Fast Fourier Transform Solvers for Quadratic Spline Collocation

Athanasia Constas
Master of Science
Department of Computer Science
University of Toronto

Abstract

The solution of linear systems arising from the discretisation of second order elliptic Boundary Value Problems (BVPs) defined in rectangular domains by solvers employing Fast Fourier Transforms (FFTs) is studied. The types of boundary conditions considered are homogeneous Dirichlet, periodic, homogeneous Neumann or combinations of these. Two-dimensional and three-dimensional problems are considered. Quadratic spline collocation methods are used for the discretisation of the BVPs. The FFT solvers are first developed for differential operators with constant coefficients and even order derivatives. Diagonal scalings of these solvers are then used as preconditioners for the solution of linear systems arising from the discretisation of general second order linear elliptic BVPs. Several acceleration methods are tested. Results from numerical experiments that demonstrate the asymptotically optimal convergence and computational performance of the FFT solvers are presented.

Acknowledgments

Primarily, I would like to express my deep appreciation to my supervisor, Professor Christina C. Christara for her thorough guidance of the project, interest and assistance.

Also, I would like to thank Professor K. R. Jackson for his very useful comments on the thesis.

Furthermore, I am grateful to my parents for their love, patience and support through my graduate studies. I thank my sister Stella for her encouragement and help.

Last but not least, I would like to acknowledge the School of Graduate Studies, the Department of Computer Science of the University of Toronto and Professor Christina C. Christara for the financial support.

Contents

Abstract ii

Acknowledgments iii

Table of Contents iv

List of Tables v

List of Figures vi

1 Introduction 1

1.1 The optimal quadratic spline collocation method 4

1.2 Spectral properties of the one-dimensional QSC matrix 6

1.2.1 *The case of the homogeneous Dirichlet boundary conditions* . . 6

1.2.2 *The case of periodic boundary conditions* 8

1.2.3 *The case of homogeneous Neumann boundary conditions* . . . 9

1.3 Spectral properties of the two-dimensional QSC matrix 11

1.4 Fast Fourier and trigonometric transforms 13

1.5 Properties of tensor products of matrices 15

2 Fast Fourier transform solution of QSC equations **17**

2.1 FFT solution of one-dimensional QSC equations 17

2.1.1 *The case of homogeneous Dirichlet boundary conditions* 17

2.1.2 *The case of periodic boundary conditions* 19

2.1.3 *The case of homogeneous Neumann boundary conditions* . . . 20

2.2 FFT solution of two-dimensional QSC equations 21
2.3 Performance results 22

3 The variable coefficient case **26**
3.1 Preconditioners for QSC equations 28
3.2 Iterative methods for QSC equations 29
3.3 Numerical results. 30

4 Three-dimensional problems **42**
4.1 Spectral properties of the three-dimensional QSC matrix 42
4.2 FFT solution of the three-dimensional QSC equations 45
4.3 The constant coefficient case 46
4.4 The variable coefficient case 49

5 Conclusions **55**

A Problems **57**

Bibliography **62**

List of Tables

2.1 Errors and respective orders of convergence corresponding to Problem 1 discretised by the QSC method, for several grid sizes n and for both steps of the QSC method. Time in seconds for the FFT solution of the QSC equations for several grid sizes $n \times n$ and for one step of the QSC method. 24

3.1 Number of iterations and time in seconds required for the convergence of various preconditioned iterative techniques for the solution of the QSC equations arising from the discretisation of Problem 2 for several grid sizes n, and for both steps of the QSC method. 32

3.1 Continued. 33

3.1 Continued. 34

3.2 Number of iterations required for the convergence of various preconditioned iterative techniques for the solution of the QSC equations arising from the discretisation of Problem 3 for several grid sizes n, for three values of the parameter δ, and for both steps of the QSC method. A '-' denotes that the method did not converge after 200 iterations. . . 36

3.3 Number of iterations required for the convergence of various preconditioned iterative techniques for the solution of the QSC equations arising from the discretisation of Problem 4 for several grid sizes n, and for both steps of the QSC method. 37

3.4 Number of iterations and time in seconds required for the convergence of GMRES for the solution of the QSC equations arising from the discretisation of Problem 5 for several grid sizes n, and for both steps of the QSC method. 41

4.1 Errors and respective orders of convergence corresponding to Problem 6 discretised by the QSC method, for several grid sizes n and for both steps of the QSC method. Time in seconds for the FFT solution of the QSC equations corresponding to Problem 6 for several grid sizes n and for one step of the QSC method. 48

4.2 Number of iterations and time in seconds required for the convergence of various preconditioned iterative techniques for the solution of the QSC equations arising from the discretisation of Problem 7 for several grid sizes n, and for both steps of the QSC method. 52

4.2 Continued. 53

4.2 Continued. 54

List of Figures

2.1 Plot of time in seconds for the FFT solution of the QSC equations corresponding to Problem 1 versus the grid size n in one-dimension. The slope of the line in a log-log plot, computed by linear least squares fit to the data, is 2.1. 25

3.1 Plot of time in seconds for both steps of the QSC method versus the grid size n in one dimension for three solvers applied to the QSC equations corresponding to Problem 4. The solid line corresponds to GMRES-DH-FFT, the heavy dotted line to MRES-DH-FFT and the dotted line to GMRES-MGJ. The slopes of the three lines in a log-log plot, computed by a linear least squares fit to the data, are: GMRES-DH-FFT: 2.05, MRES-DH-FFT: 2.08 and GMRES-MGJ: 1.9. 39

Chapter 1

Introduction

Partial Differential Equations (PDEs) are commonly used as mathematical models for many physical phenomena. Therefore, the accurate and efficient solution of PDEs is of primary concern to scientists and engineers. The numerical solution of PDEs involves two main steps: the discretisation of the continuous problem and the solution of the resulting system of discrete algebraic equations. The accuracy of the computed solution to the PDE problem depends heavily on the discretisation method used in the first step. The efficiency of the computation depends mainly on the solver used in the second step. In this thesis, a class of fast solvers for the set of linear equations arising from the discretisation of linear elliptic PDEs by quadratic spline collocation is studied.

Spline collocation discretisation methods belong to the class of Finite Element Methods (FEMs). They are simple to implement and do not require the computation of integrals. Their standard formulation leads to sub-optimal methods, in the sense that the order of convergence of the spline collocation approximation is lower than that of the interpolant in the same approximation space. Optimal collocation methods based on smooth splines have been developed and analysed by [Arc73], [DaSw75], [Fyf68], [HCR88], [Iro87], and [PaSo87] for one-dimensional Boundary Value Problems (BVPs) and uniform discretisations. For two-dimensional BVPs defined in rectangular domains, and for uniform discretisations, [Chr88], [Chr94], [HVR88], and [Iro87] derived optimal collocation methods based on tensor products of smooth splines.

The development and analysis of efficient solvers for the spline collocation equations are at an early stage. Some solvers for the quadratic spline collocation equations are considered in [Chr96] and their parallel performance tested. The present thesis contributes to this development, in the sense that it explores fast direct solvers for the spline collocation equations arising from the discretisation of a subclass of PDE problems and uses them as preconditioners for the spline collocation equations arising from the discretisation of general linear elliptic PDE problems in rectangular domains. The solvers considered are based on Fast Fourier Transforms (FFTs) and are suitable for parallel computation.

FFTs are extensively used for the solution of linear systems arising from the discretisation of Helmholtz problems on rectangular domains; see [Loa92] and references therein. The discretisation method considered in many related publications is centered differences, that is, the well-known five-point-star for two-dimensional domains. For general PDE operators the FFT solvers can be used as preconditioners.

Concus and Golub [CoGo73] study one-step preconditioned iterative methods for the solution of the linear system of equations arising from the discretisation of self-adjoint non-separable linear elliptic PDEs by standard centered differences. The preconditioner considered is the matrix arising from the discretisation of the Laplacian. Techniques for improving the convergence of the iterative procedure include scaling of the original problem, diagonally shifting the preconditioner, and Chebyshev iteration. The rate of convergence is shown to be independent of mesh size for smooth coefficients. The analysis is carried out for Dirichlet boundary conditions. It is claimed though that, with some modifications, it can be extended to other types of boundary conditions.

Bank [Ban77] follows a similar approach, by choosing the preconditioner to be the matrix arising from the discretisation of a separable problem, for which the eigenvalues possess certain properties. The separable problem is solved by an alternating direction iterative procedure. Both the development and the analysis of the proposed method depend on the discretisation method used (i.e., standard centered differences).

Elman and Schultz [ElSc86] use acceleration methods for non-symmetric systems (e.g., Conjugate Gradient on the normal equations and Orthomin(k)) for the solution of non-self-adjoint non-separable problems with Dirichlet boundary conditions. The discretisation method considered is standard centered differences. It is shown that, if the preconditioner is the symmetric part of the matrix to be solved, the convergence rate is independent of the mesh size. Numerical experiments indicate that non-symmetric preconditioners behave similarly.

Bialecki et al. [BFB92] developed FFT based solvers for the Hermite bicubic spline collocation equations arising from Poisson's equation. Applications of these solvers to one-step preconditioned iterative methods for the solution of more general elliptic PDEs are discussed in [Bia94]. It is shown that the orthogonal spline collocation operator corresponding to a self-adjoint elliptic PDE with homogeneous Dirichlet boundary conditions is spectrally equivalent to the orthogonal spline collocation Laplace operator.

Greenbaum [Gre92] studies the condition number of preconditioned matrices arising from the discretisation of self-adjoint PDE operators with homogeneous Dirichlet boundary conditions by finite element or finite difference methods. The matrices arising from the discretisation are assumed to be symmetric positive definite. The preconditioners are symmetric diagonal scalings of the Laplacian. It is shown that, when the PDE operator coefficients are discontinuous but piecewise constant, the optimal scaling is the identity. Moreover, numerical evidence shows that, when the PDE operator coefficients are smooth, the optimal scaling is the square root of the matrix diagonal.

