-01	747
MB.H1.2	F	2.65e-04	2.70e-04	8.81e-04	>15	>15	>15	6.07e-01	6.27e-01	5.99e-01	894
	2(F)	4.89e-06	6.30e-06	1.60e-05	11	10	11	3.81e-01	4.00e-01	3.73e-01	799
	W	5.08e-05	7.95e-05	3.03e-05	13	12	13	4.55e-01	4.60e-01	4.24e-01	830
	2(W)	7.97e-08	1.61e-07	3.89e-08	7	6	7	2.05e-01	2.11e-01	1.82e-01	758
MB.H2.1	F	1.75e-05	3.57e-05	3.67e-05	10	8	10	3.50e-01	3.49e-01	3.61e-01	385
	2(F)	3.90e-09	7.87e-09	1.08e-08	5	4	5	1.22e-01	1.22e-01	1.31e-01	334
MB.H2.2	F	3.68e-05	5.56e-05	1.46e-04	11	9	11	3.49e-01	3.49e-01	3.60e-01	395
	2(F)	8.17e-09	1.22e-08	4.23e-08	6	5	6	1.22e-01	1.22e-01	1.30e-01	344

In view of Table 4.4 we remark that, since the values of N_α, $\alpha = A,2,\infty$, for MB.H1.1 and MB.H1.2 with the F pattern for the LSQ-approximate inverse are not available, the values of η for these problems are underestimated. We also note that the bilinear elements are clearly the most effective, in terms of η. Nevertheless, the Hermite basis with scaling 2 performs reasonably well, in terms of N_α, though at a much higher cost in terms of work, than the former basis.

Next we give the results for our two-dimensional second order periodic model problem. Table 4.5 presents the results for MB.B.P. The finite element space used in this case consisted of biperiodic bicubic splines built through Kronecker products of one-dimensional periodic cubic splines. The coarsest grid considered was, in all runs, that corresponding to $k = 2$. Thus the case $k_m = 3$ is actually a two-grid algorithm. Two kinds of approximate pseudo-inverses were used in these experiments : LSQ-approximate pseudo-inverses (see Ch.I.D3.4) of the ID sparsity pattern and the corresponding 2(ID) (that is, two succesive iterations with a linear stationary method based on the corresponding LSQ-approximate pseudo-inverse). These approximate pseudo-inverses were used in all grids except in the coarsest one, were a 15-diagonal matrix was used (since the discrete differential operator is a singular full matrix). Some examples of the one-dimensional counterparts for these sparsity patterns (except for $k = 2$) are shown in Figure 3.2 of the previous section. We recall that the number of unknows in these experiments is $(2^{k_m})^2$.

Table 4.5 : Results for MB.B.P

AI	k_m	$P(3)\| \,.\, \|_A$	$P(8)\| \,.\, \|_A$	$P(3)\| \,.\, \|_2$	$P(8)\| \,.\, \|_2$	N_∞	N_2	Q_∞	Q_2	η
ID	3	7.25e-04	3.07e-07	1.18e-03	1.11e-06	6	6	2.33e-01	2.56e-01	96.9
	4	6.42e-04	8.57e-08	7.72e-04	2.30e-07	6	5	1.98e-01	1.72e-01	81.6
	5	6.39e-04	7.07e-08	7.23e-04	2.30e-07	6	6	1.82e-01	2.02e-01	87.8
	6	7.44e-04	8.29e-08	8.33e-04	2.66e-07	6	6	1.98e-01	2.05e-01	89.4
	7	7.37e-04	8.34e-08	8.30e-04	2.67e-07	6	6	1.93e-01	2.04e-01	89.3
2(ID)	3	3.80e-06	7.40e-11	1.02e-05	2.75e-10	3	3	4.73e-02	5.09e-02	174
	4	2.33e-06	3.65e-12	3.98e-06	1.40e-11	3	3	3.13e-02	3.35e-02	155
	5	1.84e-06	3.41e-12	3.48e-06	1.34e-11	3	3	2.66e-02	3.36e-02	158
	6	2.18e-06	2.71e-12	4.23e-06	1.04e-11	3	3	3.26e-02	3.49e-02	161
	7	2.21e-06	3.20e-12	4.18e-06	1.24e-11	3	3	2.99e-02	3.45e-02	161

The previous table is consistent with the independence of the experimental measures and the grid size. Thus we may conclude that FAPIN also behaves like a multigrid algorithm in this case. We note, however, some small deviations from the general trend in the case $k_m = 3$, perhaps due to the two-grid nature of these experiments.

5. Two-Dimensional Fourth Order Model Problems

In this section we present the experimental results for our two-dimensional fourth order problems. We begin with the non-periodic ones (Tables 5.1.a, 5.1.b, 5.2.a and

5.2.b) and summarize our results in Table 5.3. Finally we give the results for the periodic case in Table 5.4. Comments on each set of results are given after the corresponding set of tables.

Tables 5.1.a and 5.1.b present the results for PT.H1.1 and PT.H1.2, respectively. The finite element basis used in these cases consisted of piecewise bicubic Hermite functions built through Kronecker products of one-dimensional piecewise cubic Hermite functions, scaled according to scaling 1, described in Ap.I.§3, and satisfying the pertinent boundary conditions. The coarsest grid was always that corresponding to $k = 0$. For these problems we tried two different approximate inverses : An LSQ-approximate inverse of pattern F and the corresponding 2(F). The sparsity pattern for the one-dimensional counterpart of F, for the case $k = 3$, is obtained from the first pattern in Figure 4.1 (previous section) by deleting the first row and first column. These approximate inverses were used in all grids except those corresponding to $k = 1,0$, for which an LSQ-approximate inverse of pattern pattern ID was used. In an analogous fashion, the sparsity patterns for the one dimensional counterparts of the discrete differential operators, for the grids $k = 3,2,1,0$, are obtained form those in Figure 4.2 by again deleting the first row and first column. We recall that the number of unknowns for these experiments is $(2(2^{k_m} + 1)-2)^2$ (see Table 2.1).

Table 5.1.a : Results for PT.H1.1

AI	k_m	$P(3)\| . \|_A$	$P(8)\| . \|_A$	$P(3)\| . \|_2$	$P(8)\| . \|_2$	N_∞	N_2	Q_∞	Q_2	η
F	3	1.18e-02	1.00e-04	1.05e-02	9.47e-05	11	10	4.10e-01	4.06e-01	460
	4	1.48e-02	1.39e-04	1.45e-02	1.35e-04	12	10	3.97e-01	4.04e-01	471
	5	1.35e-02	1.16e-04	1.23e-02	1.08e-04	12	10	4.17e-01	4.09e-01	474
	6	1.40e-02	1.37e-04	1.30e-02	1.29e-04	12	10	4.11e-01	4.15e-01	479
2(F)	3	3.50e-04	3.19e-08	3.49e-04	3.80e-08	6	5	1.90e-01	1.50e-01	409
	4	2.28e-04	1.87e-08	2.65e-04	3.15e-08	6	5	1.96e-01	1.51e-01	413
	5	1.68e-04	1.12e-08	1.94e-04	2.19e-08	6	5	1.97e-01	1.47e-01	414
	6	1.75e-04	7.84e-09	2.04e-04	1.72e-08	6	6	1.99e-01	1.57e-01	433

Table 4.1.b : Results for PT.H1.2

AI	k_m	$P(3)\| . \|_A$	$P(8)\| . \|_A$	$P(3)\| . \|_2$	$P(8)\| . \|_2$	N_∞	N_2	Q_∞	Q_2	η
F	3	9.44e-03	1.66e-04	5.80e-03	1.08e-04	12	10	4.34e-01	4.38e-01	472
	4	9.00e-03	7.99e-05	5.10e-03	5.14e-05	12	10	4.36e-01	4.28e-01	479
	5	5.60e-03	4.59e-05	3.13e-03	3.23e-05	12	11	4.31e-01	4.37e-01	488
	6	3.51e-03	2.68e-05	2.05e-03	2.19e-05	13	11	4.32e-01	4.37e-01	498
2(F)	3	4.51e-04	9.93e-08	3.29e-04	7.68e-08	6	5	1.91e-01	1.90e-01	416
	4	2.97e-04	7.01e-08	2.31e-04	6.32e-08	7	6	1.96e-01	1.97e-01	426
	5	1.74e-04	4.47e-08	1.58e-04	4.72e-08	7	6	1.96e-01	1.99e-01	428
	6	1.04e-04	2.93e-08	1.15e-04	3.47e-08	7	6	1.96e-01	1.99e-01	440

The above tables show the multigrid behavior of FAPIN when applied to these model problems, since the experimental measures are essentially independent of the grid size.

In Tables 5.2.a and 5.2.b we give the results for PT.H2.1 and PT.H2.2, respectively. These problems differ from PT.H1.1 and PT.H1.2 in that the basis functions were scaled according to scaling 2, described in Ap.I.§4. The 'type .1' problems also differ from the previous ones in their corresponding right hand sides, since these are built by application of the discrete differential operator for the finest grid. The number of unknowns for these experiments are the same as for PT.H1.1 and PT.H1.2.

Table 5.2.a : Results for PT.H2.1

AI	k_m	$P(3)\|\,.\,\|_A$	$P(8)\|\,.\,\|_A$	$P(3)\|\,.\,\|_2$	$P(8)\|\,.\,\|_2$	N_∞	N_2	Q_∞	Q_2	η
F	3	7.71e-04	7.11e-08	9.70e-04	8.78e-08	8	8	1.64e-01	1.67e-01	343
	4	9.30e-04	8.25e-08	1.21e-03	1.11e-07	9	9	2.08e-01	1.66e-01	380
	5	9.69e-04	8.80e-08	1.26e-03	1.17e-07	10	9	2.22e-01	1.60e-01	411
	6	9.86e-04	8.97e-08	1.29e-03	1.18e-07	10	10	1.87e-01	1.59e-01	427
2(F)	3	2.04e-05	3.97e-11	2.37e-05	4.65e-11	5	5	6.73e-02	7.03e-02	378
	4	1.17e-05	2.31e-11	1.42e-05	2.81e-11	6	6	6.98e-02	7.34e-02	418
	5	8.77e-06	1.73e-11	1.09e-05	2.17e-11	7	6	6.84e-02	7.43e-02	463
	6	6.27e-06	1.22e-11	8.07e-06	1.55e-11	7	7	6.90e-02	7.89e-02	483

Table 5.2.b : Results for PT.H2.2

AI	k_m	$P(3)\|\,.\,\|_A$	$P(8)\|\,.\,\|_A$	$P(3)\|\,.\,\|_2$	$P(8)\|\,.\,\|_2$	N_∞	N_2	Q_∞	Q_2	η
F	3	1.02e-03	1.15e-07	1.24e-03	1.34e-07	9	8	2.15e-01	1.80e-01	365
	4	1.14e-03	9.83e-08	1.39e-03	1.21e-07	10	9	1.64e-01	1.56e-01	388
	5	1.20e-03	1.08e-07	1.47e-03	1.32e-07	10	9	1.60e-01	1.55e-01	420
	6	1.24e-03	1.12e-07	1.51e-03	1.37e-07	11	10	1.58e-01	1.56e-01	437
2(F)	3	2.71e-05	6.19e-11	3.00e-05	6.89e-11	6	5	6.87e-02	7.33e-02	398
	4	1.55e-05	3.25e-11	1.78e-05	3.73e-11	6	6	6.93e-02	7.38e-02	428
	5	1.00e-05	2.33e-11	1.19e-05	2.72e-11	7	6	7.00e-02	7.63e-02	475
	6	7.41e-06	1.57e-11	8.98e-06	1.87e-11	7	7	6.88e-02	7.95e-02	491

The above tables again show the multigrid behavior of the algorithm FAPIN based on the newly scaled basis. We recall that PT.H1.2 and PT.H2.2 represent two different discretizations (differing in the scaling of the finite element basis chosen) of the same boundary value problem.

The results given thus far correspond to the non-periodic cases. They are summarized in Table 5.3, where all the experimental measures are given for the indicated model problem with a finest grid corresponding to $k_m = 6$.

Table 5.3 : Non-Periodic Plate Experiments Summary

Problem	AI	$P(8)_{\Vert\,.\,\Vert_A}$	$P(8)_{\Vert\,.\,\Vert_2}$	$P(8)_{\Vert\,.\,\Vert_\infty}$	N_A	N_2	N_∞	Q_A	Q_2	Q_∞	η
PT.H1.1	F	1.37e-04	1.29e-04	2.65e-04	12	10	12	4.18e-01	4.15e-01	4.11e-01	479
	2(F)	7.84e-09	1.72e-08	8.10e-08	6	6	6	1.30e-01	1.57e-01	1.99e-01	433
PT.H1.2	F	2.68e-05	2.19e-05	9.33e-05	13	11	13	4.38e-01	4.37e-01	4.32e-01	498
	2(F)	2.93e-08	3.47e-08	2.15e-07	7	6	7	2.00e-01	1.99e-01	1.96e-01	440
PT.H2.1	F	8.97e-08	1.18e-07	1.14e-07	10	10	10	1.57e-01	1.59e-01	1.87e-01	427
	2(F)	1.22e-11	1.55e-11	4.26e-11	7	7	7	7.86e-02	7.89e-02	6.90e-02	483
PT.H2.2	F	1.12e-07	1.37e-07	2.09e-07	11	10	11	1.56e-01	1.56e-01	1.58e-01	437
	2(F)	1.57e-11	1.87e-11	1.05e-10	7	7	7	7.91e-02	7.95e-02	6.88e-02	491

Comparing the above tables we observe that the efficiency of the algorithm FAPIN is not improved , in terms of the experimental effort η, by the use of two smoothing steps (i.e, 2(F)). We also note that the effect of the new scaling of the basis is, for these problems, almost negligible.

Next we give the results for our two-dimensional fourth order periodic model problem. Table 5.5 presents the results for model problem PT.B.P. The finite element space used in this case consisted of biperiodic bicubic splines built through Kronecker products of one-dimensional periodic cubic splines. The coarsest grid was, in all runs, that corresponding to $k = 2$. Thus the case $k_m = 3$ is actually a two-grid algorithm. The approximate pseudo-inverses used were LSQ-approximate pseudo-inverses of pattern ID for all grids except the coarsest one, in which we used an LSQ-approximate pseudo-inverse with the sparsity pattern of a 15-diagonal matrix (since the discrete differential operator is a singular full matrix). Some examples of the one-dimensional counterparts of the above sparsity patterns (except for $k = 2$) are given in Figure 3.2 of the previous section. We recall that the number of unknowns for these experiments is $(2^{k_m})^2$

Table 5.4 : Results for PT.B.P

k_m	$P(3)\| \cdot \|_A$	$P(8)\| \cdot \|_A$	$P(3)\| \cdot \|_2$	$P(8)\| \cdot \|_2$	N_∞	N_2	Q_∞	Q_2	η
3	1.03e-04	4.54e-09	1.31e-04	8.76e-09	4	4	1.04e-01	1.00e-01	59.6
4	5.53e-04	1.85e-08	5.18e-04	1.69e-08	5	4	1.00e-01	1.16e-01	64.2
5	4.23e-04	8.92e-09	4.03e-04	8.39e-09	5	5	9.54e-02	1.10e-01	67.1
6	4.97e-04	9.64e-09	4.72e-04	9.03e-09	5	5	1.18e-01	1.10e-01	69.2
7	5.01e-04	1.07e-08	4.73e-04	1.00e-08	5	5	9.55e-02	1.12e-01	72

The previous table is consistent with the independence of the experimental results from the grid size. Thus we may conclude that FAPIN also behaves like a multigrid algorithm in this case.

6. Conclusions and Suggestions for Further Study

In this thesis we have shown the ability of the multigrid algorithm FAPIN to solve singular and nonsingular large sparse linear systems of equations constructed from the finite element discretization of several boundary value problems with periodic and non-periodic boundary conditions. In particular, our experiments represent new experience with FAPIN applied to fourth order problems and with FAPIN using cubic B-spline, bicubic B-spline and piecewise bicubic Hermite bases.

Our work also suggests several topics for further study :

(i) As pointed out in §1, the solutions to the boundary value problems used to build our two-dimensional model problems have singularities at the corners of the domain $\Omega \cup \Gamma$ (see Stephan [1979] and Strang & Fix [1973]). Since the ability of the discretization to approximate the eigenvectors of the continuous operator seems to play an important role in multigrid algorithm design (see McCormick [1982,1984]), the effect on FAPIN of including singular functions in the basis for the discretization, as suggested by Strang & Fix [1973, p.263], should be explored.