The discretisation of Helmholtz PDE problems with constant coefficients by quadratic spline collocation leads to linear systems which have eigenvalues and eigenvectors given by sine and cosine formulae [Chr94]. Therefore, FFT solvers for the above linear systems can be developed. In this thesis, we develop and implement FFT solvers for the above linear systems, and test their computational performance.

For the solution of the quadratic spline collocation equations arising from general elliptic PDEs with homogeneous Dirichlet, periodic, homogeneous Neumann or mixed boundary conditions we study diagonal scalings of the Laplacian as preconditioners. A variety of iterative methods and acceleration techniques is considered. Experimental evidence indicates that for most problems the rate of convergence of the resulting preconditioned iterative methods is independent of the discretisation step-size. Thus, the FFT solvers are shown to be effective methods for the solution of the quadratic spline collocation equations.

In the rest of this chapter, we briefly review the properties of the quadratic spline collocation linear system. FFT techniques used for the solution of linear systems with certain eigenproperties are also reviewed. In Chapter 2, we describe the FFT solvers for quadratic spline collocation equations arising from Helmholtz PDE operators with Dirichlet, Neumann, periodic or mixed boundary conditions, and test their computational performance. In Chapter 3, iterative methods with FFT preconditioners are tested on quadratic spline collocation equations arising from general elliptic operators. In Chapter 4, results from numerical experiments on three-dimensional problems are presented. Our concluding remarks are presented in Chapter 5.

1.1 The optimal quadratic spline collocation method

We consider a BVP described by the operator equation in a domain Ω (one-dimensional or rectangular multi-dimensional),

$$\mathbf{L}u(x) = g(x) \text{ in } \Omega, \tag{1.1}$$

and some boundary conditions defined on the boundary, $\partial\Omega$ of Ω,

$$\mathbf{B}u(x) = \gamma(x) \text{ on } \partial\Omega. \tag{1.2}$$

Here $\mathbf{L}$ is a linear elliptic differential operator, $\mathbf{B}$ is a linear boundary differential operator, g and γ are given functions of x (one-dimensional or multi-dimensional), and u is the unknown function of x.

Given a node partition Δ of Ω, we choose a set of basis functions for the quadratic spline space, that is, the space of quadratic piecewise polynomials of continuity $\mathbf{C}^1$ on the nodes of the partition. We also define the set of *data* points or *collocation* points in Ω and on $\partial\Omega$. If Ω is one-dimensional, the collocation points are the midpoints of the subintervals of the partition and the boundary nodes. If Ω is multi-dimensional, the collocation points are tuples of collocation points in each dimension. The collocation method determines the approximation u_Δ to u by requiring that the residuals $\mathbf{L}u_\Delta - g$ and $\mathbf{B}u_\Delta - \gamma$ of the differential operator $\mathbf{L}$ and boundary operator $\mathbf{B}$, respectively, are zero on the data points. If the approximate space is a space of smooth splines, this formulation leads to non-optimal solution approximations, in the sense that the convergence order of the spline collocation approximation is lower than the order of the spline interpolant in the same approximation space. The formulation of the optimal spline collocation methods is based on the construction of appropriate perturbations $\mathbf{P}_{\mathbb{L}}$ and $\mathbf{P}_{\mathbb{B}}$ of the operators $\mathbf{L}$ and $\mathbf{B}$ respectively. Two formulations were derived: the *one-step* or *extrapolated* methods, in which u_Δ is determined by requiring that the residuals $(\mathbf{L} + \mathbf{P}_{\mathbb{L}})u_\Delta - g$ and $(\mathbf{B} + \mathbf{P}_{\mathbb{B}})u_\Delta - \gamma$ of the perturbed operators $\mathbf{L} + \mathbf{P}_{\mathbb{L}}$ and $\mathbf{B} + \mathbf{P}_{\mathbb{B}}$, respectively, are zero on the data points, and the *multiple-step* or *deferred-correction* methods, in which a low (second) order approximation is generated first, and in subsequent steps, higher order approximations are generated, by moving the perturbation operators to the right side of the PDE problem, and applying them to the lower order approximations of the previous step(s). The number of such steps required depends on the order of the BVP and the degree of splines used. For example, for second-order problems and quadratic or cubic splines, two steps suffice.

Both formulations are equivalent with respect to convergence properties, but the deferred-correction methods are more efficient with respect to time and memory requirements [HRCV88], [Chr88], because they give rise to a linear system having a smaller bandwidth, with fewer nonzero entries per row than the extrapolated methods. In this thesis we consider FFT-based methods for the solution of the linear system arising from the two-step QSC equations.

1.2 Spectral properties of the one-dimensional QSC matrix

1.2.1 *The case of the homogeneous Dirichlet boundary conditions*

Consider the differential equation

$$\mathbf{L}u(x) \equiv au'' + fu = g(x) \text{ in } \Omega \equiv (0,1) \tag{1.3}$$

subject to *homogeneous Dirichlet* boundary conditions

$$\mathbf{B}u(x) \equiv u = 0 \text{ on } x = 0,\ x = 1 \tag{1.4}$$

where a and f are constants.

A set of basis functions for the quadratic spline space constructed on a partition of Ω with uniform step-size $h = \frac{1}{n}$ is the set of quadratic B-splines $\{\phi_i(x)\}_{i=0}^{n+1}$ with $\phi_i(x) = \frac{1}{2}\xi(\frac{x}{h} - i + 2)$ where the quadratic spline function ξ is defined by

$$\xi(x) = \begin{cases} x^2 & \text{if } 0 \le x \le 1, \\ -3 + 6x - 2x^2 & \text{if } 1 \le x \le 2, \\ 9 - 6x + x^2 & \text{if } 2 \le x \le 3, \\ 0 & \text{elsewhere.} \end{cases} \tag{1.5}$$

Using the functions $\phi_i(x)$ we can construct a set of quadratic spline basis functions, $\{\psi_i(x)\}_{i=1}^{n}$ that satisfy the homogeneous boundary conditions (1.4). They are

$$\psi_1(x) = \phi_1(x) - \phi_0(x);\ \psi_i(x) = \phi_i(x),\ i = 2, \ldots, n-1;\ \psi_n(x) = \phi_n(x) - \phi_{n+1}(x).$$

The QSC method applied to the BVP (1.3)-(1.4) using the basis functions $\{\psi_i(x)\}_{i=1}^{n}$ results in a tridiagonal linear system of the form

$$A^{hd}\mathrm{x} \equiv (a\frac{1}{h^2}T_{-2}^{hd} + \frac{1}{8}fT_6^{hd})\mathrm{x} = \mathrm{g} \tag{1.6}$$

where

$$T_{-2}^{hd} \equiv \begin{bmatrix} -3 & 1 & & & & \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \cdot & \cdot & \cdot & \\ & & & 1 & -2 & 1 \\ & & & & 1 & -3 \end{bmatrix}, \; T_6^{hd} \equiv \begin{bmatrix} 5 & 1 & & & & \\ 1 & 6 & 1 & & & \\ & 1 & 6 & 1 & & \\ & & \cdot & \cdot & \cdot & \\ & & & 1 & 6 & 1 \\ & & & & 1 & 5 \end{bmatrix} \text{ and } \mathrm{g} \equiv \begin{bmatrix} g(h/2) \\ g(3h/2) \\ g(5h/2) \\ \vdots \\ g(\frac{(2n-3)h}{2}) \\ g(\frac{(2n-1)h}{2}) \end{bmatrix}. \tag{1.7}$$

As shown in [Chr94], the eigenvalues λ_l^{hd}, $l = 1, \ldots, n$, of the matrix T_{-2}^{hd} in (1.7) are given by

$$\lambda_l^{hd} = -4sin^2 \frac{l\pi}{2n}, \tag{1.8}$$

and its eigenvectors δ_l^{hd}, $l = 1, \ldots, n$, are

$$\delta_{lj}^{hd} = \kappa_l^{hd} \sin \frac{(2j-1)l\pi}{2n}, \; j = 1, \ldots, n, \tag{1.9}$$

where κ_l^{hd} are constants. Note that the eigenvectors δ_l^{hd} are orthogonal. To normalise the eigenvectors with respect to the Euclidean norm, set

$$\kappa_l^{hd} = \sqrt{\frac{2}{n}}, \; l = 1, \ldots, n-1, \text{ and } \kappa_n^{hd} = \sqrt{\frac{1}{n}}. \tag{1.10}$$

By observing that $T_6^{hd} = T_{-2}^{hd} + 8\mathbb{I}$, we see that the eigenvalues μ_l^{hd}, $l = 1, \ldots, n$, of the matrix T_6^{hd} in (1.7) are

$$\mu_l^{hd} = \lambda_l^{hd} + 8, \tag{1.11}$$

and its eigenvectors are the same as those of T_{-2}^{hd}.

Clearly, the eigenvalues σ_l^{hd}, $l = 1, \cdots, n$, of A^{hd} are

$$\sigma_l^{hd} = a\frac{1}{h^2}\lambda_l^{hd} + \frac{1}{8}f\mu_l^{hd}, \tag{1.12}$$

and its eigenvectors are the same as those of T_{-2}^{hd}.

1.2.2 *The case of periodic boundary conditions*

Consider now the differential equation (1.3) subject to *periodic* boundary conditions

$$u(0) = u(1) \text{ and } u'(0) = u'(1). \tag{1.13}$$

Due to the periodic conditions (1.13) the first and last coefficients (degrees of freedom) of the finite element representation of the QSC approximation to u are equal to the one before the last and the second, respectively. Therefore, a set of quadratic spline basis functions for the BVP (1.1)-(1.13) is $\{\phi_i(x)\}_{i=1}^n$.

The QSC method applied to the BVP (1.3)-(1.13) using the basis functions $\{\phi_i(x)\}_{i=1}^n$ results in an almost tridiagonal linear system of the form

$$A^p \mathrm{x} \equiv (a\frac{1}{h^2}T_{-2}^p + \frac{1}{8}fT_6^p)\mathrm{x} = \mathrm{g} \tag{1.14}$$

where

$$T_{-2}^p \equiv \begin{bmatrix} -2 & 1 & & & & 1 \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \cdot & \cdot & \cdot & \\ & & & 1 & -2 & 1 \\ 1 & & & & 1 & -2 \end{bmatrix}, \; T_6^p \equiv \begin{bmatrix} 6 & 1 & & & & 1 \\ 1 & 6 & 1 & & & \\ & 1 & 6 & 1 & & \\ & & \cdot & \cdot & \cdot & \\ & & & 1 & 6 & 1 \\ 1 & & & & 1 & 6 \end{bmatrix}, \; \mathrm{g} \equiv \begin{bmatrix} g(h/2) \\ g(3h/2) \\ g(5h/2) \\ \vdots \\ g(\frac{(2n-3)h}{2}) \\ g(\frac{(2n-1)h}{2}) \end{bmatrix}. \tag{1.15}$$

Note that the BVP (1.3)-(1.13) discretised by standard finite differences gives rise to the matrix

$$a\frac{1}{h^2}T_{-2}^p + f\mathbb{I}_n,$$

where $\mathbb{I}_n$ is the identity matrix of dimension n. That is, the u'' term of the differential operator of (1.3) discretised by either standard differences or by QSC gives rise to the same matrix.

It is known that the eigenvalues λ_l^p, $l = 1, \cdots, n$, of the matrix T_{-2}^p in (1.15) are given by

$$\lambda_l^p = -4 \; sin^2\frac{(l-1)\pi}{n}, \tag{1.16}$$

and its eigenvectors δ_l^p, $l = 1, \ldots, n$, by

$$\delta_{lj}^p = \kappa_l^p \left(\cos \frac{2(j-1)(l-1)\pi}{n} + i \sin \frac{2(j-1)(l-1)\pi}{n} \right), \; j = 1, \ldots, n, \quad (1.17)$$

where κ_l^p are constants and $i \equiv \sqrt{-1}$. Note that the eigenvectors δ_l^p are orthogonal. To normalise the eigenvectors with respect to the Euclidean norm, set

$$\kappa_l^p = \sqrt{\frac{1}{n}}, l = 1, \ldots, n. \quad (1.18)$$

Since $T_6^p = T_{-2}^p + 8\mathbb{I}_n$, the eigenvalues μ_l^p, $l = 1, \ldots, n$, of the matrix T_6^p in (1.7) are

$$\mu_l^p = \lambda_l^p + 8, \quad (1.19)$$

and its eigenvectors are the same as those of T_{-2}^p.

Clearly, the eigenvalues σ_l^p, $l = 1, \ldots, n$, of A^p are

$$\sigma_l^p = a\frac{1}{h^2}\lambda_l^p + \frac{1}{8} f \mu_l^p, \quad (1.20)$$

and its eigenvectors are the same as those of T_{-2}^p.

1.2.3 *The case of homogeneous Neumann boundary conditions*

Finally, consider the differential equation (1.3) subject to *homogeneous Neumann* boundary conditions

$$\mathbf{B}u(x) \equiv u_n = 0 \text{ on } x = 0, \; x = 1. \quad (1.21)$$

where u_n is the normal derivative of u. For the one-dimensional BVP (1.3)-(1.21) we have $u_n = u_x$.

Using the functions $\phi_i(x)$ we can construct a set of quadratic spline basis functions, $\{\theta_i(x)\}_{i=1}^n$ that satisfy the homogeneous boundary conditions (1.21). They are

$$\theta_1(x) = \phi_1(x) + \phi_0(x); \; \theta_i(x) = \phi_i(x), \; i = 2, \ldots, n-1; \; \theta_n(x) = \phi_n(x) + \phi_{n+1}(x).$$

The QSC method applied to the BVP (1.3)-(1.21) using the basis functions $\{\theta_i(x)\}_{i=1}^n$ results in a tridiagonal linear system of the form

$$A^{hn}\mathrm{x} \equiv (a\frac{1}{h^2}T_{-2}^{hn} + \frac{1}{8}fT_6^{hn})\mathrm{x} = \mathrm{g} \tag{1.22}$$

where

$$T_{-2}^{hn} \equiv \begin{bmatrix} -1 & 1 & & & & \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \cdot & \cdot & \cdot & \\ & & & 1 & -2 & 1 \\ & & & & 1 & -1 \end{bmatrix}, \; T_6^{hn} \equiv \begin{bmatrix} 7 & 1 & & & & \\ 1 & 6 & 1 & & & \\ & 1 & 6 & 1 & & \\ & & \cdot & \cdot & \cdot & \\ & & & 1 & 6 & 1 \\ & & & & 1 & 7 \end{bmatrix}, \; \mathrm{g} \equiv \begin{bmatrix} g(h/2) \\ g(3h/2) \\ g(5h/2) \\ \vdots \\ g(\frac{(2n-3)h}{2}) \\ g(\frac{(2n-1)h}{2}) \end{bmatrix}. \tag{1.23}$$

As shown in [Chr94], the eigenvalues λ_l^{hn}, $l = 1, \ldots, n$, of the matrix T_{-2}^{hn} in (1.23) are given by

$$\lambda_l^{hn} = -4 \ sin^2\frac{(l-1)\pi}{2n}, \tag{1.24}$$

and its eigenvectors δ_l^{hn}, $l = 1, \ldots, n$, are

$$\delta_{lj}^{hn} = \kappa_l^{hn} \cos\frac{(2j-1)(l-1)\pi}{2n}, \; j = 1, \ldots, n, \tag{1.25}$$

where κ_l^{hn} are constants. Note that the eigenvectors δ_l^{hn} are orthogonal. To normalise the eigenvectors with respect to the Euclidean norm, set

$$\kappa_1^{hn} = \sqrt{\frac{1}{n}} \text{ and } \kappa_l^{hn} = \sqrt{\frac{2}{n}}, \; l = 2, \ldots, n. \tag{1.26}$$

By observing that $T_6^{hn} = T_{-2}^{hn} + 8\mathbb{I}$, we see that the eigenvalues μ_l^{hn}, $l = 1, \ldots, n$, of the matrix T_6^{hn} in (1.23) are

$$\mu_l^{hn} = \lambda_l^{hn} + 8, \tag{1.27}$$

and its eigenvectors are the same as those of T_{-2}^{hn}.

Clearly, the eigenvalues σ_l^{hn}, $l = 1, \ldots, n$, of A^{hn} are

$$\sigma_l^{hn} = a\frac{1}{h^2}\lambda_l^{hn} + \frac{1}{8}f\mu_l^{hn}, \tag{1.28}$$

and its eigenvectors are the same as those of T_{-2}^{hn}.

1.3 Spectral properties of the two-dimensional QSC matrix

Consider the PDE

$$\mathbf{L}u(x,y) \equiv au_{xx} + cu_{yy} + fu = g(x,y) \text{ in } \Omega \equiv (0,1)\times(0,1) \tag{1.29}$$

subject to *periodic* boundary conditions in the x direction

$$\mathbf{B}_1 u(x,y) \equiv u(0,y) = u(1,y), \text{ for } 0 \le y \le 1 \tag{1.30}$$

$$\mathbf{B}_2 u(x,y) \equiv u_x(0,y) = u_x(1,y), \text{ for } 0 \le y \le 1 \tag{1.31}$$

and *homogeneous Dirichlet* boundary conditions in the y direction

$$\mathbf{B}_3 u(x,y) \equiv u = 0 \text{ on } y = 0,\ y = 1, \text{ for } 0 \le x \le 1 \tag{1.32}$$

where a, c and f are constants.

We consider a rectangular partition Δ of Ω with uniform step sizes $h_x = \dfrac{1}{n}$ and $h_y = \dfrac{1}{m}$ in the x and y directions respectively. A set of basis functions for the bi-quadratic spline space constructed on the partition Δ of Ω is the tensor product of quadratic B-splines $\{\phi_i^x(x)\phi_j^y(y)\}_{i=0,j=0}^{m+1,n+1}$ with $\phi_i^x(x) = \dfrac{1}{2}\xi(\dfrac{x}{h_x} - i + 2)$ and $\phi_j^y(y) = \dfrac{1}{2}\xi(\dfrac{y}{h_y} - j + 2)$, where the quadratic spline function ξ is defined in (1.5).

Using the functions $\phi_j^y(y)$ we can construct a set of quadratic spline basis functions, $\{\psi_j^y(y)\}_{j=1}^m$ that satisfy the homogeneous boundary conditions (1.32). They are

$$\psi_1^y(y) = \phi_1^y(y) - \phi_0^y(y);\ \psi_j^y(y) = \phi_j^y(y),\ j = 2,\ldots,m-1;\ \psi_m^y(y) = \phi_m^y(y) - \phi_{m+1}^y(y).$$

The QSC method applied to the BVP (1.29)-(1.30)-(1.31)-(1.32) using the basis functions $\{\phi_i^x(x)\psi_j^y(y)\}_{i=1,j=1}^{n,m}$ results in an almost block-tridiagonal linear system of the form

$$A^{phd}\mathrm{x} \equiv \frac{1}{8}(a\frac{1}{h_x^2}T_{-2}^p \otimes T_6^{hd} + c\frac{1}{h_y^2}T_6^p \otimes T_{-2}^{hd} + \frac{1}{8}fT_6^p \otimes T_6^{hd})\mathrm{x} = \mathrm{g} \tag{1.33}$$

where T_{-2}^{p} and T_{6}^{p} are of size $n \times n$ and their form is shown in (1.15), T_{-2}^{hd} and T_{6}^{hd} are of size $m \times m$ and their form is shown in (1.7), and

$$\mathrm{g}_k \equiv g(lh_x - \frac{h_x}{2}, jh_y - \frac{h_y}{2}),$$

for $l = 1, \ldots, n$, $j = 1, \ldots, m$, with $k = (l-1)m + j$. The operation $\otimes$ denotes the tensor product (Kronecker product) of two matrices and is defined in the following way: if $A \in \mathbb{C}^{p \times q}$ and $B \in \mathbb{C}^{m \times n}$, then the Kronecker product $A \otimes B \in \mathbb{C}^{pm \times qn}$ is the matrix with elements given by $(A \otimes B)_{(i-1)m+j,(s-1)n+k} = A_{is} B_{jk}$.