(ii) We have seen (cf. §4&5) that the scaling of the basis functions can have a strong effect on the performance of FAPIN. A detailed study of scaling and FAPIN would be worthwhile.

(iii) In this thesis we have only considered constant coefficient boundary value problems. Future work should explore the applicability of FAPIN to the solution of the non-constant coefficient versions of our problems.

(iv) We have shown that the LSQ-approximate inverse and the LSQ-approximate pseudo-inverse are well suited for the construction of the required multigrid smoothers for the singular and nonsingular versions of FAPIN, respectively. Why this is the case is, as yet, not clear. Also, comparison with FAPIN built on other approximate inverses (e.g. those described in Benson [1973], or perhaps approximate inverses tailored to satisfy the nested property of Ch.I.D4.1) seems to be desirable.

(v) It has been demostrated here and elsewhere (see Frederickson & Benson [1986]) that FAPIN is an efficient Poisson solver. Thus, its applicability to coupled equation formulations of fourth order problems (e.g. Smith [1968,1970], Ehrlich [1971,1972,1973], Greenspan & Schultz [1972], McLaurin [1974]) and mixed finite element formulations (e.g. Scholtz [1978], Glowinski & Pironneau [1979], Stephan [1979], Scapolla [1980]) should be explored.

(vi) FAPIN has been implemented by Frederickson & Benson [1976] on a hypercube multiprocessor architecture and applied to the solution of periodic second order problems. New experiments with the fourth order problems considered in this thesis would be worthwhile.

BIBLIOGRAPHY

Atkinson, K. E., *An Introduction to Numerical Analysis,* John Wiley & Sons., 1978.

Bank, R. E., "Efficient Algorithms for Solving Tensor Product Finite Element Equations," *Numer. Math.*, vol. 31, pp. 49-61, 1978.

Bank, R. E. and C. C. Douglas, "Sharp Estimates for Multigrid Rates of Convergence with General Smoothing and Acceleration," *SIAM J. Numer. Anal.*, vol. 22, No. 4, pp. 617-633, 1985.

Baumgardner, J. R. and P. O. Frederickson, "Icosahedral Discretization of the Two-Sphere," *SIAM J. Numer. Anal.*, vol. 22, no. 6, pp. 1107-1115, 1985.

Ben-Israel, A., "An iterative Method for Computing the generalized Inverse of an Arbitrary Matrix," *Math. of. Comp.*, vol. 19, pp. 452-455, 1965.

Ben-Israel, A., "A Note on an Iterative Method for Generalized Inversion of Matrices," *Math. of Comp.*, vol. 20, pp. 439-440, 1966.

Ben-Israel, A. and D. Cohen, "On Iterative Computation of the Generalized Inverse and Associated Projections," *SIAM J. Numer. Anal.*, vol. 3, no. 3, pp. 410-419, 1966.

Ben-Israel, A. and T. N. E. Greville, *Generalized Inverses: Theory and Applications,* Wiley-Interscience, John Wiley & Sons, 1974.

Benson, M. W., "Iterative Solution of Large Scale Linear Systems," Mathematics Report #17-73 (M.Sc. Thesis). Lakehead University, Thunder Bay, Ontario, Canada, 1973.

Benson, M. W., "A High Level Approach to Scientific Computing," Mathematics Report #1-87. Lakehead University, Thunder BAy, Ontario, Canada, 1987.

Benson, M. W. and P. O. Frederickson, "Iterative Solution of Large Sparse Linear Systems Arising in Certain Multidimensional Approximation Problems," *Utilitas Mathematica,* vol. 22, pp. 127-140, 1982.

Benson, M. W. and P. O. Frederickson, "Fast Parallel Algorithms for the Moore-Penrose Pseudo-inverse Solution to Large Sparse Consistent Systems," Computer Science Report CCS 86/19. Chr. Michelsen Institute, Bergen, Norway., 1986.

Benson, M. W. and P. O. Frederickson, "Fast Parallel Algorithms for the Moore-Penrose Pseudo-Inverse," in *Hypercube Multiprocessors,* ed. M. T. Heath, pp. 597-604, SIAM, 1987.

Benson, M. W., J. Krettman, and M. Wright, "Parallel Algorithms for the Solution of Certain Large Sparse Linear Systems," *Intern. J. Computer Math.*, vol. 16, pp. 245-260, 1984.

Brandt, A., "Multi-Level Adaptive Solutions to Boundary Value Problems," *Math. Comp.*, vol. 31, no. 138, pp. 333-390, 1977.

Burden, R. L., J. D. Faires, and A. C. reynolds, *Numerical Analysis,* Prindle, Weber & Schmidt, 1981.

Chew, K.-T., "Finite Element Solutions to Boundary Value Problems," M.Sc. Thesis, Lakehead University, 1977.

de Boor, C., *A practical Guide to Splines,* Applied Mathematical Sciences No. 27, Springer-Verlag, 1978.

Douglas, C. C., "Multi-Grid Algorithms with applications to Elliptic Boundary Value Problems," *SIAM J. Numer. Anal.*, vol. 21, No. 2, pp. 236-254, 1984.

Douglas, J. Jr. and T. Dupont, "Alternating-Direction Galerkin Methods on Rectangles," in *Numerical Solution of Partial Differential Equations-II, SYNSPADE 1970,* ed. Bert Hubbard, pp. 133-214, Academic Press, 1971.

Ehrlich, L. W., "Solving the Biharmonic Equation as Coupled Finite Difference Equations," *SIAM J. Numer. Anal.*, vol. 8, no. 2, pp. 278-287, 1971.

Ehrlich, L. W., "Coupled Harmonic Equations, SOR, and Chebyshev Acceleration," *Math. Comp.*, vol. 26, no. 118, pp. 335-343, 1972.

Ehrlich, L. W., "Solving the Biharmonic Equation in a Square : A Direct Versus a Semidirect Method," *Comm. ACM,* vol. 16, no. 11, pp. 711-714, 1973.

Frederickson, P. O., "Fast Approximate Inversion of Large Sparse Linear Systems," Mathematics Report #7-75. Lakehad University, Thunder Bay, Ontario, Canada., 1975.

Frederickson, P. O. and M. W. Benson, "Fast Parallel Solution of Large Sparse Linear Systems," Computer Science Report CCS 86/9. Chr. Michelsen Institute, Bergen, Norway, 1986.

Froberg, C. E., *Introduction to Numerical Analysis,* Addison Wesley, 1969.

Glowinski, R. and O. Pironneau, "Numerical Methods for the First Biharmonic Equation and for the Two-Dimensional Stokes Problem," *SIAM Sirev.*, vol. 21, no. 2, pp. 165-212, 1979.

Greenbaum, A., "Analysis of a Multigrid Method as an Iterative Technique for Solving Linear Systems," *SIAM J. Numer. Anal.*, vol. 21, No. 3, pp. 473-485, 1984.

Greenspan, D. and D. Schultz, "Fast Finite-Difference Solution of Biharmonic Problems," *Comm. ACM,* vol. 15, no. 5, pp. 347-350, 1972.

Hackbusch, W. and U. Trottenberg (editors), *Multigrid Methods,* Lecture Notes in Mathematics, 960, Springer-Verlag, 1982.

Jacobson, N., *Lectures in Abstract Algebra,* Graduate Texts in Mathematics No. 31, II. Linear Algebra, Springer-Verlag, 1953.

Kaufman, L. and D. D. Warner, "High-Order, Fast-Direct Methods for Separable Elliptic Equations," *SIAM J. Numer. Anal.*, vol. 21, No. 4, pp. 672-694, 1984.

Kernighan, B. W. and R. Pike, *The UNIX Programming Environment,* Prentice-Hall Software Series, Prentice-Hall, 1984.

Kernighan, B. W. and D. M. Ritchie, *The C Programming Language,* Prentice-Hall Software Series, Prentice-Hall, 1978.

Liong, O. H., "Triangular Finite Element Solution to Boundary Value Problems," M.Sc. Thesis, Lakehead University, 1977.

Maitre, J-F. and F. Musy, "Multigrid Methods: Convergence Theory in a Variational Framework," *SIAM J. Numer. Anal.*, vol. 21, No. 4, pp. 657-671, 1984.

Margenov, S. D., "Application of Parabolic and Cubic Splines for Solving Boundary Value Problems of Mixed Type for a Biharmonic Equation in a Rectangle," *Serdica 7, (Russian),* vol. No. 3, pp. 211-216, 1981.

McCormick, S. F., "Multigrid Methods for Variational Problems: Further Results," *SIAM J. Numer. Anal.*, vol. 21, No. 2, pp. 255-263, 1984.

McCormick, S. F., "Multigrid Methods for Variational Problems: General Theory for the V-Cycle," *SIAM J. Numer. Anal.*, vol. 22, No. 4, pp. 634-643, 1985.

McCormick, S. F. and J. W. Ruge, "Multigrid Methods for Variational Problems," *SIAM J. Numer. Anal.*, vol. 19, No. 5, pp. 924-929, 1982.

McLaurin, J. W., "A General Coupled Equation Approach for Solving the Biharmonic Boundary Value Problem," *SIAM J. Numer. Anal.*, vol. 11, no. 1, pp. 14-33, 1974.

Nicolaides, R. A., "On the l Convergence of an Algorithm for Solving Finite Element Equations," *Math. Comp.*, vol. 31, No. 140, pp. 892-906, 1977.

Oden, J. T. and J. N. Reddy, *An Introduction to the Mathematics of Finite Elements,* Pure & Applied Mathematics, Wiley-Interscience, 1976.

Prenter, P. M., *Splines and Variational Methods,* Wiley-Interscience, John Wiley & Sons, 1975.

Rivara, M.-C., "Design and Data Structure of a Fully Adaptive Multigrid, Finite-Element Software," *ACM Trans. Math. Softw.*, vol. 10, no. 3, pp. 242-264, 1984.

Scapolla, T., "A Mixed Finite Element Method for the Biharmonic Problem," *RAIRO Num. Anal.*, vol. 14, no. 1, pp. 55-79, 1980.

Scholz, R., "A Mixed Method for 4th Order Problems Using Linear Finite Elements," *RAIRO Numer. Anal.*, vol. 12, no. 1, pp. 85-90, 1978.

Schultz, M. S., "Multivariate Spline Functions and Elliptic Problems," in *Approximations with Special Emphasis on Spline Functions,* ed. I. J. Schoenberg, pp. 279-347, Academic Press, 1969.

Smith, J., "The Coupled Equation Approach to the Numerical Solution of the Biharmonic Equation by Finite Differences. I," *SIAM J. Numer. Anal.*, vol. 5, no. 2, pp. 323-339, 1968.

Smith, J., "The Coupled Equation Approach to the Numerical Solution of the Biharmonic Equation by Finite Differences. II," *SIAM J. Numer. Anal.*, vol. 7, no. 1, pp. 104-111, 1970.

Soderstrom, T. and G. W. Stewart, "On the Numerical Properties of an Iterative Method for Computing the Moore-Penrose Generalized Inverse," *SIAM J. Numer. Anal.*, vol. 11, no. 1, pp. 61-74, 1974.

Stephan, E., "Conform and Mixed Finite Element Schemes for the Dirichlet Problem for the Bilaplacian in Plane Domains with Corners," *Math. Meth. Appl. Sci.*, vol. 1, no. 3, pp. 354-382, 1979.

Stewart, G. W., *Introduction to Matrix Computations,* Computer Science and Applied Mathematics, Academic Press, 1973.

Strang, G. and G. J. Fix, *An Analysis of the Finite Element Method,* Series in Atutomatic Computation, Prentice-Hall, 1973.

Young, D. M., *Iterative Solution of Large Linear Systems,* Academic Press, 1971.

APPENDIX I

Calculation of Element Matrices

In this appendix we calculate the element matrices needed for the finite element discretization of our model problems. Our work relies heavily on that in Strang & Fix [1973, pp.61-51], though we apply their methods to bases not treated there.

We do so for a cubic spline space and for a piecewise cubic Hermite space. The element matrices for a piecewise linear basis are given in Strang & Fix [1973, p.29]. Since they are calculated in a fashion similar to that of the more interesting Hermite basis, we do not present them here.

We also calculate the element matrices for two rescaled versions of the piecewise cubic Hermite basis, which lead to matrices with a mesh size dependency contained only in a single multiplicative factor. The latter matrices will be used in the construction of our non-periodic two-dimensional model problems, through a Kronecker product formulation.

1. Cubic Splines.

We want to calculate the element mass, stiffness and bending matrices (see Strang & Fix [1973]) for a B-spline basis used in the discretization of constant coefficient one-dimensional problems. Without loss of generality we assume that the problem to be solved is given in the interval $I = [0,b]$. Let Π_h denote a uniform partition of this interval with mesh size $h = \frac{b}{N,}$ $N \in \mathbf{Z}^+$. We recognize as finite element nodes the division points for the subintervals arising from the partition Π_h. First we must build a basis for the finite dimensional subspace of cubic splines over Π_h. Let $S_3(\Pi_h)$ denote the space of $\mathbf{C}^2(I)$ cubic splines over the partition Π_h. This is a subspace of the Sobolev space $\mathbf{H}^3(I)$ (see Oden & Reddy [1976, p.90]). A basis for $S_3(\Pi_h)$ is given by the set:

$$\mathbf{B}_S = \left\{ \phi_{-1}, \phi_0, \phi_1, \ . \ . \ . \ , \phi_N, \phi_{N+1} \right\},$$

where, as usual,

$$\phi_i = B(\frac{x}{h} - i)$$

and $B(t)$ is the cubic B-spline function with support $[-2,2]$ such that $B(0) = 1$:

$$B(t) = \begin{cases} \frac{1}{4}(t+2)^3 & \text{if } t \in [-2,-1] \\ \frac{1}{4} + \frac{3}{4}(t+1) + \frac{3}{4}(t+1)^2 - \frac{3}{4}(t+1)^3 & \text{if } t \in [-1,0] \\ \frac{1}{4} + \frac{3}{4}(1-t) + \frac{3}{4}(1-t)^2 - \frac{3}{4}(1-t)^3 & \text{if } t \in [0,1] \\ \frac{1}{4}(2-t)^3 & \text{if } t \in [1,2] \end{cases}$$

We let $v^h(x)$ represent an arbitrary trial function in $S_3(\Pi_h)$. Since $v^h(x) \in S_3(\Pi_h)$, it can be expanded in terms of the basis $\mathbf{B}_S$:

$$v^h(x) = \sum_{i=-1}^{N+1} q_i \phi_i(x).$$

Now we proceed to calculate the desired matrices. We are interested in evaluating integrals of the form :

$$J = \int_{jh}^{(j+1)h} (g^h(x))^2 dx \ ,$$

where $g^h(x)$ is $v^h(x)$, $\frac{d}{dx}v^h(x)$ and $\frac{d^2}{dx^2}v^h(x)$ for the mass, stiffness and bending matrices, respectively. Since, in every case, $g^h(x)$ is a linear combination of functions which have compact support in I, the number of terms in the expansion of $g^h(x)$ that do not vanish in $[jh,(j+1)h]$ is small. In fact, since the support of each ϕ_i is $[(i-2)h,(i+2)h]$, $g^h(x)$ is a linear combination of four terms involving only q_{j-1}, q_j, q_{j+1}, q_{j+2}, in each subinterval of the form $[jh,(j+1)h]$. Moreover, since each ϕ_i is a translated and scaled by h version of the same function $B(t)$, the functional dependence of J on h, j and the coefficients q_{j-1}, q_j, q_{j+1}, q_{j+2}, will be the same in every subinterval of the form $[jh,(j+1)h]$. Therefore, for simplicity in the calculations and without loss of generality, we may restrict ourselves to the interval $[0,h]$. Now, since $v^h(x) \in S_3(\Pi_h)$, it is a cubic on $[0,h]$, and thus can be expressed as :

$$v^h(x) = a_0+a_1x+a_2x^2+a_3x^3 \ , \ \ x \in [0,h].$$

This is the approach taken by Strang & Fix [1973]. It simplifies considerably the calculations and can be extended to higher dimensions. J can now be written as :

$$J = a^t \ N \ a$$

with

$$a^t = (a_0, \ a_1, \ a_2, \ a_3),$$

and all we have to calculate is the matrix H that gives the change of coefficients $q^t = (q_{-1}, q_0, q_1, q_2)$ to $a^t = (a_0, a_1, a_2, a_3)$. That is, we need a matrix H such that $a = Hq$.