Let λ_l^p and μ_l^p, $l = 1, \ldots, n$, be the eigenvalues of the matrices T_{-2}^{p} and T_{6}^{p} (of size $n \times n$), respectively, and δ_l^p, $l = 1, \ldots, n$, the associated eigenvectors. Let also λ_j^{hd} and μ_j^{hd}, $j = 1, \ldots, m$, be the eigenvalues of the matrices T_{-2}^{hd} and T_{6}^{hd} (of size $m \times m$), respectively, and δ_j^{hd}, $j = 1, \ldots, m$, the associated eigenvectors. The eigenvalues σ_k^{phd}, $k = 1, \ldots, nm$, of the matrix A^{phd} in (1.33) are given by

$$\sigma_k^{phd} = a\frac{1}{h_x^2}\lambda_l^p \mu_j^{hd} + c\frac{1}{h_y^2}\mu_l^p \lambda_j^{hd} + \frac{1}{64} f \mu_l^p \mu_j^{hd}, \tag{1.34}$$

and its eigenvectors δ_k^{phd}, $k = 1, \ldots, nm$, by

$$\delta_k^{phd} = \delta_l^p \otimes \delta_j^{hd} \tag{1.35}$$

for $l = 1, \ldots, n$, $j = 1, \ldots, m$, with $k = (l-1)m + j$. Note that the eigenvectors δ_k^{phd} are orthogonal. If we normalise the eigenvectors δ_l^p and δ_j^{hd} by choosing the constants κ_l^p and κ_j^{hd} as in (1.18) and (1.10), respectively, then the eigenvectors δ_k^{phd}, $k = 1, \ldots, nm$, are orthonormal. In the rest of the thesis, we assume that the eigenvectors δ_k^{phd}, $k = 1, \ldots, nm$, are already normalised.

The form, eigenvalues and eigenvectors of the matrix of QSC equations arising from the discretisation of the PDE (1.29) with other types of boundary conditions, that is, any combination of homogeneous Dirichlet, periodic or homogeneous Neumann conditions in the x and y directions, are not shown here for brevity. This study, though, considers all these types of boundary conditions.

1.4 Fast Fourier and trigonometric transforms

In this section we review the Fast Fourier Transform (FFT) techniques used later in the thesis for the solution of QSC equations arising from the discretisation of certain PDEs and boundary conditions. The terminology used here follows [Loa92]. The notation used here is compatible with that used in [Loa92], except that, in this thesis, the indexing of all vectors and matrices starts from 1 and not from 0.

There are six types of transforms that are going to be used: the (plain) Discrete Fourier Transform (DFT), the Discrete Sine Transform II (DST-II), the Discrete Cosine Transform II (DCT-II), the inverse Discrete Fourier Transform (iDFT), the inverse Discrete Sine Transform II (iDST-II) and the inverse Discrete Cosine Transform II (iDCT-II). These transforms may be viewed as the computation of a matrix-vector product, each one involving a highly structured matrix.

The *Discrete Fourier Transform* is the computation of the matrix-vector product $\mathrm{y} = \mathbb{F}_n \cdot \mathrm{x}$, where $\mathrm{x} = [\mathrm{x}_1, \mathrm{x}_2, \cdots, \mathrm{x}_n]^T$ is a given vector and

$$(\mathbb{F}_n)_{lj} = e^{-2(j-1)(l-1)\pi i/n} = \cos\frac{2(j-1)(l-1)\pi}{n} - i\sin\frac{2(j-1)(l-1)\pi}{n},$$

where $i = \sqrt{-1}$. In other words, for $l = 1, \ldots, n$,

$$\mathrm{y}_l = \sum_{j=1}^{n} e^{-2(j-1)(l-1)\pi i/n}\mathrm{x}_j = \sum_{j=1}^{n}\left(\cos\frac{2(j-1)(l-1)\pi}{n} - i\sin\frac{2(j-1)(l-1)\pi}{n}\right)\mathrm{x}_j,$$

$\mathbb{F}_n$ is called the DFT matrix of size n. As noted in [Loa92], the DFT matrix is symmetric and has orthogonal columns. Moreover, the matrix $\mathbb{F}_n/\sqrt{n}$ is unitary.

It can be shown [Loa92] that, if n is a power of 2, the computation of the DFT can be performed in $2.5nlog_2n$ real floating-point operations (flops), counting additions and multiplications separately. We will refer to an algorithm that computes the DFT in $2.5nlog_2n$ flops as a *Fast Fourier Transform* (FFT). Several FFTs are described in [Loa92]. It is worth noticing that standard matrix-vector multiplication requires $2n^2$ flops to compute the DFT. It is the structure of the DFT matrix $\mathbb{F}_n$ that allows the required flops to be reduced to $2.5nlog_2n$.

The *Discrete Sine Transform II* is the computation of the matrix-vector product $y = \mathbb{S}_n \cdot x$, where $x = [x_1, x_2, \cdots, x_n]^T$ is a given vector and

$$(\mathbb{S}_n)_{lj} = \sin \frac{(2j-1)l\pi}{2n}.$$

In other words, for $l = 1, \ldots, n$,

$$y_l = \sum_{j=1}^{n} \sin(\frac{(2j-1)l\pi}{2n}) x_j.$$

We refer to $\mathbb{S}_n$ as the DST-II matrix of size n. [Note that the DST-II matrix has orthogonal rows, and with appropriate normalisation they can be made orthonormal.]

It can be shown [Loa92] that, if n is a power of 2, the computation of the DST-II can be performed in $2.5nlog_2n$ flops. An algorithm to perform the above computation is described in [Loa92]. We will refer to this algorithm as a *Fast Sine Transform II* (FST-II). The FST-II involves an FFT of size n and other operations of order less than $O(nlogn)$. For this reason, the term FFT is sometimes used in the literature to denote not only fast algorithms for the computation of the DFT, but also various fast trigonometric transforms (including the FST-II).

The *Discrete Cosine Transform II* is the computation of the matrix-vector product $y = C_n \cdot x$, where $x = [x_1, x_2, \cdots, x_n]^T$ is a given vector and

$$(C_n)_{lj} = \cos \frac{(2j-1)(l-1)\pi}{2n}.$$

In other words, for $l = 1, \ldots, n$,

$$y_l = \sum_{j=1}^{n} \cos(\frac{(2j-1)(l-1)\pi}{2n}) x_j.$$

We refer to C_n as the DCT-II matrix of size n. [Note that the DCT-II matrix has orthogonal rows, and with appropriate normalisation they can be made orthonormal.]

It can be shown [Loa92] that, if n is a power of 2, the computation of the DCT-II can be performed in $2.5n \log_2 n$ flops. An algorithm to perform the above computation is described in [Loa92]. We will refer to this algorithm as a *Fast Cosine Transform II* (FCT-II). The FCT-II involves an FFT of size n and other operations of order less than $O(nlogn)$.

The *inverse Discrete Fourier Transform* is the inverse computation of the DFT, that is, the computation of the matrix-vector product $y = \mathbb{F}_n^{-1} \cdot x$, where $x = [x_1, x_2, \cdots, x_n]^T$ is a given vector and $\mathbb{F}_n^{-1}$ the inverse of the DFT matrix.

It can be shown [Loa92] that, if n is a power of 2, the computation of the iDFT can be performed in $2.5n \log_2 n$ flops. We will refer to an algorithm that computes the iDFT in $2.5n \log_2 n$ flops as a iFFT.

The *inverse Discrete Sine Transform II* is the inverse computation of the DST-II, that is, the computation of the matrix-vector product $y = \mathbb{S}_n^{-1} \cdot x$, where $x = [x_1, x_2, \cdots, x_n]^T$ is a given vector and $\mathbb{S}_n^{-1}$ the inverse of the DST-II matrix.

It can be shown [Loa92] that, if n is a power of 2, the computation of the iDST-II can be performed in $2.5n \log_2 n$ flops. We will refer to an algorithm that computes the iDST-II in $2.5n \log_2 n$ flops as a iFST-II.

The *inverse Discrete Cosine Transform II* is the inverse computation of the DCT-II, that is, the computation of the matrix-vector product $y = \mathrm{C}_n^{-1} \cdot x$, where $x = [x_1, x_2, \cdots, x_n]^T$ is a given vector and C_n^{-1} the inverse of the DCT-II matrix.

It can be shown [Loa92] that, if n is a power of 2, the computation of the iDCT-II can be performed in $2.5n \log_2 n$ flops. We will refer to an algorithm that computes the iDCT-II in $2.5n \log_2 n$ flops as a iFCT-II.

1.5 Properties of tensor products of matrices

In this section we review some properties of tensor (Kronecker) products of matrices which we consider non-standard and will use later in the thesis. We first introduce some notation which is convenient for denoting vectors regarded as matrices. This notation follows the notation used in [Loa92].

Let $x \in \mathbb{C}^n$, with $n = n_r \cdot n_c$, where n_r and n_c are positive integers. We denote by $x_{n_r \times n_c}$ the matrix X of size $n_r \times n_c$ for which $X_{lj} = x_{(j-1)n_r+l}$, $j = 1, \ldots, n_c$, $l = 1, \ldots, n_r$, that is, each column of X is a block of n_r (consecutive) components of x.

The first two properties of tensor products of matrices appear in [Loa92] and are referred to as **Kron5** and **Kron6**.

Kron5: If $B \in \mathbb{C}^{n_r \times n_r}$, $\mathbb{I}_{n_c}$ is the identity matrix of size n_c and $\mathrm{x} \in \mathbb{C}^n$ with $n = n_r \cdot n_c$, then

$$(\mathbb{I}_{n_c} \otimes B)\mathrm{x} = \mathrm{y} \Rightarrow \mathrm{y}_{n_r \times n_c} = B \cdot \mathrm{x}_{n_r \times n_c}.$$

Kron6: If $A \in \mathbb{C}^{n_c \times n_c}$, $\mathbb{I}_{n_r}$ is the identity matrix of size n_r and $\mathrm{x} \in \mathbb{C}^n$ with $n = n_r \cdot n_c$, then

$$(A \otimes \mathbb{I}_{n_r})\mathrm{x} = \mathrm{y} \Rightarrow \mathrm{y}_{n_r \times n_c} = \mathrm{x}_{n_r \times n_c} \cdot A^T.$$

The next property is a generalisation of Kron5 and Kron6, but is not mentioned in [Loa92]. We will refer to it as **Kron56**.