We will use k with a subscript 0, 1 or 2 to denote the element mass, stiffness or bending matrices, respectively. With this notation we have :

$$k_i = H^t N_i H \ , \ \ i = 0, 1, 2 \ \ , \qquad (1.1)$$

where :

$$H = \begin{pmatrix} \frac{1}{4} & 1 & \frac{1}{4} & 0 \\ \frac{-3}{4h} & 0 & \frac{3}{4h} & 0 \\ \frac{3}{4h^2} & \frac{-6}{4h^2} & \frac{3}{4h^2} & 0 \\ \frac{-1}{4h^3} & \frac{3}{4h^3} & \frac{-3}{4h^3} & \frac{1}{4h^3} \end{pmatrix}$$

which gives the desired transformation among the coefficients q and a. The matrices N_i are easily calculated. They are (cf. Strang & Fix, p. 57,58) :

$$N_0 = \begin{pmatrix} h & \frac{h^2}{2} & \frac{h^3}{3} & \frac{h^4}{4} \\ \frac{h^2}{2} & \frac{h^3}{3} & \frac{h^4}{4} & \frac{h^5}{5} \\ \frac{h^3}{3} & \frac{h^4}{4} & \frac{h^5}{5} & \frac{h^6}{6} \\ \frac{h^4}{4} & \frac{h^5}{5} & \frac{h^6}{6} & \frac{h^7}{7} \end{pmatrix}$$

$$N_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & h & h^2 & h^3 \\ 0 & h^2 & \frac{4h^3}{3} & \frac{3h^4}{2} \\ 0 & h^3 & \frac{3h^4}{2} & \frac{9h^5}{5} \end{pmatrix}$$

$$N_2 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 4h & 6h^2 \\ 0 & 0 & 6h^2 & 12h^3 \end{pmatrix}$$

Carrying out the matrix multiplications indicated in (1.1) we arrive to the desired element matrices, which are to be completed by symmetry :

$$k_0 = h \begin{pmatrix} \frac{1}{112} & \frac{129}{2240} & \frac{3}{112} & \frac{1}{2240} \\ & \frac{297}{560} & \frac{933}{2240} & \frac{3}{112} \\ & & \frac{297}{560} & \frac{129}{2240} \\ & & & \frac{1}{112} \end{pmatrix}$$

$$k_1 = \frac{1}{h} \begin{pmatrix} \frac{9}{80} & \frac{21}{160} & \frac{-9}{40} & \frac{-3}{160} \\ & \frac{51}{80} & \frac{-87}{160} & \frac{-9}{40} \\ & & \frac{51}{80} & \frac{21}{160} \\ & & & \frac{18}{160} \end{pmatrix}$$

and finally :

$$k_2 = \frac{1}{h^3} \begin{pmatrix} \frac{3}{4} & \frac{-9}{8} & 0 & \frac{3}{8} \\ & \frac{9}{4} & \frac{-9}{8} & 0 \\ & & \frac{9}{4} & \frac{-9}{8} \\ & & & \frac{3}{4} \end{pmatrix}$$

We remark that the matrices that would result from assembling the above element matrices, apply only to problems with free boundaries (natural boundary conditions). In order to use them for problems with essential boundary conditions, these conditions must be imposed on the basis functions. This is done in the next appendix for the case of periodic and certain homogeneous boundary conditions.

2. Piecewise Cubic Hermites

We want to calculate the element mass, stiffness and bending matrices for a piecewise cubic Hermite basis (PWC-Hermite) used for the discretization of constant coefficient one-dimensional problems. We use the same notation as in the previous section and, again, without loss of generality we assume that the problem to be solved is given in the interval $I = [0,b]$. We recognize as finite element nodes the division points for the subintervals arising from the partition Π_h, but in this case, there are two nodal parameters associated with each node : The function value and the first derivative at the corresponding node. First we build a basis for the finite dimensional space of PWC-Hermite functions over Π_h. Let $\mathbf{S}_H(\Pi_h)$ denote the space of $\mathbf{C}^1(I)$ PWC-Hermites over the partition Π_h. This is a subspace of the Sobolev space $\mathbf{H}^2(I)$ (see Oden & Reddy [1976, p.90]). A basis for $S_H(\Pi_h)$ is given by the set:

$$\mathbf{B}_H = \left\{ \psi_0, \omega_0, \psi_1, \omega_1, \ldots, \psi_N, \omega_N \right\},$$

where, as usual in the nodal finite element method (see Strang & Fix [1974, p.101]),

$$\psi_i = P(\frac{x}{h} - i)$$
$$\omega_i = h \ W(\frac{x}{h} - i)$$

and $P(t)$, $W(t)$ are the Hermite cubics with support $[-1,1]$ and such that $P(0) = 1$, $\dfrac{dW(0)}{dx} = 1$, that is (see Strang & Fix [1974, p.56) :

$$P(t) = \begin{cases} (t+1)^2\,(1-2t) & \text{if } t \in [-1,0] \\ (t-1)^2\,(2t+1) & \text{if } t \in [0,1] \end{cases}$$

for the function values and :

$$W(t) = \begin{cases} t\ (t+1)^2 & \text{if } t \in [-1,0] \\ t\ (t-1)^2 & \text{if } t \in [0,1] \end{cases}$$

for the derivative values. Observe that :

$$\begin{cases} P(0) = 1 \quad \dfrac{dP}{dx}(0) = 0 \quad P(1) = 0 \quad \dfrac{dP}{dx}(1) = 0 \\ W(0) = 0 \quad \dfrac{dW}{dx}(0) = 1 \quad W(1) = 0 \quad \dfrac{dW}{dx}(1) = 0 \end{cases}$$

thus satisfying the requirements of a nodal finite element basis.

Following Strang & Fix [1974, pp.56-57] we observe that a general cubic function $v^h(x)$ in the interval $[0,h]$ can be expressed as :

$$v^h(x) = v_0\, P(\frac{x}{h}) + v'_0\, h\;\; W(\frac{x}{h}) + v_1\, P(\frac{x}{h} - 1) + v'_1\, h\;\; W(\frac{x}{h} - 1)\,. \tag{2.1}$$

where the parameters v, v' denote the function values and the derivative values of $v(x)$ at the points 0 and 1. The h factors are needed since $\frac{dW}{dx}(\frac{x}{h}) = \frac{1}{h}\,\frac{dW}{dx}(x)$.

Now, substituting and collecting terms in powers of x, we obtain :

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \frac{-3}{h^2} & \frac{-2}{h} & \frac{3}{h^2} & \frac{-1}{h} \\ \frac{2}{h^3} & \frac{1}{h^2} & \frac{-2}{h^3} & \frac{1}{h^2} \end{pmatrix} \begin{pmatrix} v_0 \\ v'_0 \\ v_1 \\ v'_1 \end{pmatrix}$$

where, as in the previous section :

$$v^h(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3\ ,\quad x \in [0,h]\ .$$

Observe that the required integrals are still given by the same matrices N_i as in the B-spline case. Thus the desired element matrices (which are to be completed by symmetry) are (see Strang & Fix [1974, p.58]) :

$$k_0 = \frac{h}{420}\begin{pmatrix} 156 & 22h & 54 & -13h \\ & 4h^2 & 13h & -3h^2 \\ & & 156 & -22h \\ & & & 4h^2 \end{pmatrix},$$

$$k_1 = \frac{1}{30h}\begin{pmatrix} 36 & 3h & -36 & 3h \\ & 4h^2 & -3h & -h^2 \\ & & 36 & -3h \\ & & & 4h^2 \end{pmatrix},$$

$$k_2 = \frac{1}{h^3}\begin{pmatrix} 12 & 6h & -12 & 6h \\ & 4h^2 & -6h & 2h^2 \\ & & 12 & -6h \\ & & & 4h^2 \end{pmatrix}.$$

We note that the mesh size h appears within the element matrices. Preliminary one-dimensional second order experiments showed divergence of the algorithm FAPIN built on this basis. A remedy was found by rescaling the basis $\mathbf{B}_H$ in such a manner that the

h dependency of the entries in the above matrices was contained only in a single multiplicative factor. This is the subject of the next section.

The matrices that would result from the assembly of the above element matrices, apply only to problems with free boundaries. For this basis, however, imposing homogeneous boundary conditions only amounts to a simple deletion of appropriate rows and columns in the assembled matrix.

3. Rescaling of the Piecewise Cubic Hermite Basis : Scaling 1

We will now show a redefinition of the basis $\mathbf{B}_H$ that allows the reduction of the mesh size dependency in the above element matrices to a single multiplicative factor.

We let $\eta_i \equiv h\ \psi_i$, $i = 1, \cdots, N$ be the new rescaled basis functions. The ω_i remain unchanged. Thus, the rescaled basis is given by the set :

$$\mathbf{B}_{H1} = \left\{ \eta_0, \omega_0, \eta_1, \omega_1, \ . \ . \ . \ , \eta_N, \omega_N \right\},$$

and the vector of coefficients q becomes :

$$(\frac{1}{h}v_0,\ v'_0,\ \frac{1}{h}v_1,\ v'_1)^t \equiv (z_0,\ z'_0,\ z_1,\ z'_1)^t .$$

Thus the element matrices have to be rescaled accordingly :

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = H\ D_h\ D_h^{-1} \begin{pmatrix} v_0 \\ v'_0 \\ v_1 \\ v'_1 \end{pmatrix} = H\ D_h \begin{pmatrix} z_0 \\ z'_0 \\ z_1 \\ z'_1 \end{pmatrix} = M \begin{pmatrix} z_0 \\ z'_0 \\ z_1 \\ z'_1 \end{pmatrix},$$

where $M \equiv H\ D_h$ gives the new change of coefficients and $D_h \equiv diag(h,\ 1,\ h,\ 1)$. Thus the new element matrices are given by :

$$\begin{aligned} l_0 &= M^t\ N_0\ M = D_h{}^t\ H^t\ N_0\ H\ D_h, \\ l_1 &= M^t\ N_1\ M = D_h{}^t\ H^t\ N_1\ H\ D_h, \\ l_2 &= M^t\ N_2\ M = D_h{}^t\ H^t\ N_2\ H\ D_h, \end{aligned}$$

or, explicitly :

$$l_0 = \frac{h^3}{420}\begin{pmatrix} 156 & 22 & 54 & -13 \\ & 4 & 13 & -3 \\ & & 156 & -22 \\ & & & 4 \end{pmatrix},$$

$$l_1 = \frac{h}{30}\begin{pmatrix} 36 & 3 & -36 & 3 \\ & 4 & -3 & -1 \\ & & 36 & -3 \\ & & & 4 \end{pmatrix},$$

$$l_2 = \frac{1}{h}\begin{pmatrix} 12 & 6 & -12 & 6 \\ & 4 & -6 & 2 \\ & & 12 & -6 \\ & & & 4 \end{pmatrix}.$$

A FAPIN algorithm using this new basis $\mathbf{B}_{H1}$ showed good convergence for some one-dimensional tests. Thus, this basis was used to build the matrices for some of our model problems.

4. Rescaling of the Piecewise Cubic Hermite Basis : Scaling 2

In this section we will show another redefinition of the basis $\mathbf{B}_H$ that allows the reduction of the mesh size dependency in the corresponding element matrices to a single multiplicative factor..

As suggested by Greenbaum [1984, p.482], we use the basis functions defined in equations (3.1), (3.2) normalized to have unit $\mathbf{L}^2(I)$-norm (for simplicity, the same scaling factor is used for the basis functions at the boundaries). Thus the new basis is defined by :

$$\mathbf{B}_{H2} = \left\{ \chi_0, \theta_0, \chi_1, \theta_1, \ldots, \chi_N, \theta_N \right\},$$

where we have defined :

$$\chi_i = n_\psi \, \psi_i = n_\psi \, P(\frac{x}{h} - i)$$

$$\theta_i = n_\omega \, \omega_i = n_\omega \, h \;\; W(\frac{x}{h} - i)$$

and

$$\frac{1}{n_\psi} \equiv \left(\int_{(i-1)h}^{(i+1)h} \psi_i^2(x)\, dx \right)^{½} = \left(\frac{26}{35} h \right)^{½},$$
$$\frac{1}{n_\omega} \equiv \left(\int_{(i-1)h}^{(i+1)h} \omega_i^2(x)\, dx \right)^{½} = \left(\frac{2}{105} h^3 \right)^{½},$$

Thus, the vector of coefficients q must now be redefined as :

$$(\frac{1}{n_\psi} v_0, \frac{1}{n_\omega} v'_0, \frac{1}{n_\psi} v_1, \frac{1}{n_\omega} v'_1)^t \equiv (y_0, y'_0, y_1, y'_1)^t,$$

and the element matrices are to be rescaled accordingly :

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = H\ E_h\ E_h^{-1} \begin{pmatrix} v_0 \\ v'_0 \\ v_1 \\ v'_1 \end{pmatrix} = H\ E_h \begin{pmatrix} y_0 \\ y'_0 \\ y_1 \\ y'_1 \end{pmatrix} = Q \begin{pmatrix} y_0 \\ y'_0 \\ y_1 \\ y'_1 \end{pmatrix},$$

where $Q \equiv H\ E_h$ gives the new change of coefficients and $E_h \equiv diag(n_\psi, n_\omega, n_\psi, n_\omega)$.

Thus the new element matrices are given by :

$$m_0 = Q^t\ N_0\ Q = E_h{}^t\ H^t\ N_0\ H\ E_h,$$
$$m_1 = Q^t\ N_1\ Q = E_h{}^t\ H^t\ N_1\ H\ E_h,$$
$$m_2 = Q^t\ N_2\ Q = E_h{}^t\ H^t\ N_2\ H\ E_h,$$

or, explicitly, by :

$$m_0 = \frac{1}{21840} \begin{pmatrix} 10920 & 1144C & 3780 & -676C \\ & 10920 & 676C & -8190 \\ & & 10920 & -1144C \\ & & & 10920 \end{pmatrix},$$

$$m_1 = \frac{1}{520h^2} \begin{pmatrix} 840 & 52C & -840 & 52C \\ & 3640 & -52C & -910 \\ & & 840 & -52C \\ & & & 3640 \end{pmatrix},$$

$$m_2 = \frac{1}{13h^4} \begin{pmatrix} 210 & 78C & -210 & 78C \\ & 2730 & -78C & 1365 \\ & & 210 & -78C \\ & & & 2730 \end{pmatrix},$$

where :

$$C \equiv \left(\frac{3675}{52} \right)^{1/2} .$$

One-dimensional experiments with the FAPIN algorithm using this new basis $\mathbf{B}_{H2}$ showed good convergence. Thus we also used $\mathbf{B}_{H2}$ to construct the matrices for some of our model problems.

APPENDIX II

Imposing Essential Boundary Conditions

In this appendix we show how certain essential boundary conditions may be imposed on cubic B-spline discretizations of some two point constant coefficient boundary value problems. We examine the cases of certain homogeneous boundary conditions as well as periodic boundary conditions. In the case of non-periodic boundary conditions, this is done inexpensively by substituting certain entries in the desired assembled matrices by linear combinations of entries in the assembled matrix for the corresponding problem with natural boundary conditions. The periodic case is also inexpensive and reduces to the construction of a circulant matrix based on certain 'molecules' (or 'stencils') derived from the corresponding element matrices.

1. A Second Order Problem.

Consider the following two point boundary value problem (see Strang & Fix [1974, p.10]) :

$$(P1) \equiv \begin{cases} -\dfrac{d^2}{dx^2}u(x) = 1, \quad x \in [0,\pi]. \\ u(0) = 0, \\ \dfrac{du}{dx}(\pi) = 0. \end{cases}$$

We want to build a cubic B-spline discretization of $(P1)$. Since $u(0) = 0$ is an essential boundary condition, the Sobolev space, within which we define the finite element minimization subspace, is a subset of $\mathbf{H}^3$ (see Oden & Reddy [1976, p.90]), namely :

$$\mathbf{H}^3{}_E(I) = \left\{ \ v^h(x) \in \mathbf{H}^3(I) \mid \ v^h(0) = 0 \right\}.$$

Observe that $\mathbf{H}^3{}_E \subset \mathbf{H}^3$ is actually a subspace.