Kron56: If $A \in \mathbb{C}^{n_c \times n_c}$, $B \in \mathbb{C}^{n_r \times n_r}$ and $\mathrm{x} \in \mathbb{C}^n$ with $n = n_r \cdot n_c$, then

$$(A \otimes B)\mathrm{x} = \mathrm{y} \Rightarrow \mathrm{y}_{n_r \times n_c} = B \cdot \mathrm{x}_{n_r \times n_c} \cdot A^T = B(A(\mathrm{x}_{n_r \times n_c})^T)^T.$$

Chapter 2

Fast Fourier transform solution of QSC equations

In this chapter we consider the FFT solution of QSC equations arising from the discretisation of one- and two-dimensional BVPs with constant coefficients and even order derivatives differential operators.

2.1 FFT solution of one-dimensional QSC equations

2.1.1 *The case of homogeneous Dirichlet boundary conditions*

Consider the BVP (1.3)-(1.4) and the respective discretised problem (1.6). Let Λ^{hd} be the diagonal matrix holding the eigenvalues of the matrix A^{hd} of (1.6) as defined in (1.12). Let also V^{hd} be the matrix whose columns are the normalised eigenvectors of A^{hd}, i.e.,

$$V_{jl}^{hd} = \delta_{lj}^{hd}, \text{ for } l = 1, \dots, n, \; j = 1, \dots, n, \tag{2.1}$$

with δ_{lj}^{hd} and κ_l^{hd} defined in (1.9) and (1.10), respectively.

Clearly, $(V^{hd})^{-1}A^{hd}V^{hd} = \Lambda^{hd}$, and thus $(A^{hd})^{-1} = V^{hd}(\Lambda^{hd})^{-1}(V^{hd})^{-1}$. Hence, the solution x to the linear system (1.6) can be written as

$$\mathrm{x} = V^{hd}(\Lambda^{hd})^{-1}(V^{hd})^{-1}\mathrm{g}.$$

Let D^{hd} be the diagonal matrix holding the constants κ_l^{hd} of (1.10), that is,

$$D^{hd} = diag(\sqrt{\frac{2}{n}}, \sqrt{\frac{2}{n}}, \cdots, \sqrt{\frac{2}{n}}, \sqrt{\frac{1}{n}}).$$

Note now that the matrix V^{hd} is a scaled transpose of the DST-II matrix of size n. That is,

$$V^{hd} = (\mathbb{S}_n)^T D^{hd}.$$

Recall also that V^{hd} is orthogonal. Then, $(V^{hd})^{-1} = (V^{hd})^T$, yielding $(V^{hd})^{-1} = D^{hd}\mathbb{S}_n$, and $V^{hd} = \mathbb{S}_n^{-1}(D^{hd})^{-1}$. Thus, the solution x to the linear system (1.6) can be written as

$$\mathrm{x} = \mathbb{S}_n^{-1}(\Lambda^{hd})^{-1}\mathbb{S}_n\mathrm{g}. \tag{2.2}$$

The computation involved in (2.2) can be performed in three steps:

Step 1: Perform FST-II to vector g to obtain vector $\mathrm{g}^{(1)} = \mathbb{S}_n\mathrm{g}$.

Step 2: Divide each of the components of $\mathrm{g}^{(1)}$ by the respective eigenvalue of A^{hd} to obtain vector $\mathrm{g}^{(2)} = (\Lambda^{hd})^{-1}\mathrm{g}^{(1)}$.

Step 3: Perform iFST-II to vector $\mathrm{g}^{(2)}$ to obtain x.

The above three steps form an FFT technique for solving (1.6). Clearly, the three steps involve $5nlog_2n$ (+ lower order terms of n) flops.

It is worth pointing out that a standard direct tridiagonal solver applied to the system (1.6) requires only $8n$ flops (including the LU factorisation and forward-and-back substitutions), that is, it is of lower order of computational complexity than the above FFT technique. The advantages, though, of the FFT technique become apparent when higher dimensions or periodic boundary conditions are considered (see further in this section), as well as when parallelism is to be exploited.

2.1.2 *The case of periodic boundary conditions*

Consider now the BVP (1.3)-(1.13) and the respective discretised problem (1.14). Let Λ^p be the diagonal matrix holding the eigenvalues of the matrix A^p of (1.14) as defined in (1.20). Let also V^p be the matrix whose columns are the normalised eigenvectors of A^p, i.e.,

$$V^p_{jl} = \delta^p_{lj}, \text{ for } l = 1, \dots, n, \; j = 1, \dots, n, \tag{2.3}$$

with δ^p_{lj} and κ^p_l defined in (1.17) and (1.18), respectively.

As in the case of homogeneous Dirichlet boundary conditions, the solution x to the linear system (1.14) is written as

$$\mathrm{x} = V^p(\Lambda^p)^{-1}(V^p)^{-1}\mathrm{g}.$$

Note now that the matrix V^p is a scaled conjugate transpose of the DFT matrix of size n. That is,

$$V^p = (\mathbb{F}_n)^H/\sqrt{n}.$$

Recall also that V^p is unitary. Proceeding as in the case of homogeneous Dirichlet boundary conditions, the solution x to the linear system (1.14) is written as

$$\mathrm{x} = \mathbb{F}_n^{-1}(\Lambda^p)^{-1}\mathbb{F}_n\mathrm{g}. \tag{2.4}$$

The computation involved in (2.4) can be performed in three steps:

Step 1: Perform FFT to vector g to obtain vector $\mathrm{g}^{(1)} = \mathbb{F}_n\mathrm{g}$.

Step 2: Divide each of the components of $\mathrm{g}^{(1)}$ by the respective eigenvalue of A^p to obtain vector $\mathrm{g}^{(2)} = (\Lambda^p)^{-1}\mathrm{g}^{(1)}$.

Step 3: Perform iFFT to vector $\mathrm{g}^{(2)}$ to obtain x.

The above three steps form an FFT technique for solving (1.14). Clearly, the three steps involve $5nlog_2n$ (+ lower order terms of n) flops.

It is worth pointing out that a standard direct tridiagonal solver cannot be applied to the system (1.14). We can consider, though, a 2×2 block partition of the matrix A^p,

$$A^p = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix},$$

where A_{11} is of size $(n-1) \times (n-1)$, A_{12} is of size $(n-1) \times 1$, A_{21} is of size $1 \times (n-1)$, and A_{21} is of size 1×1 ($A_{22} = [A^p_{nn}]$). Using the Schur Complement Method, and applying a direct tridiagonal solver to A_{11} twice we can solve (1.14) in $13n$ flops. This technique, though, is not straightforward parallelisable, since it involves a direct tridiagonal solver.

2.1.3 *The case of homogeneous Neumann boundary conditions*

Consider finally the BVP (1.3)-(1.21) and the respective discretised problem (1.22). Let Λ^{hn} be the diagonal matrix holding the eigenvalues of the matrix A^{hn} of (1.22) as defined in (1.28). Let also V^{hn} be the matrix whose columns are the normalised eigenvectors of A^{hn}, i.e.,

$$V^{hn}_{jl} = \delta^{hn}_{lj}, \text{ for } l = 1, \dots, n, \; j = 1, \dots, n, \tag{2.5}$$

with δ^{hn}_{lj} and κ^{hn}_l defined in (1.25) and (1.26), respectively.

As in the case of homogeneous Dirichlet boundary conditions, the solution x to the linear system (1.22) is written as

$$\mathrm{x} = V^{hn}(\Lambda^{hn})^{-1}(V^{hn})^{-1}\mathrm{g}.$$

Let D^{hn} be the diagonal matrix holding the constants κ^{hn}_l of (1.26), that is,

$$D^{hn} = diag(\sqrt{\frac{1}{n}}, \sqrt{\frac{2}{n}}, \sqrt{\frac{2}{n}}, \cdots, \sqrt{\frac{2}{n}}).$$

Note now that the matrix V^{hn} is a scaled transpose of the DCT-II matrix of size n. That is,

$$V^{hn} = (\mathrm{C}_n)^T D^{hn}.$$

Recall also that V^{hn} is orthogonal. Proceeding as in the case of homogeneous Dirichlet boundary conditions, the solution x to the linear system (1.22) is written as

$$\mathrm{x} = \mathrm{C}_n^{-1}(\Lambda^{hn})^{-1}\mathrm{C}_n \mathrm{g}. \tag{2.6}$$

The computation involved in (2.6) can be performed in three steps:

Step 1: Perform FCT-II to vector g to obtain vector $\mathrm{g}^{(1)} = \mathrm{C}_n \mathrm{g}$.

Step 2: Divide each of the components of $\mathrm{g}^{(1)}$ by the respective eigenvalue of A^{hn} to obtain vector $\mathrm{g}^{(2)} = (\Lambda^{hn})^{-1}\mathrm{g}^{(1)}$.

Step 3: Perform iFCT-II to vector $\mathrm{g}^{(2)}$ to obtain x.

The above three steps form an FFT technique for solving (1.22). Clearly, the three steps involve $5nlog_2 n$ (+ lower order terms of n) flops.

It is worth pointing out that a standard direct tridiagonal solver applied to the system (1.22) requires only $8n$ flops, but it is not straightforward parallelisable.