Let $I = [0,\pi]$ and let Π_h denote a uniform partition of the above interval with mesh size $h = \dfrac{\pi}{N,}$ $N \in \mathbf{Z}^+$. We recognize as finite element nodes the division points for the subintervals arising from the partition Π_h. Let $S^3(\Pi_h)$ denote the space of $\mathbf{C}^2(I)$ cubic splines over the partition Π_h. We choose as the finite element minimization subspace the finite dimensional space of cubic splines over Π_h which are zero at the origin, i.e.

$$S^3{}_E(\Pi_h) = \left\{ \ v^h(x) \in S^3(\Pi_h) \mid \ v^h(0) = 0 \right\}.$$

We also have that $\mathbf{S}^3{}_E(\Pi_h) \subset \mathbf{S}^3(\Pi_h)$ is a subspace of $\mathbf{H}^3{}_E \subset \mathbf{H}^3$.

Since the boundary condition $u(0) = 0$ is essential, in the Ritz method the basis functions must satisfy this condition. A basis for $S^3{}_E(\Pi_h)$ will satisfy it and can be easily constructed from a corresponding one for $S^3(\Pi_h)$. A basis for $S^3(\Pi_h)$ is given by the set (see Ap.II.§1) :

$$\mathbf{B}_S = \left\{ \ \phi_{-1}, \phi_0, \phi_1, \ . \ . \ . \ , \phi_N, \ \phi_{N+1} \right\},$$

where

$$\phi_i = B(\frac{x}{h} - i)$$

and $B(t)$ is the cubic B-spline function with support [—2,2] and such that $B(0) = 1$.

Define (see, for example, Prenter [1975, p.208] or Burden, Faires & Reynolds [1981, p.498]) :

$$\psi_0 = a\ \phi_{-1} + b\ \phi_0,$$
$$\psi_1 = c\ \phi_0 + d\ \phi_1,$$

and impose the conditions :

$$\psi_0(0) = 0,\ \ \psi_0(-h) = 1,$$
$$\psi_1(0) = 0,\ \ \psi_1(h) = 1.$$

Solving this system we obtain :

$$\psi_0 = \frac{16}{15}\ \phi_{-1} - \frac{4}{15}\ \phi_0,$$
$$\psi_1 = -\frac{4}{15}\ \phi_0 + \frac{16}{15}\ \phi_1.$$

Observe that the above conditions preserve the scaling of the original basis $\mathbf{B}_S$. Further, if we define :

$$\psi_i(x) = \phi_i(x),\ \ 2 \leq i \leq N{+}1,$$

it is clear that a basis for $S^3{}_E(\Pi_h)$ is given by the set :

$$\mathbf{B}_{SE} = \left\{\ \psi_0,\ \psi_1,\ .\ .\ .\ ,\psi_N,\ \psi_{N+1}\right\}.$$

This is so, since these functions satisfy $\psi_i(0) = 0,\ \forall\ i$, and are linear combinations of the ϕ_i's (which are a basis for $S^3(\Pi_h)$) and therefore span the desired subspace of $\mathbf{H}^3{}_E$. We note that since we are imposing a non-trivial condition among the functions $\phi_i(x)$, and thus removing one degree of freedom, the dimension of the minimization subspace $S^3{}_E(\Pi_h)$ is N+2 rather than N+3 as it is the case for the problem with free boundaries.

Next we proceed to calculate the assembled matrices for problem ($P1$). We will assume that the corresponding matrices in terms of the basis $\mathbf{B}_S$ are known. That is, the assembled matrices for the two point boundary value problem for the same equation with natural boundary conditions are known. This will save some work since the 'unconstrained' matrices are easily calculated by means of a simple assembly program. All we have to do now is to impose the linear relationships that define the new functions $\psi_i(x)$ in terms of the ϕ_i's.

Let $K0$, $K1$ and $K2$ denote the assembled mass, stiffness and bending matrices, respectively. Since the entries in these matrices, in the constant coefficient case, are just the $\mathbf{L}^2(I)$ inner products of the corresponding basis functions (or derivatives of the appropriate order (see Prenter [1975, p.210]), we may easily calculate the entries in these matrices in terms of the entries in the unconstrained ones. To do so, let $< , >$ denote the inner product in $\mathbf{L}^2(I)$. Then, in the case of the mass matrix $K0$, and by definition of the ψ_i's, the following relationships hold :

$$
\begin{aligned}
K0_{0,0} &= <\psi_0,\psi_0> = (\frac{4}{15})^2\,(16<\phi_{-1},\phi_{-1}> + <\phi_0,\phi_0> - 8<\phi_0,\phi_1>)\\
K0_{0,1} &= K0_{1,0} = <\psi_0,\psi_1> = (\frac{4}{15})^2\,(-4<\phi_{-1},\phi_{-1}> + 16<\phi_{-1},\phi_1> +\\
&\qquad\qquad <\phi_0,\phi_0> - 4<\phi_0,\phi_1>)\\
K0_{0,2} &= K0_{2,0} = <\psi_0,\psi_2> = (\frac{4}{15})\,(4<\phi_{-1},\phi_2> - <\phi_0,\phi_2>)\\
K0_{0,3} &= K0_{3,0} = <\psi_0,\psi_3> = (\frac{4}{15})\,(4<\phi_{-1},\phi_3> - <\phi_0,\phi_3>)\\
K1_{1,1} &= <\psi_1,\psi_1> = (\frac{4}{15})^2\,(<\phi_0,\phi_0> + 16<\phi_1,\phi_1> - 8<\phi_0,\phi_1>)\\
K0_{1,2} &= K0_{2,1} = <\psi_1,\psi_2> = (\frac{4}{15})\,(- <\phi_0,\phi_2> + 4<\phi_1,\phi_2>)\\
K0_{1,3} &= K0_{3,1} = <\psi_1,\psi_3> = (\frac{4}{15})\,(- <\phi_0,\phi_3> + 4<\phi_1,\phi_3>)\\
K0_{1,4} &= K0_{4,1} = <\psi_1,\psi_4> = (\frac{4}{15})\,(- <\phi_0,\phi_4> + 4<\phi_1,\phi_4>)
\end{aligned}
$$

Analogous relationships hold, with the basis functions ϕ_i and ψ_i replaced with their first and second derivatives, for the stiffness and bending matrices, respectively.

Thus, to calculate the assembled matrices for $(P1)$, all that has to be done, once the unconstrained matrices are known, is substitute the above entries by the corresponding linear combinations and delete the rows and columns that correspond to the degree of freedom removed by imposing the essential boundary condition. In our case, this corresponds to deleting the first row and column in the unconstrained assembled matrices.

This process also applies to the calculation of the right hand side of the linear system arising from the minimization of the energy functional over the chosen finite dimensional subspace.

2. A Fourth Order Problem.

Consider the following two point boundary value problem (see Strang & Fix [1974, p.62]) :

$$(P2) \equiv \begin{cases} \dfrac{d^4}{dx^4}u(x) = 1, \quad x \in [0,\pi]. \\ u(0) = 0, \dfrac{du}{dx}(0) = 0, \\ \dfrac{d^2}{dx^2}u(\pi) = 0, \left(\dfrac{d}{dx} - \dfrac{d^3}{dx^3}\right)u(\pi) = 0 \end{cases}$$

We want to build a cubic B-spline discretization of $(P2)$. The first two boundary conditions are essential. Thus the Sobolev space, within which we define the finite element minimization subspace, is :

$$\mathbf{H}^3{}_{E1}(I) = \left\{ \; v^h(x) \in \mathbf{H}^3(I) \mid \; v^h(0) = 0, \; \frac{d}{dx}u(0) = 0 \right\}.$$

Using the notation of the previous section, we choose as the finite element minimization subspace the finite dimensional space of cubic splines over Π_h which are zero at the origin, together with their first derivative, i.e.

$$S^3{}_{E1}(\Pi_h) = \left\{ \; v^h(x) \in S^3(\Pi_h) \mid \; v^h(0) = 0, \; \frac{d}{dx}v^h(0) = 0 \right\}.$$

We see that $\mathbf{S}^3{}_{E1}(\Pi_h)$ is a subspace of $\mathbf{H}^3{}_{E1}$.

Since these boundary conditions are essential, once again the basis functions must satisfy these conditions. A basis for $S^3{}_{E1}(\Pi_h)$ will satisfy them and it can be easily constructed from a corresponding one for $S^3{}_E(\Pi_h)$ (see previous section).

Let :

$$\xi_1 = a\ \psi_0 + b\ \psi_1,$$

and impose the conditions :

$$\xi_1(h) = 0,\ \ \frac{d}{dx}\xi_1(0) = 0.$$

Solving this system we get :

$$\xi_1 = \frac{15}{14}(\ \psi_0 + \psi_1).$$

Again the above conditions preserve the scaling of the basis $\mathbf{B}_{SE}$. If we define :

$$\xi_i(x) = \psi_i(x),\ \ 2 \leq i \leq N{+}1,$$

then a basis for $S^3{}_{E1}(\Pi_h)$ is given by the set :

$$\mathbf{B}_{SE1} = \left\{\ \xi_1,\ \xi_2,\ .\ .\ .\ ,\xi_N,\ \xi_{N+1}\ \right\},$$

since these functions satisfy the essential boundary conditions, are linear combinations of the ξ_i's and therefore span the desired subspace of $\mathbf{H}^3{}_{E1}$. We note that since we are removing one more degree of freedom, the dimension of the minimization subspace $S^3{}_{E1}(\Pi_h)$ is $N{+}1$.

With the same notation and following the procedure described in the previous section, we proceed to calculate the assembled matrices. We assume that the corresponding matrices for $(P1)$ are known. In this case, the following relationships hold among the mass matrix entries and the members of $\mathbf{B}_{SE}$ and $\mathbf{B}_{SE1}$:

$$K0_{1,1} = <\xi_1,\xi_1> = (\frac{15}{14})^2\ (<\psi_0,\psi_0> + <\psi_1,\psi_1> + 2<\psi_0,\psi_1>)$$

$$K0_{1,2} = K0_{2,1} = <\xi_1,\xi_2> = (\frac{15}{14})\ (<\psi_0,\psi_2> + <\psi_1,\psi_2>)$$

$$K0_{1,3} = K0_{3,1} = <\xi_1,\xi_3> = (\frac{15}{14})\ (<\psi_0,\psi_3> + <\psi_1,\psi_3>)$$

$$K0_{1,4} = K0_{4,1} = <\xi_1,\xi_4> = (\frac{15}{14})\,(<\phi_0,\phi_4> + <\phi_1,\phi_4>)$$

Analogous relationships hold, with the basis functions ψ_i and ξ_i replaced by their first and second derivatives, for the stiffness and bending matrices, respectively.

Thus again, to calculate the assembled matrices for $(P2)$, all that has to be done, once the matrices for $(P1)$ are known, is substitute for the above entries the corresponding linear combinations of entries and delete the rows and columns that correspond to the degree of freedom removed by imposing the new essential boundary condition (i.e. derivative condition). In this case, the first row and column in the assembled matrices for $(P1)$ should be removed.

This process again applies to the calculation of the right hand side of the linear system arising from the minimization of the energy functional over the chosen finite dimensional subspace.

3. Periodic Boundary Conditions

To handle periodic boundary conditions we use a method that parallels that used by de Boor [1978, p.326] for periodic spline interpolation. We assume we are given a two point boundary value problem in the interval $I = [0,\pi]$, with periodic boundary conditions (these will depend on the order of the problem). We consider, as before, a uniform partition $\Pi_h(I)$ of the interval with mesh size $h = \frac{\pi}{N}$ and uniformly extend it to the rest of the real axis $\mathbf{R}$. Thus we arrive at a bi-infinite knot sequence $T = \{ t_i \}$ which we consider centered at the origin (i.e. $t_i = ih, \forall\, i \in \mathbf{Z}$). We now consider the linear space of π-periodic cubic splines over the knot sequence T and denote it by $S^3{}_P(T)$. We will need the following result from de Boor [1978, p.325] :

Theorem 3.1 : *Let $s(x)$ be a cubic spline over the knot sequence T. Then $s(x) \in S^3{}_P(T)$ ($s(x)$ is π-periodic) if and only if its expansion in terms of B-splines $\phi_i(x)$ is an N-periodic sequence, i.e.*

$$s(x) = \sum_i \alpha_i \phi_i(x), \textit{ with } \alpha_{i+N} = \alpha_i, \forall\, i.$$

where $\phi_i(x) = B(\frac{x}{h} - i)$ *is defined in Ap.I.§1.*

Consider now the problem of numerically solving $u''''(x) = g(x)$, $x \in I$ with periodic boundary conditions and where $g(x)$ is π-periodic. We seek a finite element discretization with ${S^3}_P(T)$ as the minimization space. To this end, we let T also be the sequence of finite element nodes for the discretization. Since there is only a fourth derivative term in the differential equation we need only assemble the bending matrix (see Ap.I.§1) for the sequence T. Doing this we obtain the linear system :

$$\sum_j b_{ij}\alpha_j = g_i, \forall i \in \mathbf{Z}\ , \tag{3.1}$$

where $B = [b_{ij}]$ is an infinite seven-band symmetric Toeplitz matrix and g_i is the N-periodic vector defined by :

$$g_i \equiv \int_{-\infty}^{\infty} \phi_i(x) g(x)\ dx = \int_{(i-2)h}^{(i+2)h} \phi_i(x) g(x)\ dx\ ,\ \forall\, i.$$

Now, using theorem 3.1 we can rewrite (3.1) as :

$$\sum_{j=0}^{N-1} a_{ij}\alpha_j = g_i,\ i = 0, 1, \cdots, N-1, \tag{3.2}$$

where we have defined

$$a_{ij} \equiv \sum_k b_{i\ (j-kN)}, \forall\, k \in \mathbf{Z}.$$

Now, since $B = [b_{ij}]$ is Toeplitz, we have that $b_{ij} = b_{rs}$ if $i - j = r - s$ and hence

$$a_{ij} = a_{rs} \text{ if } i - j = r - s\ (mod\ N). \tag{3.3}$$

Therefore $A = [a_{ij}]$ is a circulant matrix and thus the finite discrete version of the problem becomes :

$$\begin{cases} Au = f, \\ A \in \mathbf{M}_N(\mathbf{R}),\ A \textit{ circulant}, \end{cases} \tag{3.4}$$

where f is the vector

$$f_i \equiv g_i,\ i = 0, 1, \cdots ,N-1.$$

The above procedure can be extended to the non-constant coefficient case in which, of course, the assembly process has to be modified to include the coefficient functions. Nevertheless, from the necessary π-periodicity of the coefficients, it can be shown that we again obtain a $N-1 \times N-1$ matrix with the same sparsity pattern as before, though the circulant nature would be lost in general.

Thus, in the periodic constant coefficient case, all that needs to be calculated are the seven non-zero entries in an arbitrary row of A. These are easily obtained from the element matrices for the B-spline basis (see Ap.I.§1). Once they are known, the assembled matrices can be represented by the following molecules :

Mass :

$$m \equiv \frac{h}{2240} \left\{1,\ 120,\ 1191,\ 2416,\ 1191,\ 120,\ 1\right\}, \tag{3.5}$$

Stiffness :

$$s \equiv \frac{1}{160h} \left\{-3,\ -72,\ -45,\ 240,\ -45,\ -72,\ -3\right\} \tag{3.6}$$

and **Bending :**

$$b \equiv \frac{1}{8h^3} \left\{3,\ 0,\ -27,\ 48,\ -27,\ 0,\ 3\right\}. \tag{3.7}$$

The molecules are taken to act N-periodically (that is, modulo-N) over the set of nodes $\Pi_h(I)$.

APPENDIX III

Interpolation and Collection Operators

This appendix details the calculation of the interpolation and collection operators necessary for the construction of the algorithm FAPIN. This is done for the different finite element bases used for the construction of our model problems. Because of multigrid design considerations (see Ch.I.§4), we choose the collection operator to be, in each case, the transpose of the corresponding interpolation operator. Thus only the latter need be calculated.

We begin with the simpler case of periodic cubic splines and extend this to the cases with non-periodic boundary conditions. We then calculate these operators for a piecewise cubic Hermite basis and consider the three different scalings of this basis that were used in our experiments. We also give these operators for a piecewise linear basis ('roof' functions).

The operators given for both the Hermite and the linear bases apply to problems with natural boundary conditions. The corresponding operators for problems with essential homogeneous boundary conditions are easily obtained from them by deleting appropriate rows and columns.

In the last section we show how the operators for the one dimensional cases can be used to build the corresponding two dimensional ones, provided that the latter act on linear spaces which have a Kronecker product structure.

1. Periodic Cubic Splines

Let I denote the interval $[0,\pi]$ and consider two uniform partitions $\Pi_h(I)$, $\Pi_{2h}(I)$ of I, with mesh sizes $h = \frac{\pi}{N}$, $h' \equiv 2h$, respectively and where $N = 2^k$, for some $k \in \mathbf{Z}^+$. We now uniformly extend these partitions to the rest of the real axis $\mathbf{R}$. Thus we define two bi-infinite knot sequences $T^h = \{t^h{}_i\}$, $T^{2h} = \{t^{2h}{}_i\}$, which we consider centered at the origin (i.e. $t^h{}_i = ih$, $t^{2h}{}_i = i(2h), \forall\, i$). We now consider the linear spaces of π-periodic cubic splines over the knot sequences T^h, T^{2h} and denote them by $S^3{}_P(T^h)$, $S^3{}_P(T^{2h})$, respectively. Clearly $S^3{}_P(T^{2h}) \subset S^3{}_P(T^h)$, and thus $s^{2h}(x) \in S^3{}_P(T^{2h})$, implies $s^{2h}(x) \in S^3{}_P(T^h)$. Let $\mathbf{B}^h = \{\phi^h{}_i(x)\}$ and $\mathbf{B}^{2h} = \{\phi^{2h}{}_i(x)\}$ denote B-spline bases for $S^3{}_P(T^h)$, $S^3{}_P(T^{2h})$, respectively. With these definitions we have :

$$s^{2h}(x) = \sum_i \alpha^{2h}{}_i \phi^{2h}{}_i(x) = \sum_j \alpha^h{}_j \phi^h{}_j(x) \tag{1.1}$$

Thus, given $s^{2h}(x)$ in the basis $\mathbf{B}^{2h}$, its expression in the basis $\mathbf{B}^h$ can be obtained by expressing a generic $\phi^{2h}{}_i \in \mathbf{B}^{2h}$ in terms of $\mathbf{B}^h$. This is a simple interpolation problem whose solution is (from now to the end of this section we omit the independent variable in the functions) :

$$\phi^{2h}{}_i = \frac{1}{8}\phi^h{}_{2i-2} + \frac{1}{2}\phi^h{}_{2i-1} + \frac{3}{4}\phi^h{}_{2i} + \frac{1}{2}\phi^h{}_{2i+1} + \frac{1}{8}\phi^h{}_{2i+2}\,,\ \forall\, i, \tag{1.2}$$

and therefore :

$$\begin{aligned} s^{2h} &= \sum_i \alpha^{2h}{}_i \phi^{2h}{}_i \\ &= \sum_i \alpha^{2h}{}_i \left[\frac{1}{8}\phi^h{}_{2i-2} + \frac{1}{2}\phi^h{}_{2i-1} + \frac{3}{4}\phi^h{}_{2i} + \frac{1}{2}\phi^h{}_{2i+1} + \frac{1}{8}\phi^h{}_{2i+2}\right], \\ &= \sum_j \alpha^h{}_j \phi^h{}_j \end{aligned} \tag{1.3}$$

which, by collecting terms in $\phi^h{}_j$, implies $\forall\, i$:

$$\begin{cases} \alpha^h{}_{2i} = \frac{1}{8}\alpha^{2h}{}_{i-1} + \frac{3}{4}\alpha^{2h}{}_i + \frac{1}{8}\alpha^{2h}{}_{i+1} \\ \alpha^h{}_{2i+1} = \frac{1}{2}\alpha^{2h}{}_i + \frac{1}{2}\alpha^{2h}{}_{i+1} \end{cases} \tag{1.4}$$

Now, since $s^{2h} \in S^3{}_P(T^{2h})$, by Ap.II.T3.1 we know that α^{2h} is a $\frac{N}{2}$-periodic sequence. Thus we have :

$$\begin{aligned} \alpha^h{}_{2(i+\frac{N}{2})} &= \frac{1}{8}\alpha^{2h}{}_{(i+\frac{N}{2})-1} + \frac{3}{4}\alpha^{2h}{}_{(i+\frac{N}{2})} + \frac{1}{8}\alpha^{2h}{}_{(i+\frac{N}{2})+1} \\ &= \frac{1}{8}\alpha^{2h}{}_{i-1} + \frac{3}{4}\alpha^{2h}{}_{i} + \frac{1}{8}\alpha^{2h}{}_{i+1} \\ &= \alpha^h{}_{2i} \ , \ \forall i \end{aligned}$$

and

$$\begin{aligned} \alpha^h{}_{2(i+\frac{N}{2})+1} &= \frac{1}{2}\alpha^{2h}{}_{(i+\frac{N}{2})} + \frac{1}{2}\alpha^{2h}{}_{(i+\frac{N}{2})+1} \\ &= \frac{1}{2}\alpha^{2h}{}_{i} + \frac{1}{2}\alpha^{2h}{}_{i+1} \\ &= \alpha^h{}_{2i+1} \ , \ \forall i \end{aligned}$$

Therefore, the calculated coefficient sequence α^h is N-periodic, in agreement with the referenced theorem. Because of this periodicity we may consider only one period of the previous sequences, thus arriving at the following expressions :

$$\begin{cases} \alpha^h{}_{(2i) \bmod N} &= \frac{1}{8}\alpha^{2h}{}_{(i-1+\frac{N}{2}) \bmod \frac{N}{2}} + \frac{3}{4}\alpha^{2h}{}_{(i+\frac{N}{2}) \bmod \frac{N}{2}} + \frac{1}{8}\alpha^{2h}{}_{(i+1+\frac{N}{2}) \bmod \frac{N}{2}} \\ \alpha^h{}_{(2i+1) \bmod N} &= \frac{1}{2}\alpha^{2h}{}_{(i+\frac{N}{2}) \bmod \frac{N}{2}} + \frac{1}{2}\alpha^{2h}{}_{(i+1+\frac{N}{2}) \bmod \frac{N}{2}} \end{cases} \tag{1.5}$$

where the index i takes the values : $i = 0, 1, \ldots, \frac{N}{2}-1$.

Equations (1.5) define the desired interpolation operator from the space $S^3{}_P(T^{2h})$, into the space $\mathbf{S}^3{}_P(T^h)$. We denote it by $I_{2h}{}^h \in \mathbf{M}_{N\times\frac{N}{2}}$.

For example, for the case $N = 8$, the interpolation operator has the form (we omit the zero entries) :

$$I_{2h}{}^{h} = \begin{pmatrix} \frac{3}{4} & \frac{1}{8} & & \frac{1}{8} \\ \frac{1}{2} & \frac{1}{2} & & \\ \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & \\ & \frac{1}{2} & \frac{1}{2} & \\ & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} \\ & & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{8} & & \frac{1}{8} & \frac{3}{4} \\ \frac{1}{2} & & & \frac{1}{2} \end{pmatrix}$$

These operators will be used to build the required interpolation operators for our one and two dimensional periodic model problems. The latter will be constructed by means of a Kronecker product formulation (see §5 in this appendix).

2. Non-Periodic Cubic Splines

In this section we calculate the interpolation and collection operators needed in the algorithm FAPIN when applied to the solution of our one-dimensional non-periodic model problems (see Ch.II.§1). We follow a method similar to that used to derive the element matrices (see Ap.I.§1). That is, we begin with the case of free boundaries and proceed to 'add' the required essential boundary conditions by means of linear combinations of basis functions.

Following the notation of the previous section, let I denote the interval $[0,\pi]$ and consider two uniform partitions $\Pi_h(I)$, $\Pi_{2h}(I)$ of I, with mesh sizes $h = \frac{\pi}{N}$, $h' \equiv 2h$, respectively and where $N = 2^k$, for some $k \in \mathbf{Z}^+$. We now consider the linear spaces of $C^2(I)$ cubic splines over the these partitions and denote them by $S_3(\Pi_h)$, $S_3(\Pi_{2h})$, respectively. Using the same superscript notation as in §1, bases for these spaces are given by :

$$\mathbf{B}^h{}_S \equiv \{\phi^h{}_i\}_{i=-1}^{N+1} \tag{2.1}$$

$$\mathbf{B}^{2h}{}_S \equiv \{\phi^{2h}{}_i\}_{i=-1}^{\frac{N}{2}+1} \tag{2.2}$$

where, as usual,

$$\phi_i = B(\frac{x}{h} - i)$$

and $B(t)$ is the cubic B-spline function with support $[-2,2]$ and such that $B(0) = 1$:

$$B(t) = \begin{cases} B_{-2}(t) \equiv \frac{1}{4}(t+2)^3 & \text{if } t \in [-2,-1] \\ B_{-1}(t) \equiv \frac{1}{4} + \frac{3}{4}(t+1) + \frac{3}{4}(t+1)^2 - \frac{3}{4}(t+1)^3 & \text{if } t \in [-1,0] \\ B_1(t) \equiv \frac{1}{4} + \frac{3}{4}(1-t) + \frac{3}{4}(1-t)^2 - \frac{3}{4}(1-t)^3 & \text{if } t \in [0,1] \\ B_2(t) \equiv \frac{1}{4}(2-t)^3 & \text{if } t \in [1,2] \end{cases} \tag{2.3}$$

We have $S_3(\Pi_{2h}) \subset S_3(\Pi_h)$, and thus $v^{2h}(x) \in S_3(\Pi_{2h})$ implies $v^{2h}(x) \in S_3(\Pi_h)$. Therefore, we may write :

$$v^{2h}(x) = \sum_{i=-1}^{\frac{N}{2}+1} q^{2h}{}_i \phi^{2h}{}_i(x) = \sum_{j=-1}^{N+1} q^h{}_j \phi^h{}_j(x) \tag{2.4}$$

Thus, given $v^{2h}(x)$ in the basis $\mathbf{B}^{2h}{}_S$, its expression in the basis $\mathbf{B}^h{}_S$ can be obtained by expressing a generic $\phi^{2h}{}_i \in \mathbf{B}^{2h}{}_S$ in terms of $\mathbf{B}^h{}_S$. In contrast to the previous section, however, this does not reduce to a single interpolation problem for all values of N since we now must consider the boundaries. Thus we separate the calculations by different values of N (we again omit the independent variable in our notation).

(i) $N = 2$:

We must solve four interpolation problems which, by symmetry, reduce to two :

1: Express the B_2 part of $\phi^{2h}{}_{-1}$ (see (2.3)) in terms of $\mathbf{B}^h{}_S$.

2: Express the B_2 ,B_1 parts of $\phi^{2h}{}_0$ (see (2.3)) in terms of $\mathbf{B}^h{}_S$.

Solving these interpolation problems we obtain :

$$\begin{cases} \begin{array}{lllllll} \phi^{2h}{}_{-1} = & \frac{1}{2}\phi^h{}_{-1} + & \frac{1}{8}\phi^h{}_0 & & & \\ \phi^{2h}{}_0 = & \frac{1}{2}\phi^h{}_{-1} + & \frac{3}{4}\phi^h{}_0 + & \frac{1}{2}\phi^h{}_1 + & \frac{1}{8}\phi^h{}_2 & \\ \phi^{2h}{}_1 = & & \frac{1}{8}\phi^h{}_0 + & \frac{1}{2}\phi^h{}_1 + & \frac{3}{4}\phi^h{}_2 + & \frac{1}{2}\phi^h{}_3 \\ \phi^{2h}{}_2 = & & & & \frac{1}{8}\phi^h{}_2 + & \frac{1}{2}\phi^h{}_3 \end{array} \end{cases} \tag{2.5}$$

We will use the notation $I_{\frac{N}{2}}{}^N$ for the calculated interpolation operators. With this notation, we obtain from (2.5) the desired $I_1{}^2$ which has the form :

$$I_1{}^2 = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & & \\ \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & \\ & \frac{1}{2} & \frac{1}{2} & \\ & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} \\ & & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

Proceeding in a fashion similar to that of (i) we obtain the interpolation operators for the remaining values of N. For brevity we only give the final operators :

(ii) $N = 4$:

$$I_2^{\,4} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & & \\ & \frac{1}{2} & \frac{1}{2} & & \\ & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & \\ & & \frac{1}{2} & \frac{1}{2} & \\ & & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} \\ & & & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

(iii) $N = 2^k$, $k \geq 3$:

$$I_{2^{k-1}}^{\;2^k} = \begin{pmatrix} I_2^{\,4} & & & & \\ & I_r & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & \cdot \\ & & & & & I_r \end{pmatrix} , \quad I_r = \begin{pmatrix} \frac{1}{8} & \frac{3}{4} & \frac{1}{8} \\ 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

where the 2×3 repeated block I_r appears $\dfrac{N-5}{2}$ times, with its first column shifted to the right by one position with respect to the first column of the previous block.

Once the operators for the case of natural boundary conditions have been obtained, we proceed to calculate the interpolation operators between cubic spline spaces whose vectors satisfy the essential boundary condition $v^h(0) = 0$. We use a notation consistent with that of Ap.II.§1 and the one used so far in this section. Thus we let $S^3{}_E(\Pi_h)$, $S^3{}_E(\Pi_{2h})$ denote the linear spaces of cubic splines over the partitions Π_h, Π_{2h} whose elements vanish at the origin :

$$S^3{}_E(\Pi_h) = \left\{ v^h(x) \in S^3(\Pi_h) \mid \ v^h(0) = 0 \right\},$$

$$S^3{}_E(\Pi_{2h}) = \left\{ v^{2h}(x) \in S^3(\Pi_{2h}) \mid \ v^{2h}(0) = 0 \right\}.$$

We recall that bases for these spaces are given, respectively, by (cf. Ap.II.§1) :

$$\mathbf{B}^h{}_{SE} \equiv \{\psi^h{}_i\}_{i=0}^{N+1} \tag{2.6}$$

$$\mathbf{B}^{2h}{}_{SE} \equiv \{\psi^{2h}{}_i\}_{i=0}^{\frac{N}{2}+1} \tag{2.7}$$

where :

$$\begin{aligned} \psi^h{}_0 &= \frac{16}{15}\cdot\phi^h{}_{-1} - \frac{4}{15}\,\phi^h{}_0, \\ \psi^h{}_1 &= -\frac{4}{15}\,\phi^h{}_0 + \frac{16}{15}\,\phi^h{}_1. \\ \psi^h{}_i(x) &= \phi^h{}_i(x), \ \ 2 \le i \le N{+}1, \end{aligned} \tag{2.8}$$

and

$$\begin{aligned} \psi^{2h}{}_0 &= \frac{16}{15}\,\phi^{2h}{}_{-1} - \frac{4}{15}\,\phi^{2h}{}_0, \\ \psi^{2h}{}_1 &= -\frac{4}{15}\,\phi^{2h}{}_0 + \frac{16}{15}\,\phi^{2h}{}_1. \\ \psi^{2h}{}_i(x) &= \phi^{2h}{}_i(x), \ \ 2 \le i \le \frac{N}{2}{+}1, \end{aligned} \tag{2.9}$$

From the previous definitions we observe that only the two first basis functions of the two new bases have been redefined in terms of linear combinations of old basis functions. Thus we only need to recalculate the change of basis for them, since the rest will transform as the ϕ's did. We again separate the calculations by the different values of N.

(i) $N = 2$:

We must perform the following calculations :

1: Express $\psi^{2h}{}_0$ (see (2.9)) in terms of $\mathbf{B}^h{}_{SE}$.

2: Express $\psi^{2h}{}_1$ (see (2.9)) in terms of $\mathbf{B}^h{}_{SE}$.