2.2 FFT solution of two-dimensional QSC equations

Consider again the BVP (1.29)-(1.30)-(1.31)-(1.32) and the associated discretised problem (1.33). Let Λ^{phd} be the diagonal matrix holding the eigenvalues of the matrix A^{phd} of (1.33) as defined in (1.34). Let also V^{phd} be the matrix whose columns are the normalised eigenvectors of A^{phd}, i.e.,

$$V_{qk}^{phd} = \delta_{kq}^{phd}, \text{ for } k = 1, \ldots, nm,\ q = 1, \ldots, nm,$$

with δ_{kq}^{phd} defined in (1.35). Note that $V^{phd} = V^p \otimes V^{hd}$, where V^p and V^{hd} are the matrices defined in (2.3) and (2.1), respectively.

The solution x to the linear system (1.33) can be written as

$$\begin{aligned} \mathrm{x} \;=\; & V^{phd}(\Lambda^{phd})^{-1}(V^{phd})^{-1}\mathrm{g} = (V^p \otimes V^{hd})(\Lambda^{phd})^{-1}(V^p \otimes V^{hd})^{-1}\mathrm{g} = \\ & (V^p \otimes V^{hd})(\Lambda^{phd})^{-1}((V^p)^{-1} \otimes (V^{hd})^{-1})\mathrm{g}. \end{aligned} \tag{2.7}$$

Using the properties of the matrices V^p and V^{hd} as described in the previous subsection, x can be written as

$$\mathrm{x} = (\mathbb{F}_n^{-1} \otimes \mathbb{S}_m^{-1})(\Lambda^{phd})^{-1}(\mathbb{F}_n \otimes \mathbb{S}_m)\mathrm{g}. \tag{2.8}$$

Taking into account property **Kron56** of Section 1.5, the computation involved in (2.8) can be performed in the following steps:

Step 1: Perform FFT to each of the m columns of the vector $(\mathrm{g}_{m\times n})^T$ to obtain $(\mathrm{g}^{(1)})_{n\times m} = \mathbb{F}_n(\mathrm{g}_{m\times n})^T$.

Step 2: Perform FST-II to each of the n columns of the vector $((\mathrm{g}^{(1)})_{n\times m})^T$ to obtain $(\mathrm{g}^{(2)})_{m\times n} = \mathbb{S}_m((\mathrm{g}^{(1)})_{n\times m})^T$, or, equivalently, $\mathrm{g}^{(2)} = (\mathbb{F}_n \otimes \mathbb{S}_m)\mathrm{g}$.

Step 3: Divide each of the components of $\mathrm{g}^{(2)}$ by the respective eigenvalue of A^{phd} to obtain vector $\mathrm{g}^{(3)} = (\Lambda^{phd})^{-1}\mathrm{g}^{(2)}$.

Step 4: Perform iFST-II to each of the n columns of the vector $((\mathrm{g}^{(3)})_{n\times m})^T$ to obtain $(\mathrm{g}^{(4)})_{m\times n} = \mathbb{S}_m^{-1}((\mathrm{g}^{(3)})_{n\times m})^T$.

Step 5: Perform iFFT to each of the m columns of the vector $((\mathrm{g}^{(4)})_{m\times n})^T$ to obtain $\mathrm{x}_{n\times m} = \mathbb{F}_n^{-1}((\mathrm{g}^{(4)})_{m\times n})^T$, or, equivalently, $\mathrm{x} = (\mathbb{F}_n^{-1} \otimes \mathbb{S}_m^{-1})(\Lambda^{phd})^{-1}(\mathbb{F}_n \otimes \mathbb{S}_m)\mathrm{g}$.

The above five steps form an FFT technique for solving (1.33). Clearly, the five steps involve $5nmlog_2(nm)$ (+ lower order terms of nm) flops.

2.3 Performance results

In this section, results from numerical experiments are presented that demonstrate the performance of the FFT solvers for the two-dimensional QSC equations.

For the implementation of the FFT, iFFT, FST-II and iFST-II a package developed at the NEC Systems Laboratory was used [Mos92]. The routines of the package implement the Stockham algorithm, using only real operations and omitting any redundant operations with zero imaginary parts. The package provides an option for multiple FFTs, such as those required for Steps 1, 2, 4 and 5 of the previous subsection. With this option, multiple FFTs, i.e. FFTs applied to a set of vectors, are handled with one call to a routine which takes as argument the set of vectors stored in a one-dimensional array, one after another. From the description of Steps 1 and 2 in the previous subsection, it is clear that, in order to proceed from Step 1 to Step 2, the two-dimensional array $(\mathrm{g}^{(1)})_{n\times m}$ corresponding to vector $\mathrm{g}^{(1)}$ needs to be transposed. This is handled by the package by taking an appropriate stride between the elements of the one-dimensional array corresponding to $\mathrm{g}^{(1)}$. The handling of the rest of the multiple FFTs required by Steps 4 and 5 of the previous subsection is similar.

The results shown in Table 2.1, correspond to Problem 1 of the Appendix. The error and convergence results are presented in order to verify the performance of the QSC method, as it is described in [Chr94]. For all computations we have used double precision. For each discretisation grid size shown in Table 2.1, the "global error"corresponds to the maximum absolute value of the error of the QSC approximation on a uniform grid of 20×20 points, while the "error on grid points" corresponds to the maximum absolute value of the error of the QSC approximation on the discretisation grid. The "global error" is taken as an approximation of the uniform norm of the error. The results show that the QSC method is globally of third order, while on the grid points it is of fourth order, as expected.

In the same table, we show the time in seconds required for the FFT solution of the QSC equations corresponding to Problem 1. The timing results were obtained on a Sparcstation20. Figure 2.1 shows graphically the timing results of Table 2.1. If we use constant least-squares to approximate the data of Figure 2.1 by a function of the form $\rho n^2 \log_2 n$ then $\rho = 1.82\times 10^{-6}$. If we use linear least-squares to approximate the data of Figure 2.1 by a function of the form $\rho_2 n^{\rho_1}$ then $\rho_1 = 2.1$ and $\rho_2 = 8\times 10^{-6}$.

	Error				Order of Convergence				Time
	First step		Second step		First step		Second step		
n	Global error	Error on grid	Global error	Error on grid	Global	On grid	Global	On grid	
32	$2.77E-05$	$2.82E-05$	$1.23E-05$	$2.91E-06$	2.39	2.39	2.65	4.01	0.013
64	$6.41E-06$	$6.51E-06$	$1.86E-06$	$1.81E-07$	2.11	2.11	2.72	4.00	0.051
128	$1.57E-06$	$1.59E-06$	$1.57E-07$	$1.13E-08$	2.03	2.03	3.56	4.00	0.223
256	$3.90E-07$	$3.96E-07$	$2.29E-08$	$7.08E-10$	2.01	2.01	2.78	4.00	0.953

Table 2.1: Errors and respective orders of convergence corresponding to Problem 1 discretised by the QSC method, for several grid sizes n and for both steps of the QSC method. Time in seconds for the FFT solution of the QSC equations for several grid sizes $n \times n$ and for one step of the QSC method.

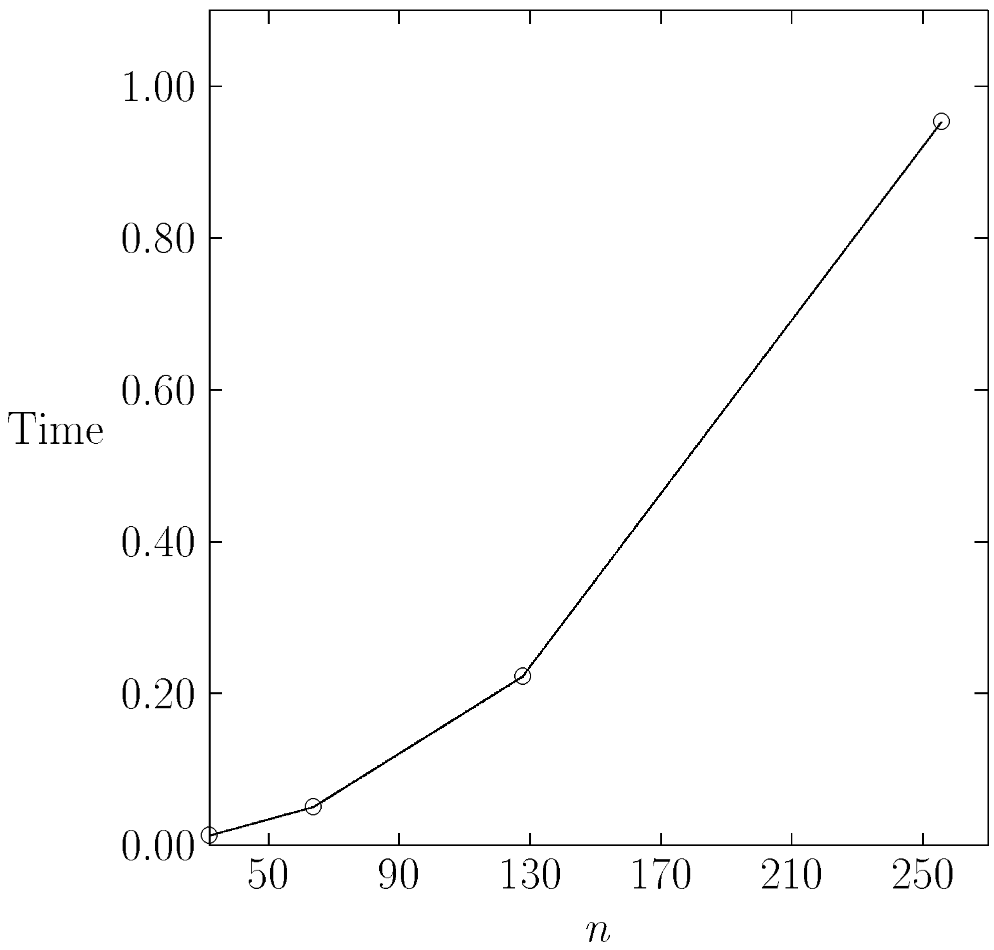

Figure 2.1: Plot of time in seconds for the FFT solution of the QSC equations corresponding to Problem 1 versus the grid size n in one-dimension. The slope of the line in a log-log plot, computed by linear least squares fit to the data, is 2.1.