We first solve problem 1. By definition of $\psi^{2h}{}_0$ we have :

$$\begin{aligned}\psi^{2h}{}_0 &= \frac{16}{15}\,\phi^{2h}{}_{-1} - \frac{4}{15}\,\phi^{2h}{}_0 \\ &= \frac{16}{15}(\frac{1}{2}\phi^h{}_{-1} + \frac{1}{8}\phi^h{}_0) - \frac{4}{15}(\frac{1}{2}\phi^h{}_{-1} + \frac{3}{4}\phi^h{}_0 + \frac{1}{2}\phi^h{}_1 + \frac{1}{8}\phi^h{}_2) \\ &= \frac{16}{15}\phi^h{}_{-1} - \frac{1}{15}\phi^h{}_0 - \frac{2}{15}\phi^h{}_1 - \frac{1}{30}\phi^h{}_2 \,. \end{aligned} \tag{2.10}$$

Since the only functions involved in the right hand side of (2.10) are $\phi^h{}_i$, $i = -1, 0, 1, 2$, and neither $\psi^h{}_0$ nor $\psi^h{}_1$ depend on $\phi^h{}_2$, we can equate $\psi^{2h}{}_0$ to an undetermined linear combination of $\psi^h{}_i$, $i = 0, 1$ plus a term in $\phi^h{}_2$. Doing this and using the definitions of $\psi^h{}_i$, $i = 0, 1$ we obtain :

$$\begin{aligned}\psi^{2h}{}_0 &= a\psi^h{}_0 + b\psi^h{}_1 - \frac{1}{30}\phi^h{}_2 \,. \\ &= a(\frac{16}{15}\,\phi^h{}_{-1} - \frac{4}{15}\,\phi^h{}_0) + b(-\frac{4}{15}\,\phi^h{}_0 + \frac{16}{15}\,\phi^h{}_1) - \frac{1}{30}\phi^h{}_2 \,. \end{aligned} \tag{2.11}$$

(2.10) and (2.11) to a linear system of equations from which we obtain the desired coefficients :

$$a = \frac{3}{8},\ b = \frac{-1}{8}$$

Thus, from the previous calculations, we have :

$$\psi^{2h}{}_0 = \frac{3}{8}\psi^h{}_0 - \frac{1}{8}\psi^h{}_1 - \frac{1}{30}\psi^h{}_2 \,, \tag{2.12}$$

which constitutes the solution to problem 1.

In a similar fashion, the solution to problem 2 can be obtained. This is given by :

$$\psi^{2h}{}_1 = \frac{-1}{8}\psi^h{}_0 + \frac{3}{8}\psi^h{}_1 + \frac{23}{30}\psi^h{}_2 + \frac{8}{15}\psi^h{}_3 \,. \tag{2.13}$$

Now, using (2.12) and (2.13) we can give the change of basis :

$$\begin{cases} \psi^{2h}{}_0 = & \frac{3}{8}\psi^h{}_0 - & \frac{1}{8}\psi^h{}_1 - & \frac{1}{30}\psi^h{}_2 & \\ \psi^{2h}{}_1 = & \frac{-1}{8}\psi^h{}_0 + & \frac{3}{8}\psi^h{}_1 + & \frac{23}{30}\psi^h{}_2 + & \frac{8}{15}\psi^h{}_3 \\ \psi^{2h}{}_2 = & & & \frac{1}{8}\psi^h{}_2 + & \frac{1}{2}\psi^h{}_3 \end{cases} \tag{2.14}$$

Thus the desired interpolation operator is given by :

$$I_1^2 = \begin{pmatrix} \frac{3}{8} & \frac{-1}{8} & \\ \frac{-1}{8} & \frac{3}{8} & \\ \frac{-1}{30} & \frac{23}{30} & \frac{1}{8} \\ & \frac{8}{15} & \frac{1}{2} \end{pmatrix}$$

Proceeding in an analogous fashion to that in (i), we obtain the interpolation operators for the remaining values of N. Once again, for brevity we only give the final operators. These have the form :

(ii) $N = 4$:

$$I_2^4 = \begin{pmatrix} \frac{3}{8} & \frac{-1}{8} & & \\ \frac{-1}{8} & \frac{3}{8} & & \\ \frac{-1}{30} & \frac{23}{30} & \frac{1}{8} & \\ & \frac{8}{15} & \frac{1}{2} & \\ & \frac{2}{15} & \frac{3}{4} & \frac{1}{8} \\ & & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

(iii) $N = 2^k$, $k \geq 3$:

$$I_{2^{k-1}}^{2^k} = \begin{pmatrix} I_2^4 & & & & \\ & I_r & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & \cdot \\ & & & & & I_r \end{pmatrix} , \quad I_r = \begin{pmatrix} \frac{1}{8} & \frac{3}{4} & \frac{1}{8} \\ 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

where the 2x3 repeated block I_r appears $\frac{N-5}{2}$ times, with its first column shifted to the right by one position with respect to the first column of the previous block.

Next we calculate the interpolation operators between cubic spline spaces whose functions satisfy two (essential) boundary conditions, namely : $v^h(0) = 0$ and $\frac{d}{dx} v^h(0) = 0$. Our notation will again be consistent with that of Ap.II.§2 and the one used so far in this section. Thus, let $S^3{}_{E1}(\Pi_h)$, $S^3{}_{E1}(\Pi_{2h})$ denote the following linear spaces of cubic splines over the partitions Π_h, Π_{2h} :

$$S^3{}_{E1}(\Pi_h) = \left\{ v^h(x) \in S^3{}_E(\Pi_h) \mid \frac{d}{dx} v^h(0) = 0 \right\}.$$

$$S^3{}_{E1}(\Pi_{2h}) = \left\{ v^{2h}(x) \in S^3{}_E(\Pi_{2h}) \mid \frac{d}{dx} v^{2h}(0) = 0 \right\}.$$

We recall that bases for these spaces are given, respectively, by (see Ap.II.§2) :

$$\mathbf{B}^h{}_{SE1} \equiv \{\xi^h{}_i\}_{i=0}^{N+1} \tag{2.15}$$

$$\mathbf{B}^{2h}{}_{SE1} \equiv \{\xi^{2h}{}_i\}_{i=0}^{\frac{N}{2}+1} \tag{2.16}$$

where :

$$\xi^h{}_1 = \frac{15}{14} (\psi^h{}_0 + \psi^h{}_1). \tag{2.17}$$

$$\xi^h{}_i(x) = \psi^h{}_i(x), \;\; 2 \leq i \leq N{+}1,$$

and

$$\xi^{2h}{}_1 = \frac{15}{14} (\psi^{2h}{}_0 + \psi^{2h}{}_1). \tag{2.18}$$

$$\xi^{2h}{}_i(x) = \psi^{2h}{}_i(x), \;\; 2 \leq i \leq N{+}1,$$

From the previous definitions we observe that only the first basis function of the two new bases has been redefined in terms of a linear combination of ψ basis functions. Thus we only need to recalculate one of the equations in the change of basis. We again separate the calculations by the different values of N.

(i) $N = 2$:

We must express $\xi^{2h}{}_1$ (see (2.18)) in terms of $\mathbf{B}^h{}_{SE1}$.

By definition of $\xi^{2h}{}_1$ we have :

$$
\begin{aligned}
\xi^{2h}{}_1 &= \frac{15}{14}\,(\psi^{2h}{}_0 + \psi^{2h}{}_1). \\
&= \frac{15}{14}\left\{(\frac{3}{8}\psi^h{}_0 - \frac{1}{8}\psi^h{}_1 - \frac{1}{30}\psi^h{}_2) + (\frac{-1}{8}\psi^h{}_0 + \frac{3}{8}\psi^h{}_1 + \frac{23}{30}\psi^h{}_2 + \frac{8}{15}\psi^h{}_3)\right\} \\
&= \frac{15}{56}(\psi^h{}_0 + \psi^h{}_1) + \frac{11}{14}\psi^h{}_2 + \frac{8}{14}\psi^h{}_3\,. \\
&= \frac{1}{4}\xi^h{}_1 + \frac{11}{14}\xi^h{}_2 + \frac{8}{14}\xi^h{}_3\,.
\end{aligned}
\tag{2.19}
$$

Thus, from (2.19) we see that the change of basis is given in this case by :

$$
\left\{
\begin{array}{llll}
\psi^{2h}{}_1 = & \frac{1}{4}\psi^h{}_1 + & \frac{11}{14}\psi^h{}_2 + & \frac{8}{14}\psi^h{}_3 \\
\psi^{2h}{}_2 = & & \frac{1}{8}\psi^h{}_2 + & \frac{1}{2}\psi^h{}_3
\end{array}
\right.
\tag{2.20}
$$

and the interpolation operator is given by :

$$
I_1{}^2 = \begin{pmatrix}
\frac{1}{4} & \\
\frac{11}{14} & \frac{1}{8} \\
\frac{8}{14} & \frac{1}{2}
\end{pmatrix}
$$

In an analogous fashion we obtain the interpolation operators for the remaining values of N. We again only give the final operators :

(ii) $N = 4$:

$$
I_2{}^4 = \begin{pmatrix}
\frac{1}{4} & & \\
\frac{11}{14} & \frac{1}{8} & \\
\frac{8}{14} & \frac{1}{2} & \\
\frac{2}{14} & \frac{3}{4} & \frac{1}{8} \\
 & \frac{1}{2} & \frac{1}{2}
\end{pmatrix}
$$

(iii) $N = 2^k$, $k \geq 3$:

$$I_{2^{k-1}}{}^{2^k} = \begin{pmatrix} I_2{}^4 & & & & \\ & I_r & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & \cdot \\ & & & & & I_r \end{pmatrix}, \quad I_r = \begin{pmatrix} \frac{1}{8} & \frac{3}{4} & \frac{1}{8} \\ 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

where the 2×3 repeated block I_r appears $\frac{N-5}{2}$ times, with its first column shifted to the right by one position with respect to the first column of the previous block.

3. Piecewise Cubic Hermites

In this section we calculate the interpolation operators defined between spaces of $\mathbf{C}^1$ piecewise cubic polynomials. These are used to build the corresponding operators for spaces of piecewise bicubic polynomials, using the Kronecker product nature of their bases. We derive the operators for the case with natural boundary conditions and the three different scalings of the basis which were considered in our experiments (see Ap.I.§2,3&4). Essential homogeneous boundary conditions are easily imposed by deleting appropriate rows and columns in the given operators.

Following the notation of the previous sections, we let I denote the interval $[0,\pi]$ and consider two uniform partitions $\Pi_h(I)$, $\Pi_{2h}(I)$ of I, with mesh sizes $h = \frac{\pi}{N}$, $h' \equiv 2h$, respectively and where $N = 2^k$, for some $k \in \mathbf{Z}^+$. We now consider the linear spaces of $\mathbf{C}^1(I)$ piecewise cubic Hermite functions over the above partitions and denote them by $S_H(\Pi_h)$, $S_H(\Pi_{2h})$, respectively. Corresponding bases for these spaces are given by :

$$\mathbf{B}^h{}_H \equiv \{\psi^h{}_i, \omega^h{}_i\}_{i=0}^{N} \tag{3.1}$$

$$\mathbf{B}^{2h}{}_H \equiv \{\psi^{2h}{}_i, \omega^{2h}{}_i\}_{i=0}^{\frac{N}{2}} \tag{3.2}$$

where (see Ap.I.§2) :

$$\psi_i = P(\frac{x}{h} - i)$$
$$\omega_i = h \ \ W(\frac{x}{h} - i)$$

and $P(t)$, $W(t)$ are the Hermite Cubic basis functions with support $[-1,1]$ described in Ap.I.§2.

Clearly $S_H(\Pi_{2h}) \subset S_H(\Pi_h)$, and thus $v^{2h}(x) \in S_H(\Pi_{2h})$ implies $v^{2h}(x) \in S_H(\Pi_h)$. Therefore, given $v^{2h}(x)$ in the basis $\mathbf{B}^{2h}{}_H$, its expression in the basis $\mathbf{B}^h{}_H$ can be obtained by expressing generic $\psi^{2h}{}_i$, $\omega^{2h}{}_i \in \mathbf{B}^{2h}{}_H$ in terms of $\mathbf{B}^h{}_H$. These are again interpolation problems. They are easier to solve than the ones for the B-splines case since , now, the basis functions can be considered one subinterval at a time.

Let $I_1 = [j2h, (j+1)2h]$ denote an arbitrary but fixed subinterval in the partition $\Pi_{2h}(I)$. Then, letting $I_1 = I_2 \cup I_3$, where I_2, $I_3 \in \Pi_h(I)$ and recalling that $h' = 2h$, it can be shown that :

$$\begin{cases} \psi^{2h}{}_j = \psi^h{}_{2j} + \frac{1}{2}\psi^h{}_{2j+1} - \frac{3}{2h'}\omega^h{}_{2j+1} \,, \ \text{in } I_1 \\ \psi^{2h}{}_{j+1} = \frac{3}{2h'}\psi^h{}_{2(j+1)-1} + \frac{1}{2}\psi^h{}_{2(j+1)-1} + \omega^h{}_{2(j+1)} \,, \ \text{in } I_1 \end{cases} \tag{3.3}$$

and

$$\begin{cases} \omega^{2h}{}_j = \omega^h{}_{2j} + \frac{h'}{8}\psi^h{}_{2j+1} - \frac{1}{4}\omega^h{}_{2j+1} \,, \ \text{in } I_1 \\ \omega^{2h}{}_{j+1} = -\frac{h'}{8}\psi^h{}_{2(j+1)-1} - \frac{1}{4}\omega^h{}_{2(j+1)-1} + \omega^h{}_{2(j+1)} \,, \ \text{in } I_1 \end{cases} \tag{3.4}$$

Thus, the basis functions as defined in (3.2) transform according to :

$$\begin{cases} \psi^{2h}{}_j = \frac{1}{2}\psi^h{}_{2j-1} + \frac{3}{2h'}\omega^h{}_{2j-1} + \psi^h{}_{2j} + \frac{1}{2}\psi^h{}_{2j+1} - \frac{3}{2h'}\omega^h{}_{2j+1} \\ \omega^{2h}{}_j = -\frac{h'}{8}\psi^h{}_{2j-1} - \frac{1}{4}\omega^h{}_{2j-1} + \omega^h{}_{2j} + \frac{h'}{8}\psi^h{}_{2j+1} - \frac{1}{4}\omega^h{}_{2j+1} \end{cases} \tag{3.5}$$

From equations (3.3), (3.4) and (3.5) we can now calculate the interpolation operators. Recalling that h' denotes the mesh size of the coarse partition $\Pi_{2h}(I)$ and defining the blocks :

$$I_u = \begin{pmatrix} 1 & 0 & \frac{1}{2} & \frac{-3}{2h'} \\ 0 & 1 & \frac{h'}{8} & \frac{-1}{4} \end{pmatrix}, \quad I_b = \begin{pmatrix} \frac{1}{2} & \frac{3}{2h'} & 1 & 0 \\ \frac{-h'}{8} & \frac{-1}{4} & 0 & 1 \end{pmatrix}$$

$$I_r = \begin{pmatrix} \frac{1}{2} & \frac{3}{2h'} & 1 & 0 & \frac{1}{2} & \frac{-3}{2h'} \\ \frac{-h'}{8} & \frac{-1}{4} & 0 & 1 & \frac{h'}{8} & \frac{-1}{4} \end{pmatrix},$$

the desired operators (transposed for convenience) are given by :

(i) $N = 2$:

$$\left(I_1{}^2\right)^t = \begin{pmatrix} I_u & \\ & I_b \end{pmatrix}$$

where the block $I_b I$ is shifted by two columns to the right with respect to the first column of the block I_u.

(ii) $N = 2^k$, $k \geq 2$:

$$\left(I_{2^{k-1}}{}^{2^k}\right)^t = \begin{pmatrix} I_u & & & & & & \\ & I_r & & & & & \\ & & \cdot & & & & \\ & & & \cdot & & & \\ & & & & \cdot & & \\ & & & & & I_r & \\ & & & & & & I_b \end{pmatrix}$$

where the repeated block I_r appears $\dfrac{N-2}{2}$ times. Its first column is shifted by two columns with respect to the first column of I_u and the first column of I_b is shifted by four columns with respect to the first column of the last I_r block. In the repetition part the shift is four columns.

We remark that, for the previous basis, the entries in the interpolation and collection operators depend explicitly on the coarse grid mesh size (as was the case with the element matrices (see Ap.I.§2)).

Next, we consider the two rescalings of the previous basis that were used in this thesis. These are described extensively in Ap.I.§3&4 . For brevity, we only recall them here and give the corresponding operators. These are obtained in a completely analogous manner to that used in the above calculations.

Scaling 1 :

This scaling uses $h\psi$ instead of the functions ψ as defined in equations (3.1), (3.2). Thus the new bases become :

$$\mathbf{B}^h{}_{H1} \equiv \{\eta^h{}_i, \omega^h{}_i\}_{i=0}^{N} \tag{3.6}$$

$$\mathbf{B}^{2h}{}_{H1} \equiv \{\eta^{2h}{}_i, \omega^{2h}{}_i\}_{i=0}^{\frac{N}{2}} \tag{3.7}$$

where (see Ap.I.§2) :

$$\eta_i = h\ \psi_i = h\ P(\frac{x}{h} - i)$$

$$\omega_i = h\ W(\frac{x}{h} - i)$$

In terms of (3.6), (3.7) the new interpolation operators have the same block structure as for $\mathbf{B}_{H1}$, but with redefined blocks of the form :

$$I_u = \begin{pmatrix} 2 & 0 & 1 & \frac{-3}{2} \\ 0 & 1 & \frac{1}{4} & \frac{-1}{4} \end{pmatrix}, \quad I_b = \begin{pmatrix} 1 & \frac{3}{2} & 2 & 0 \\ \frac{-1}{4} & \frac{-1}{4} & 0 & 1 \end{pmatrix}$$

$$I_r = \begin{pmatrix} 1 & \frac{3}{2} & 2 & 0 & 1 & \frac{-3}{2} \\ \frac{-1}{4} & \frac{-1}{4} & 0 & 1 & \frac{1}{4} & \frac{-1}{4} \end{pmatrix}$$

Scaling 2 :

This scaling uses the basis functions defined in equations (3.1), (3.2) but normalized to have unit $\mathbf{L}^2(I)$-norm (for simplicity, the same scaling factor is used for the basis functions at the boundaries). Thus the new bases become :

$$\mathbf{B}^h{}_{H2} \equiv \{\chi^h{}_i, \theta^h{}_i\}_{i=0}^{N} \tag{3.8}$$

$$\mathbf{B}^{2h}{}_{H2} \equiv \{\chi^{2h}{}_i, \theta^{2h}{}_i\}_{i=0}^{\frac{N}{2}} \tag{3.9}$$

where (see Ap.I.§3) :

$$\chi_i = n_\psi\, \psi_i = n_\psi\, P(\frac{x}{h} - i)$$

$$\theta_i = n_\omega\, \omega_i = n_\omega\, h\ \ W(\frac{x}{h} - i)$$

and

$$\frac{1}{n_\psi} = \left[\int\limits_{(i-1)h}^{(i+1)h} \psi_i^2(x)\ dx\right]^{1/2} = \left[\frac{26}{35}h\right]^{1/2},$$

$$\frac{1}{n_\omega} = \left[\int\limits_{(i-1)h}^{(i+1)h} \omega_i^2(x)\ dx\right]^{1/2} = \left[\frac{2}{105}h^3\right]^{1/2},$$

where we have used a generic notation to embrace both the $\mathbf{B}^h$ and the $\mathbf{B}^{2h}$ bases. Once again, the new interpolation operators for (3.8), (3.9) have the same block structure as before, but with redefined blocks. Letting a, b and c stand for :

$$a = \frac{\sqrt{2}}{2}, \quad b = \frac{\sqrt{78}}{16} \quad c = -\frac{3\sqrt{78}}{2\ 156},$$

these blocks have the form :

$$I_u = \begin{bmatrix} a & 0 & \frac{a}{2} & c \\ 0 & \frac{a}{2} & b & \frac{-a}{8} \end{bmatrix}, \quad I_b = \begin{bmatrix} \frac{a}{2} & -c & a & 0 \\ -b & \frac{-a}{8} & 0 & \frac{a}{2} \end{bmatrix}$$

$$I_r = \begin{bmatrix} \frac{a}{2} & -c & a & 0 & \frac{a}{2} & c \\ -b & \frac{-a}{8} & 0 & \frac{a}{2} & b & \frac{-a}{8} \end{bmatrix}.$$

As noted in Ap.I.§3&4, the above scalings of the bases provided convergence of the algorithm FAPIN for the one dimensional tests that we conducted. Thus these scaled versions were used to build the two dimensional operators needed for the solution of the two dimensional non-periodic model problems considered in this thesis. The two dimensional operators are easily built from Kronecker products of their one dimensional counterparts. We examine this process in §5 of this appendix.

4. Piecewise Linear Basis

In this section we calculate the interpolation operators defined between spaces of $\mathbf{C}^0$ piecewise linear polynomials. These are used to build the corresponding operators for spaces of piecewise bilinear polynomials, using the Kronecker product nature of their bases. The interpolation operators are obtained in a completely analogous manner to that of the Hermite bases in the previous section. Again we restrict ourselves to the case with natural boundary conditions, since essential homogeneous boundary conditions are easily imposed by deleting appropriate rows and columns in the given operators.

Following the notation of the previous sections, we let I denote the interval $[0,\pi]$ and consider two uniform partitions $\Pi_h(I)$, $\Pi_{2h}(I)$ of I, with mesh sizes $h = \frac{\pi}{N}$, $h' \equiv 2h$, respectively and where $N = 2^k$, for some $k \in \mathbf{Z}^+$. We now consider the linear spaces of $C^0(I)$ piecewise linear functions over the above partitions and denote them by $S_L(\Pi_h)$, $S_L(\Pi_{2h})$, respectively. Corresponding bases for these spaces are given by :

$$\mathbf{B}^h{}_L \equiv \{\lambda^h{}_i\}_{i=0}^{N} \tag{4.1}$$

$$\mathbf{B}^{2h}{}_L \equiv \{\lambda^{2h}{}_i\}_{i=0}^{\frac{N}{2}} \tag{4.2}$$

where (see Strang & Fix [1973, p.27])

$$\lambda_i = L(\frac{x}{h} - i)$$

and $L(t)$ is the 'roof' function with support $[-1,1]$ such that $L(0) = 1$:

$$L(t) = \begin{cases} (t+1) & \text{if } t \in [-1,0] \\ (1-t) & \text{if } t \in [0,1] \end{cases}$$

Clearly $S_L(\Pi_{2h}) \subset S_L(\Pi_h)$, and thus $v^{2h}(x) \in S_L(\Pi_{2h})$ implies $v^{2h}(x) \in S_L(\Pi_h)$. Therefore, given $v^{2h}(x)$ in the basis $\mathbf{B}^{2h}{}_L$, its expression in the basis $\mathbf{B}^h{}_L$ can be obtained by expressing generic $\lambda^{2h}{}_i \in \mathbf{B}^{2h}{}_L$ in terms of $\mathbf{B}^h{}_L$. This is again an interpolation problem.

Let $I_1 = [j2h,(j+1)2h]$ denote an arbitrary but fixed subinterval in the partition $\Pi_{2h}(I)$. Then, letting $I_1 = I_2 \cup I_3$, where I_2, $I_3 \in \Pi_h(I)$ and recalling that $h' = 2h$, the solution to the above interpolation problem can be shown to be :

$$\begin{cases} \lambda^{2h}{}_j = \lambda^h{}_{2j} + \frac{1}{2}\lambda^h{}_{2j+1} \text{ , in } I_1 \\ \lambda^{2h}{}_{j+1} = \frac{1}{2}\lambda^h{}_{2(j+1)-1} + \lambda^h{}_{2(j+1)} \text{ , in } I_1 \end{cases} \tag{4.3}$$

Thus, the basis functions as defined in (4.2) transform according to :

$$\lambda^{2h}{}_j = \frac{1}{2}\lambda^h{}_{2j-1} + \lambda^h{}_{2j} + \frac{1}{2}\lambda^h{}_{2j+1} \tag{3.4}$$

From equations (4.3) and (4.4) the desired interpolation operators can now be calculated. Defining the blocks :

$$I_u = \begin{bmatrix} 1 & \frac{1}{2} \end{bmatrix} , \quad I_b = \begin{bmatrix} \frac{1}{2} & 1 \end{bmatrix} , \quad I_r = \begin{bmatrix} \frac{1}{2} & 1 & \frac{1}{2} \end{bmatrix}$$

these operators (transposed for convenience) are given by :

(i) $N = 2$:

$$\left(I_1{}^2\right)^t = \begin{pmatrix} I_u & \\ & I_b \end{pmatrix}$$

where the first column of the block I_b is shifted by one column to the right of the first column of the block I_u.

(ii) $N = 2^k$, $k \geq 2$:

$$\left(I_{2^{k-1}}{}^{2^k}\right)^t = \begin{pmatrix} I_u & & & & & \\ & I_r & & & & \\ & & \cdot & & & \\ & & & \cdot & & \\ & & & & \cdot & \\ & & & & & I_r \\ & & & & & & I_b \end{pmatrix}$$

where the repeated block I_r appears $N-1$ times. Its first column is shifted by one column with respect to the first column of I_u and the first column of I_b is shifted by two columns with respect to the first column of the last I_r block. In the repetition part the shift is two columns.

5. Kronecker Product Formulation of Interpolation and Collection Operators

In this section we show how interpolation operators between linear spaces which are Kronecker products of two other linear spaces (see Jacobson [1953, p.208]), can be expressed as Kronecker products of the interpolation operators defined between the spaces whose product is taken. This is the case for two dimensional bicubic spline, piecewise bicubic Hermite and piecewise bilinear spaces defined on rectangular domains. We follow the notation used in the previous sections with some adaptations to our present needs.

Let $\mathbf{X}^h$, $\mathbf{X}^{2h}$, $\mathbf{Y}^h$ and $\mathbf{Y}^{2h}$ be finite dimensional linear spaces such that $\mathbf{X}^{2h} \subset \mathbf{X}^h$ and $\mathbf{Y}^{2h} \subset \mathbf{Y}^h$. Clearly $\mathbf{X}^{2h} \otimes \mathbf{Y}^{2h} \subset \mathbf{X}^h \otimes \mathbf{Y}^h$, where the symbol $\otimes$ denotes Kronecker product. We want to express an arbitrary element $v \in \mathbf{X}^{2h} \otimes \mathbf{Y}^{2h}$ in terms of a basis for $\mathbf{X}^h \otimes \mathbf{Y}^h$. This is a well defined interpolation problem. To solve it we make use of the Kronecker product structure.

Let $\phi^h{}_i(x)$, $\phi^{2h}{}_i(x)$, $\phi^h{}_i(y)$ and $\phi^{2h}{}_i(y)$ denote generic members of a basis for $\mathbf{X}^h$, $\mathbf{X}^{2h}$, $\mathbf{Y}^h$ and $\mathbf{Y}^{2h}$, respectively (the range of the indices is the dimension of the corresponding spaces and is omited for simplicity).

From the previous definitions it is clear that generic members of a basis for $\mathbf{X}^{2h} \otimes \mathbf{Y}^{2h}$ and $\mathbf{X}^h \otimes \mathbf{Y}^h$ will have the form :

$$\begin{cases} \Phi^h{}_{i,j} = \phi^h{}_i(x)\phi^h{}_j(y) \\ \Phi^{2h}{}_{k,l} = \phi^{2h}{}_k(x)\phi^{2h}{}_l(y) \ , \end{cases} \tag{4.1}$$

respectively (again the range of the indices is clear and thus it is omited). Since $\phi^{2h}{}_i(x) \in \mathbf{X}^h$ and $\phi^{2h}{}_j(y) \in \mathbf{Y}^h$ by hypothesis, we can write :

$$\begin{cases} \phi^{2h}{}_i(x) = \sum\limits_m a^i{}_m \phi^h{}_m(x) \\ \phi^{2h}{}_j(y) = \sum\limits_n b^j{}_n \phi^h{}_n(y) \end{cases}$$

Thus, we have :

$$\begin{aligned} \Phi^{2h}{}_{i,j} &= \sum_{m,n} a^i{}_m b^j{}_n \phi^h{}_m(x) \phi^h{}_n(y) \\ &= \sum_{m,n} a^i{}_m b^j{}_n \Phi^h{}_{m,n} \ . \end{aligned} \tag{4.2}$$

Since $v \in \mathbf{X}^{2h} \otimes \mathbf{Y}^{2h}$ implies $v \in \mathbf{X}^h \otimes \mathbf{Y}^h$, we must have :

$$\sum_{\alpha,\beta} q^{2h}{}_{\alpha,\beta} \Phi^{2h}{}_{\alpha,\beta} = \sum_{\mu,\nu} q^h{}_{\mu,\nu} \Phi^h{}_{\mu,\nu} \ (iff)$$

$$\sum_{\alpha,\beta} q^{2h}{}_{\alpha,\beta} \sum_{m,n} a^\alpha{}_m b^\beta{}_n \Phi^h{}_{m,n} = \sum_{\mu,\nu} q^h{}_{\mu,\nu} \Phi^h{}_{\mu,\nu} \ (iff)$$

$$\sum_{\alpha,\beta} q^{2h}{}_{\alpha,\beta} \sum_{\mu,\nu} a^\alpha{}_\mu b^\beta{}_\nu \Phi^h{}_{\mu,\nu} = \sum_{\mu,\nu} q^h{}_{\mu,\nu} \Phi^h{}_{\mu,\nu} \ ,$$

where the last step uses the fact that m,n are dummy indices. Therefore, we finally obtain :

$$\sum_{\mu,\nu} \left(\sum_{\alpha,\beta} a^\alpha{}_\mu b^\beta{}_\nu q^{2h}{}_{\alpha,\beta} - q^h{}_{\mu,\nu} \right) \Phi^h{}_{\mu,\nu} = 0 \ ,$$

which, by the linear independence of the basis elements Φ^h, implies :

$$q^h{}_{\mu,\nu} = \sum_{\alpha,\beta} a^\alpha{}_\mu b^\beta{}_\nu q^{2h}{}_{\alpha,\beta} \tag{4.3}$$

Equation (4.3) defines the interpolation operator from $\mathbf{X}^{2h} \otimes \mathbf{Y}^{2h}$ to $\mathbf{X}^h \otimes \mathbf{Y}^h$, expressed in the bases defined in (4.1). If we assume a lexicographical ordering of the bases, (4.3) can be written as :

$$\begin{cases} q^h{}_i = \sum\limits_j (I^h{}_{2h})_{i,j} q^{2h}{}_j \\ (I^h{}_{2h})_{i,j} \equiv a^\alpha{}_\mu b^\beta{}_\nu \end{cases} \tag{4.4}$$

Defining matrices $A = [a^\alpha{}_\mu]$ and $B = [b^\beta{}_\nu]$ we see that, with this ordering, $(I^h{}_{2h})_{i,j}$ is nothing but the i,j-th entry in the matrix $A \otimes B$, which is, by definition (see (4.1) and

Ch.I.D2.3) the Kronecker product of the interpolation operators between $\mathbf{X}^{2h}$ and $\mathbf{X}^{h}$, and between $\mathbf{Y}^{2h}$ and $\mathbf{Y}^{h}$, respectively, expressed in the same bases.

This proves our claim that the Kronecker product structure of the spaces is inherited by the interpolation operators. We made use of this fact when setting up the algorithm FAPIN to solve our two-dimensional model problems. We also note that the proof does not make explicit use of the fact that the spaces $\mathbf{X}$, $\mathbf{Y}$ are function spaces. Thus the result holds for general finite dimensional linear vector spaces.

APPENDIX IV

The High Level Environment HL

In this appendix we describe the main characteristics of the High Level Environment HL. We begin by briefly stating its main features. In Section 2 we introduce some of the sparse operators contained in HL and describe the form of sparse storage used. Section 3 presents the C-source code for one of the simplest HL sparse operators. Finally, section Four presents the HL code for the algorithm FAPIN used to implement the nonsingular experiments of this thesis. We assume some familiarity of the reader with the C programming language (see, for example, Kernighan & Ritchie [1978]).

1. Description

HL was developed by Benson [1987] and is, in essence, a sophisticated calculator based on an extension of the 'High Order Calculator' ('hoc') developed by Kernighan & Pike [1984, pp. 233-287]. It is implemented in the C programming language and it provides us with control of flow statements, the usual arithmetic and logical operators (extended to handle arrays), matrix operations and eigenanalysis, a sparse matrix set of operators and the ability to define HL functions and procedures. HL also supports recursion. Storage management is dynamic and its is handled through the UNIX environment (see Kernighan & Pike[1984]).

Its basic data structure is a record (called a 'structure' in C). This structure is defined in an 'include file' and it is shown in Figure 1.1 . Its fields include pointers to data (*index, *data), description strings (*name, *varname, *mark), object size information (row, col) and pointers for internal use by HL.

```
typedef struct Symbol { /* symbol table entry */
        char    *name;
        int     type;
        int     scratch;
        int     row;
        int     col;
        int     *index;
        double      *data;
        union {
              double  (*ptr)();     /* BLTIN */
              int     (*defp)();    /* FUNCTION, PROCEDURE */
              char    *str;         /* STRING */
        } u;
        struct Symbol *varlist;
        struct Symbol  *next;  /* to link to another */
        char *varname;
        char *mark;
} Symbol;
```

Fig. 1.1

Execution is controlled by a parser which generates a vector of instructions (i.e. pointers to variables and pointers to the C-functions that implement the corresponding operators), while evaluation of expressions is accomplished through a stack of pointers to the structures describing the objects (i.e. data). The parser is generated using the parser-generator 'yacc' (see Kernighan & Pike[1984, pp.235-265]) available under UNIX. This approach allows the user to easily add new operators which are either linked directly to the main body of HL, or are separate executable modules, which interface with HL through temporary files.