Chapter 3

The variable coefficient case

In this chapter we consider the FFT solution of QSC equations arising from the discretisation of two-dimensional BVPs with variable coefficient differential operators.

The BVPs considered are defined by the PDE

$$\begin{aligned}\mathbf{L}u(x,y) \equiv a(x,y)u_{xx} + b(x,y)u_{xy} + c(x,y)u_{yy} + \\ + d(x,y)u_x + e(x,y)u_y + f(x,y)u = g(x,y) \text{ in } \Omega \equiv (0,1)\times(0,1)\end{aligned} \quad (3.1)$$

subject to boundary conditions on the boundary of the domain Ω. At each line of the boundary of Ω the boundary conditions may be of the following types: *homogeneous Dirichlet*, *homogeneous Neumann*, or *periodic*. Here are some examples of complete sets of boundary conditions considered:

- *Homogeneous Dirichlet on all sides of the boundary,*

$$\begin{aligned}\mathbf{B}u(x,y) \equiv u = 0 \text{ on } x=0,\ x=1,\ \text{for } 0\le y\le 1 \\ \text{and on } y=0,\ y=1,\ \text{for } 0\le x\le 1\end{aligned} \quad (3.2)$$

- *Homogeneous Neumann on all sides of the boundary,*

$$\begin{aligned}\mathbf{B}u(x,y) \equiv u_n = 0 \text{ on } x=0,\ x=1,\ \text{for } 0\le y\le 1 \\ \text{and on } y=0,\ y=1,\ \text{for } 0\le x\le 1\end{aligned} \quad (3.3)$$

- *Periodic in both directions*,

$$\mathbf{B}_1 u(x,y) \equiv u(0,y) = u(1,y), \text{ for } 0 \le y \le 1 \tag{3.4}$$

$$\mathbf{B}_2 u(x,y) \equiv u_x(0,y) = u_x(1,y), \text{ for } 0 \le y \le 1 \tag{3.5}$$

$$\mathbf{B}_3 u(x,y) \equiv u(x,0) = u(x,1), \text{ for } 0 \le x \le 1 \tag{3.6}$$

$$\mathbf{B}_4 u(x,y) \equiv u_y(x,0) = u_y(x,1), \text{ for } 0 \le x \le 1 \tag{3.7}$$

- *Combinations* of the above, e.g. periodic boundary conditions in the x direction and homogeneous Dirichlet in the y, i.e. those described by (1.30)-(1.31)-(1.32).

The methods described in this thesis are applicable to BVPs with any combination of the above types of boundary conditions.

Consider a BVP defined by (3.1) subject to any of the boundary conditions indicated above (e.g., (1.30)-(1.31)-(1.32)) and a rectangular partition Δ of Ω with uniform step sizes $h_x = \dfrac{1}{n}$ and $h_y = \dfrac{1}{m}$ in the x and y directions respectively. Consider also a set of basis functions for the bi-quadratic spline space constructed on the partition Δ of Ω, so that they satisfy the boundary conditions of the BVP. The QSC method applied to the BVP using basis functions which satisfy the boundary conditions by construction results in a (usually block tridiagonal or almost block tridiagonal) linear system

$$A\mathrm{x} = \mathrm{g} \tag{3.8}$$

where A is of size $nm \times nm$. Due to the variable coefficients of the PDE operator in (3.1), the matrix A can no longer be written in tensor product form, as, for example, in (1.33). For the same reason, the eigenvalues of A are not known in analytic form. In this chapter we study preconditioned iterative methods for solving (3.8). The preconditioners considered are based on matrices that arise from the discretisation of BVPs with constant coefficients by QSC. These preconditioners are solved by the FFT techniques described in the previous chapter.

3.1 Preconditioners for QSC equations

Consider the QSC equations (3.8) arising from the discretisation of PDE (3.1) subject to any of the types of boundary conditions described above. An obvious choice of preconditioner for these QSC equations is the matrix L arising from the discretisation of the Laplace operator subject to the same type of boundary conditions. We refer to this type of preconditioner as *Laplace preconditioner.*

If the boundary conditions are purely periodic, i.e., those described by (3.4)-(3.5)-(3.6)-(3.7), the matrix arising from the discretisation of the Laplace operator is singular. The matrix arising from the discretisation of the BVP, though, may be uniquely solvable, if the zeroth or first order terms of the operator are non-zero at at least one collocation point.

Similarly, if the boundary conditions are purely Neumann, i.e. those described by (3.3), the matrix arising from the discretisation of the Laplace operator is singular. The matrix arising from the discretisation of the BVP, though, may be uniquely solvable, if the zeroth order term of the operator is non-zero at at least one collocation point. In order to be able to apply the preconditioner to all the types of boundary conditions considered we choose as preconditioner the matrix H arising from the discretisation of the Helmholtz operator

$$u_{xx} + u_{yy} - u \tag{3.9}$$

subject to the same boundary conditions as the original BVP. We refer to this type of preconditioner as *Helmholtz (H) preconditioner.*

Greenbaum [Gre92] studies preconditioners of the form $P = SLS$, for any positive definite diagonal matrix S. Numerical experiments indicate that, if the coefficients of the PDE operator are smooth, the diagonal matrix S which minimises the condition number of the preconditioned matrix $P^{-1}A$ is the square root of the diagonal D of A. Since, in that paper, Greenbaum deals only with self-adjoint PDE operators, homogeneous Dirichlet boundary conditions and discretisation methods, which give rise to symmetric positive definite matrices A, it is natural to consider such preconditioners. In the case of the spline collocation discretisation method, though, the resulting matrix A for the class of problems considered in [Gre92] is not necessarily symmetric positive definite. Moreover, in this thesis, we deal with a broader class of problems, which are not necessarily self-adjoint and include several types of boundary conditions. Therefore, we study preconditioners of three forms:

a) $P = \sqrt{D} H \sqrt{D}$,

b) $P = DH$,

c) $P = HD$.

We refer to the first type of preconditioner as the *DHD preconditioner*, to the second type of preconditioner as the *DH preconditioner*, and to the third type of preconditioner as the *HD preconditioner*. These preconditioners are tested with various iterative methods, including one-step methods and acceleration techniques.

3.2 Iterative methods for QSC equations

Consider a linear system such as (3.8). The kth iteration of the preconditioned damped *Richardson (RICH) method* with damping factor τ and preconditioner P is described by the relations

$$\mathrm{r}^{(k)} = \mathrm{g} - A\mathrm{x}^{(k)} \tag{3.10}$$

$$\mathrm{x}^{(k+1)} = \mathrm{x}^{(k)} + \tau P^{-1}\mathrm{r}^{(k)}, \tag{3.11}$$

where $\mathrm{x}^{(k)}$ and $\mathrm{r}^{(k)}$ are the solution and residual vectors at the k-th iteration respectively. Bialecki [Bia94] studies Laplace preconditioned RICH methods for the solution of self-adjoint elliptic BVPs with homogeneous Dirichlet boundary conditions, discretised by Hermite cubic spline collocation. The damping factor τ depends on the coefficients of the PDE operator. Bialecki gives a simple relation that approximates the optimal damping factor.

In the same paper, Bialecki studies Laplace preconditioned *Minimum Residual (MRES) methods* arising from applying the Minimum Residual method [EES83] to the preconditioned linear system

$$\hat{A}\hat{\mathrm{x}} = \hat{\mathrm{g}} \tag{3.12}$$

where $\hat{A} = P^{-1/2}AP^{-1/2}$, $\hat{\mathrm{x}} = P^{1/2}\mathrm{x}$ and $\hat{\mathrm{g}} = P^{-1/2}\mathrm{g}$. Thus, the kth iteration of the preconditioned MRES method with preconditioner P is described by the relations

$$\mathrm{r}^{(k)} = \mathrm{g} - A\mathrm{x}^{(k)} \tag{3.13}$$

$$\mathrm{x}^{(k+1)} = \mathrm{x}^{(k)} + \tau^{(k)}P^{-1}\mathrm{r}^{(k)} \tag{3.14}$$

$$\tau^{(k)} = \frac{(A\mathrm{w}^{(k)}, \mathrm{w}^{(k)})}{(A\mathrm{w}^{(k)}, \mathrm{u}^{(k)})},\ P\mathrm{w}^{(k)} = \mathrm{r}^{(k)},\ P\mathrm{u}^{(k)} = A\mathrm{w}^{(k)}. \tag{3.15}$$

This is a one-step gradient method and makes minimal use of storage per iteration.

In order to accelerate convergence, one could use a multi-step method for non-symmetric systems, for example, GMRES [SaSc86] or BiCGStab [vdV92]. In our study, we have considered several such methods.

3.3 Numerical results.

We present numerical results that demonstrate the convergence and overall performance of the Fourier preconditioned iterative solvers discussed in Section 3.2. The preconditioners considered are those discussed in Section 3.1. The stopping criterion used in all experiments was the relative Euclidean norm of the residual and the tolerance was set to 10^{-6} for the first step of the QSC method and to 10^{-4} for the second step. The initial guess for the solution vector of the first step was the zero vector, while that for the solution vector of the second step was the already computed solution vector of the first step. For the implementation of the acceleration methods, e.g., GMRES and BiCGStab, the KSP package [GrSm93] was used. For the rest of the methods we used our own code in FORTRAN. All results were obtained on a Sparcstation20 in double precision. The maximum number of previous iteration vectors that GMRES takes into account in one iteration, i.e., the "restart", was set to 20, for all tests performed.