2. Some Sparse Operators

HL provides us with a form of sparse storage for matrices and a set of operators to handle objects stored in this form. In this section we describe the implementation of this kind of storage and comment briefly on some of the associated operators.

Given a matrix $A \in \mathbf{M}_{m\times n}(\mathbf{R})$, its sparse representation consists of two arrays of $m\times\mu$ elements, where μ is the maximum number of non-zero entries in the rows of A. A row of the first array, which we call 'index', contains the column indices of the non-zero entries in the corresponding row of A. The same row of the second array, which we call 'data', contains the actual values of the non-zero entries in that row of A. These arrays are pointed to by the fields '*index' and '*data' in the C structure representing the sparse version of A (see Fig 1.1). A null index pointer indicates a non-sparse object. As an example, consider the following matrix :

$$A = \begin{pmatrix} 7 & -5 & & & -2 \\ -2 & 7 & -5 & & \\ & -2 & 7 & -5 & \\ & & -2 & -7 & -5 \\ -5 & & & -2 & -7 \end{pmatrix}$$

Then, its HL sparse representation is :

```
INDEX ARRAY

[ 1]      [ 2]      [ 3]
         1         2         5
         1         2         3
         2         3         4
         3         4         5
         1         4         5
DATA ARRAY

[ 1]      [ 2]      [ 3]
         7        -5        -2
        -2         7        -5
        -2         7        -5
        -2         7        -5
        -5        -2         7
```

where the number in square brackets indicates the column in the corresponding array. Of course, HL provides us with an operator 'sparse' to transform a matrix from its full representation to the sparse one and an 'unsparse' operator that performs the reverse operation.

Of special interest to our experiments were the HL family of 'stretch' operators. These operators make use of the mark field in a given symbol (see Fig. 1.1) to store a string that describes the structure of matrices that have repeated blocks. This string can be used to considerably reduce the amount of storage required to represent certain kinds of matrices . The simplest of these operators are 'stretch' and 'stretchop'. Given a matrix in sparse representation, 'stretch' returns the sparse representation of the matrix resulting from repeating a certain specified block of rows of the original matrix, a specified number of times and shifted to the right in each repetition by a specified number of columns to the right. The necessary specifications are contained in the mark field of the original object for the matrix. 'Stretchop' applies the matrix resulting from 'stretch' to a given vector, without actually performing the stretch. This last operator is useful when implementing iterative algorithms that contain applications , to vectors, of matrices of the type that result from the discretization of two-point constant coefficient boundary value problems.

There are also two-dimensional counterparts of ‘stretchop’. First HL provides us with ‘stretchop2d’, which is a nested version of ‘stretchop’, that works with matrices of the type that result from the discretization of constant coefficient boundary value problems in two dimensions (i.e block-striped matrices). Stretching is done for each block in a fashion similar to ‘stretchop’. Stretching at the block level is performed using the same pattern as for the individual block stretching. The necessary information is again contained in the mark field of the sparse object.

A periodic version of ‘stretchop2d’ also exists in HL. This is the operator ‘pstretchop2d’, which works with matrices of the type that appear in the discretization of constant coefficient boundary value problems with periodic boundary conditions. (e.g. problems on a two-torus). Stretching is done as in ‘stretchop2d’ except that periodic ‘wrap-around’ is added both at the block level and internal to the blocks.

HL also provides us with some other sparse operators : ‘spsum’ adds matrices in sparse format; the operator ‘splsq’ calculates an LSQ-approximate inverse to a given matrix (see Ch.I.§3). The sparsity pattern to be used is specified by the user.

For a more extensive description of these operators, as well as several examples of their use, we refer the reader to Benson [1987].

3. Sample C-Source Code for an HL Operator

In this section we present an example of source code for the operator ‘stretchop’ described in the previous section.

Observe that the actual code for the numerical operator is surrounded by the code responsible for interfacing with the internal HL structure (that is, managing the stack, symbol fields, etc.). This interface is quite uniform, thus making it easy for the user to add new operators to HL.

```
stretchop()              /* returns stretch operator times column vector */
{
        Datum          d1, d2;
        int            end1, end2, end3, irow, blocks, nrep, shift, bstart, bend, block;
        int            bshift;
        char           *emalloc();
        int            *datind;
        double         *datbuf;
        double         sum;
        char           *key = "        ";
        int            i, j, k, l;
        int            ind;

        /*
         * HL Interface :
         * Pop stack,
         * Extract fields from symbol,
         * Check validity of input.
         */

        d1 = pop("stretchop");
        d2 = pop("stretchop");
        extracb(d1);
        extraca(d2);
        /*
         * read mark field
         */
        sscanf(amark, "%s%d%d%d%d", key, &bstart, &bend, &nrep, &shift);
        if (strcmp(key, "stretch") != 0)
                execerror("stretch given bad mark field", 0);
        block = bend - bstart + 1;
        /*
         * allocate working space
         */
        datbuf = (double *) emalloc("stretchop", sizeof(double) * (arow + (nrep - 1) * block));

        /*
         * Code for actual operator begins
         */

        for (i = 0; i < bend; i++) {
                sum = 0;
                for (k = 0; k < acol; k++) {
                        ind = (*(aindex + i * acol + k));
                        if (ind != 0)
                                sum += (*(adata + i * acol + k)) *
                                        *(bdata + ind - 1);
                }
                *(datbuf + i) = sum;
        }
        if (!((bend == arow) && (shift == 0))) {
                /* stretching required */
                blocks = (bstart - 1) * acol;
                for (l = 0; l < (nrep - 1); l++) {
```

```
            bshift = shift * (l + 1);
            for (i = 0; i < block; i++) {
                sum = 0;
                irow = bend + l * block + i;
                for (j = 0; j < acol; j++) {
                    ind = (*(aindex + blocks + i * acol + j));
                    if (ind != 0)
                        sum += (*(adata + blocks + i * acol + j)) *
                            *(bdata + ind - 1 + bshift);
                }
                *(datbuf + irow) = sum;
            }
        }
        end1 = acol * (nrep * block + bstart - 1);
        end2 = (nrep - 1) * shift;
        end3 = nrep * block + bstart - 1;
        for (i = bend; i < arow; i++) {
            sum = 0;
            for (k = 0; k < acol; k++) {
                ind = (*(aindex + i * acol + k));
                if (ind != 0)
                    sum += (*(adata + i * acol + k)) *
                        *(bdata + ind - 1 + end2);
            }
            *(datbuf + end3 + i - bend) = sum;
        }
    }

    /*
     * Code for operator ends
     */

    /*
     * HL interface :
     * Free space not needed,
     * Define fields in new symbol representing the result,
     * Create symbol,
     * Push symbol onto stack.
     */

    freesymb("stretchop", d2);
    d1.sym->row = arow + (nrep - 1) * block;
    d1.sym->col = 1;

    d1.sym->data = datbuf;
    d1.sym->scratch = 1;
    symdef(d1.sym);
    d1.sym->type = VEC;
    push(d1);
}
```

4. FAPIN code in HL

This section presents the HL implementation of the algorithm FAPIN defined in §4 of Chapter One. We give the direct implementation of the pseudo-code described in D4.2 of Chapter One. This is :

```
func fapin() {

        /*
         * arguments to function :
         * $1 -> grid index
         * $2 -> residual vector
         */

if ( $1 == 0) {

        zk = applylsq($1,$2)
        return zk

} else {

        rkm = collect($1,$2)
        zkm = fapin(($1-1),1*rkm)
        zk = interpolate(($1-1),zkm)
                iter = 1
                while(iter <= relax){

                        s = $2 - (applyop($1,zk))
                        zk = zk + (applylsq($1,s))
                        iter = iter + 1

                }

        return zk

        }

}
```

The above implementation uses the functions 'collect' and 'interpolate' to accomplish the inter-grid transfers. The functions 'applyop' and 'applylsq' return the result of applying the discrete differential operator and its LSQ-approximate inverse (in the corresponding grid) to a given vector, respectively. The form of these functions is illustrated by 'applyop' (for the one dimensional non-periodic case) :

```
func applyop() {

        /*
         * arguments to function :
         * $1 -> Grid index
         * $2 -> The vector to be acted upon
         */

        if($1 == 10) return (A10 stretchop $2)
        if($1 == 9) return (A9 stretchop $2)
        if($1 == 8) return (A8 stretchop $2)
        if($1 == 7) return (A7 stretchop $2)
        if($1 == 6) return (A6 stretchop $2)
        if($1 == 5) return (A5 stretchop $2)
        if($1 == 4) return (A4 stretchop $2)
        if($1 == 3) return (A3 stretchop $2)
        if($1 == 2) return (A2 stretchop $2)
        if($1 == 1) return (A1 stretchop $2)
        if($1 == 0) return (A0 stretchop $2)

}
```

We note that the ‘while’ loop in FAPIN effectively amounts to the use of a different approximate inverse (see Ch.I.§2). The number of relaxations is controlled by the global variable ‘relax’ which is initialized in the main program.

APPENDIX V

Proof of Theorem 3.1 of Chapter One

In this appendix we give the proof of Ch.I.T3.1. We recall it here, for convenience and refer the reader to Ch.I.§3 for further details :

Theorem 3.1 : *Let* $A = A' \otimes A''$, $A' \in \mathbf{M}_m(\mathbf{R})$, $A'' \in \mathbf{M}_n(\mathbf{R})$. Let $B\,' \in [R]$, $B\,'' \in [S]$, be LSQ-approximate inverses of A', A'', respectively. Then, the LSQ-approximate inverse $B \in [R \otimes S]$ of A satisfies :

$$B = B\,' \otimes B\,'' \quad . \tag{1}$$

Proof :

In this proof all vectors considered are row vectors. We remark that the notation used is the following : Capital letters denote matrices; capital letters subscripted by a lower case letter denote the row vector formed by the indicated row of the corresponding matrix; subscripted lower case Greek letters denote the row vector formed by only the non-zero entries in the indicated row of the matrix represented by the Latin counterpart of the greek symbol; finally, doubly subscripted lower case letters, denote entries in the corresponding matrix.

Let $\beta_k C_{A' \otimes A''} = e_k$, $\beta'_i C_{A'} = e'_i$, and $\beta''_j C_{A''} = e''_j$, be the systems to be satisfied (in the least squares sense) by the k-th, i-th, and j-th row of B, $B\,'$, and $B\,''$, respectively (see equations Ch.I.(3.3), Ch.I.(3.4)). These are equivalent to the normal equations :

$$\beta_k C_{A' \otimes A''} C_{A' \otimes A''}{}^t = e_k C_{A' \otimes A''}{}^t \tag{2}$$
$$\beta'_i C_{A'} C_{A'}{}^t = e'_i C_{A'}{}^t \tag{3}$$
$$\beta''_i C_{A''} C_{A''}{}^t = e''_i C_{A''}{}^t , \tag{4}$$

respectively.

We first prove that, under the previous assumptions

$$C_{A'\otimes A''} = C_{A'}\otimes C_{A''} \tag{5}$$

holds. Since by hypothesis $B \in [R\otimes S]$, the sparsity pattern of B is that of $W \equiv R\otimes S$. Let

$$\rho_i = (r_{ii_1}, r_{ii_2}, \dots, r_{ii_r})$$
$$\sigma_j = (s_{jj_1}, s_{jj_2}, \dots, s_{jj_s})$$

be the row-vectors formed by the nonzero entries in the i-th, j-th row of R, S, respectively. Then, since $W_k = R_i\otimes S_j$, $k = (i-1)n + j$, the nonzero entries of the k-th row of W are :

$$\begin{aligned}\omega_k = \rho_i\otimes\sigma_j = (&r_{ii_1}s_{jj_1}, r_{ii_1}s_{jj_2}, \dots, r_{ii_1}s_{jj_s},\\ &r_{ii_2}s_{jj_1}, r_{ii_2}s_{jj_2}, \dots, r_{ii_2}s_{jj_s},\\ &, \dots ,\\ &r_{ii_r}s_{jj_1}, r_{ii_r}s_{jj_2}, \dots, r_{ii_r}s_{jj_s}),\end{aligned}$$

which are located in columns

$$\begin{aligned}&(i_1-1)n + j_1, (i_1-1)n + j_2, \dots, (i_1-1)n + j_s,\\ &(i_2-1)n + j_1, (i_2-1)n + j_2, \dots, (i_2-1)n + j_s,\\ &, \dots ,\\ &(i_r-1)n + j_1, (i_r-1)n + j_2, \dots, (i_r-1)n + j_s,\end{aligned}$$

of W_k. Thus, the block $C_{A'\otimes A''}$ consists of the rows (see Ch.I.(3.5)) $(i_1-1)n + j_1, \dots, (i_r-1)n + j_s$ of $A'\otimes A''$. But since

$$C_{A'}\otimes C_{A''} = \begin{pmatrix} A'_{i_1}\otimes A''_{j_1} \\ A'_{i_1}\otimes A''_{j_2} \\ \cdots \\ A'_{i_1}\otimes A''_{j_s} \\ A'_{i_2}\otimes A''_{j_1} \\ A'_{i_2}\otimes A''_{j_2} \\ \cdots \\ A'_{i_2}\otimes A''_{j_s} \\ A'_{i_r}\otimes A''_{j_1} \\ A'_{i_r}\otimes A''_{j_2} \\ \cdots \\ A'_{i_r}\otimes A''_{j_s} \end{pmatrix}$$

looking at the structure of $A'\otimes A''$, written as Kronecker products of rows of A' and A'', we see that :

$A'_{i_1}\otimes A''_{j_1}$ *is the* $(i_1-1)n + j_1$*-th row of* $A'\otimes A''$,

, . . . ,

$A'_{i_r}\otimes A''_{j_s}$ *is the* $(i_r-1)n + j_s$*-th row of* $A'\otimes A''$,

thus proving (5).

Now, using (5), (2) can be rewritten :

$$\begin{aligned} \beta_k\, C_{A'\otimes A''}\, C_{A'\otimes A''}{}^t &= e_k\, C_{A'\otimes A''}{}^t \quad \textit{iff} \quad (\textit{by } (5)) \\ \beta_k(C_{A'}\otimes C_{A''})\,(C_{A'}\otimes C_{A''})^t &= e_k(C_{A'}\otimes C_{A''})^t \quad \textit{iff} \quad (\textit{by Ch.I.}(3.6) \text{ and } \textit{Ch.I.}(3.7)) \\ \beta_k(C_{A'}\, C_{A'}{}^t\otimes C_{A''}\, C_{A''}{}^t) &= e_k(C_{A'}{}^t\otimes C_{A''}{}^t) \quad \textit{iff} \quad (\textit{since } e_k = e'_i\otimes e''_j) \\ \beta_k(C_{A'}\, C_{A'}{}^t\otimes C_{A''}\, C_{A''}{}^t) &= (e'_i\otimes e''_j)(C_{A'}{}^t\otimes C_{A''}{}^t) \quad \textit{iff} \quad (\textit{by Ch.I.}(3.6)) \\ \beta_k(C_{A'}\, C_{A'}{}^t\otimes C_{A''}\, C_{A''}{}^t) &= (e'_i\, C_{A'}{}^t\otimes e''_j\, C_{A''}{}^t) \ . \end{aligned}$$

Letting $\beta_k = \beta'_i\otimes\beta''_j$ we get :

$$\begin{aligned} (\beta'_i\otimes\beta''_j)\,(C_{A'}\, C_{A'}{}^t\otimes C_{A''}\, C_{A''}{}^t) &= (e'_i\, C_{A'}{}^t\otimes e''_j\, C_{A''}{}^t) \quad \textit{iff} \quad (\textit{by Ch.I.}(3.6)) \\ (\beta'_i\, C_{A'}\, C_{A'}{}^t\otimes\beta''_j\, C_{A''}\, C_{A''}{}^t) &= (e'_i\, C_{A'}{}^t\otimes e''_j\, C_{A''}{}^t) \quad , \end{aligned}$$

which is an identity (see (3) and (4)), thus proving the theorem □.

We note that an analogous result, for a different type of approximate inverse, is stated without proof in Benson & Frederickson [1982, p.132].