Consider Problem 2 of the Appendix. This is a variable coefficient problem, for which the u_{xx} coefficient at the collocation points differs, in general, from the u_{yy} one. The range of both coefficients is (1/3, 1). Table 3.1 shows the number of iterations and the respective time required for the convergence of the GMRES, BiCGStab, MRES and RICH methods, with various preconditioners applied to the QSC equations arising from the discretisation of Problem 2.

method	GMRES															
precond.	H		DHD		DH		HD		H		DHD		DH		HD	
	no. of iter.								time							
	step		step		step		step		step 1		step 1		step 1		step 1	
n	1	2	1	2	1	2	1	2	total	per iter.	total	per iter.	total	per iter.	total	per iter.
32	6	4	11	7	9	5	12	8	0.214	0.0356	0.391	0.0355	0.355	0.0395	0.443	0.0369
64	6	4	11	7	9	5	12	8	0.880	0.1466	1.713	0.1557	1.424	0.1582	1.797	0.1498
128	6	4	11	7	9	6	12	8	3.688	0.6147	6.911	0.6282	5.956	0.6618	7.552	0.6294
256	6	5	11	8	9	7	12	9	13.589	2.2648	30.716	2.7923	24.456	2.7174	31.400	2.6167

method	BiCGStab															
precond.	H		DHD		DH		HD		H		DHD		DH		HD	
	no. of iter.								time							
	step		step		step		step		step 1		step 1		step 1		step 1	
n	1	2	1	2	1	2	1	2	total	per iter.	total	per iter.	total	per iter.	total	per iter.
32	3	2	4	3	4	3	5	3	0.228	0.0761	0.288	0.0721	0.303	0.0758	0.345	0.06911
64	3	2	4	3	4	3	5	3	0.907	0.3022	1.184	0.2960	1.211	0.3027	1.381	0.2762
128	3	2	4	4	4	3	5	4	3.804	1.2682	5.037	1.2594	5.094	1.2736	5.710	1.1419
256	3	2	4	4	4	4	5	4	15.949	5.3162	21.398	5.3495	21.169	5.2922	23.931	4.7861

Table 3.1: Number of iterations and time in seconds required for the convergence of various preconditioned iterative techniques for the solution of the QSC equations arising from the discretisation of Problem 2 for several grid sizes n, and for both steps of the QSC method.

method	MRES															
precond.	H		DHD		DH		HD		H		DHD		DH		HD	
	no. of iter.								time							
	step		step		step		step		step 1		step 1		step 1		step 1	
n	1	2	1	2	1	2	1	2	total	per iter.	total	per iter.	total	per iter.	total	per iter.
32	7	4	15	9	11	6	16	10	0.121	0.0174	0.477	0.0318	0.340	0.0309	0.472	0.02951
64	7	4	16	10	11	6	16	11	0.515	0.0735	2.133	0.1333	1.443	0.1312	1.949	0.1218
128	7	5	16	11	11	8	17	11	2.513	0.3590	8.816	0.5510	6.216	0.5651	8.743	0.5143
256	7	5	17	12	11	10	17	12	11.428	1.6326	30.661	1.8036	26.064	2.3695	36.415	2.1420

Table 3.1: Continued.

method	RICH			
precond.	H			
	no. of iter.		time	
	step		step 1	
n	1	2	total	per iter.
32	7	4	0.218	0.03121
64	7	4	0.883	0.1262
128	7	5	3.724	0.5320
256	7	6	15.855	2.2650

Table 3.1: Continued.

First note that the number of iterations required for the convergence of the methods above is independent of the grid size n for all preconditioners and iterative techniques. Since each iteration takes $O(n^2 \log n)$ time, the total computational effort required for the solution of the QSC equations is also $O(n^2 \log n)$, which is asymptotically close to the optimal $O(n^2)$ behaviour.

Comparing the two acceleration methods, BiCGStab and GMRES, considered, BiCGStab clearly requires fewer iterations than GMRES, for the same preconditioner, but does not necessarily take less time. This happens because BiCGStab requires two matrix-vector multiplications per iteration, while GMRES needs only one. In some cases, BiCGStab slightly outperforms GMRES, in some other, the opposite happens. In general, the two methods are roughly equivalent as far as efficiency is concerned. MRES requires more iterations than either BiCGStab or GMRES, for the same preconditioner. Concerning efficiency, MRES outperforms BiCGStab and GMRES for the preconditioner H and falls just slightly behind both methods for the other preconditioners. It is worth noting that MRES is an one-step method and requires less storage per iteration than GMRES. With respect to efficiency, the RICH method with the damping factor chosen as suggested in [Bia94] is comparable to BiCGStab for the preconditioner H.

Among the four types of preconditioners considered, for this problem, the H preconditioner outperforms the rest. This was not generally the case for other problems tested, as will be shown later in this chapter. It is worth noting also that the RICH method did not converge with the DH, DHD and HD preconditioners. This is not surprising, since the role of scaling is played by the damping factor τ, so no additional scaling makes sense.

Consider next Problem 3 of the Appendix. This is a variable coefficient problem, for which the u_{xx} and u_{yy} coefficients are equal, but they vary a lot. In order to investigate the effect of the variation of the dominant coefficients to the convergence rate of the methods we introduce a parameter δ. With δ= 1 the coefficients' range is roughly (1, 21), with $\delta = 5$ roughly (1, 100), and with $\delta = 10$ roughly (1, 200). Table 3.2 shows the number of iterations required for the convergence of the GMRES and MRES methods, with various preconditioners applied to the QSC equations arising from the discretisation of Problem 3, for three values of the parameter δ.

For this problem, we note that the convergence rates of GMRES and MRES with the H and DH preconditioners are (almost) independent of the grid size n. The methods with the DHD and HD preconditioners either do not converge, or, in a few cases and for small grid sizes only, converge very slowly. The convergence rate of all methods, though, except those with the DH preconditioner, depends on δ. In this problem the superiority of the DH preconditioner is evident. Though we do not include timing results in the table, we should note that GMRES slightly outperforms MRES. As an indication, we mention that for the first step of the QSC method, with δ=1, and a 128×128 grid, for the DH preconditioner, GMRES takes 3.688 seconds, while MRES 4.393 .

Consider now Problem 4 of the Appendix. This problem is a modification of the problem used in [Bia94]. We have added non-zero coefficients for u_x and u_y. The ranges of the u_{xx}, u_{yy}, u_x and u_y coefficients are roughly (1, 2.7), (1, 1.8), (3.9e-3, 2.7) and (3.9e-3, 2.7), respectively. Table 3.3 shows the number of iterations required for the convergence of the GMRES, MRES and RICH methods, with various preconditioners applied to the QSC equations arising from the discretisation of Problem 4.

method	GMRES																							
precond.	H		DHD		DH		HD		H		DHD		DH		HD		H		DHD		DH		HD	
	$\delta = 1$								$\delta = 5$								$\delta = 10$							
	step																							
n	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
32	28	17	18	14	6	5	62	44	51	29	32	19	6	5	-	-	64	36	39	29	6	5	-	-
64	28	17	39	27	6	5	-	-	52	32	-	-	6	5	-	-	66	40	-	-	6	5	-	-
128	28	18	-	-	6	5	-	-	52	35	-	-	6	5	-	-	66	46	-	-	6	5	-	-
256	28	18	-	-	6	5	-	-	52	39	-	-	6	5	-	-	66	55	-	-	6	5	-	-

method	MRES																							
precond.	H		DHD		DH		HD		H		DHD		DH		HD		H		DHD		DH		HD	
	$\delta = 1$								$\delta = 5$								$\delta = 10$							
	step																							
n	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
32	38	32	-	-	8	6	-	-	100	40	-	-	8	5	-	-	99	61	-	-	8	5	-	-
64	42	32	-	-	8	6	-	-	120	54	-	-	9	5	-	-	109	70	-	-	9	5	-	-
128	42	38	-	-	8	6	-	-	108	98	-	-	6	5	-	-	132	144	-	-	9	5	-	-
256	46	42	-	-	8	6	-	-	144	142	-	-	8	6	-	-	182	-	-	-	9	5	-	-

Table 3.2: Number of iterations required for the convergence of various preconditioned iterative techniques for the solution of the QSC equations arising from the discretisation of Problem 3 for several grid sizes n, for three values of the parameter δ, and for both steps of the QSC method. A '-' denotes that the method did not converge after 200 iterations.

method	GMRES								MRES								RICH	
precond.	H		DHD		DH		HD		H		DHD		DH		HD		H	
	step																	
n	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
32	11	8	7	5	7	5	7	5	15	10	9	6	7	5	8	5	16	11
64	11	8	7	5	7	5	7	5	16	11	9	6	7	5	8	6	16	11
128	11	8	8	5	7	5	8	6	17	12	9	6	7	5	9	6	16	11
256	11	8	8	5	7	5	8	6	17	13	10	6	7	6	9	6	16	11

Table 3.3: Number of iterations required for the convergence of various preconditioned iterative techniques for the solution of the QSC equations arising from the discretisation of Problem 4 for several grid sizes n, and for both steps of the QSC method.

For this problem, the convergence rate of all methods is independent of the grid size n. The DH preconditioner clearly outperforms the rest for both the MRES and GMRES methods. Though the timing results are not listed, the MRES method with the DH preconditioner is obviously the fastest for this problem, since it usually takes the same number of iterations as the GMRES method with the DH preconditioner, and its iterations are less costly in computational effort than those of GMRES. The RICH method for this problem clearly falls behind in convergence rate and efficiency. As an indication, we mention that for the first step of the QSC method, with a 128×128 grid, for the DH preconditioner, GMRES takes 5.440 seconds and MRES 5.426 while for the H preconditioner, the RICH method takes 10.174 seconds.

Figure 3.1 plots the time in seconds taken by three solvers applied to the QSC equations corresponding to Problem 4, versus the grid size n. The time is the sum of the times taken by the two steps of the QSC method. The three solvers compared are a) the GMRES method with the DH preconditioner and FFT solution (GMRES-DH-FFT), b) the MRES method with the DH preconditioner and FFT solution (MRES-DH-FFT), and c) the GMRES method with a coarse grid and Jacobi preconditioning (GMRES-MGJ). The coarse grid and Jacobi preconditioning technique is developed and analysed in [ChSm95].

The differences in the timing results of the three solvers are minor. For Problem 4, and for the largest grid, the MRES-DH-FFT solver is in absolute terms the fastest solver and GMRES-DH-FFT the slowest one. The ranking though between these two solvers varies with the BVP, as seen for example in Table 3.1. Applying a linear least squares fit to the timing data of Figure 3.1, we see that there are minor differences in the slopes of the lines plotted. All three solvers appear asymptotically optimal.

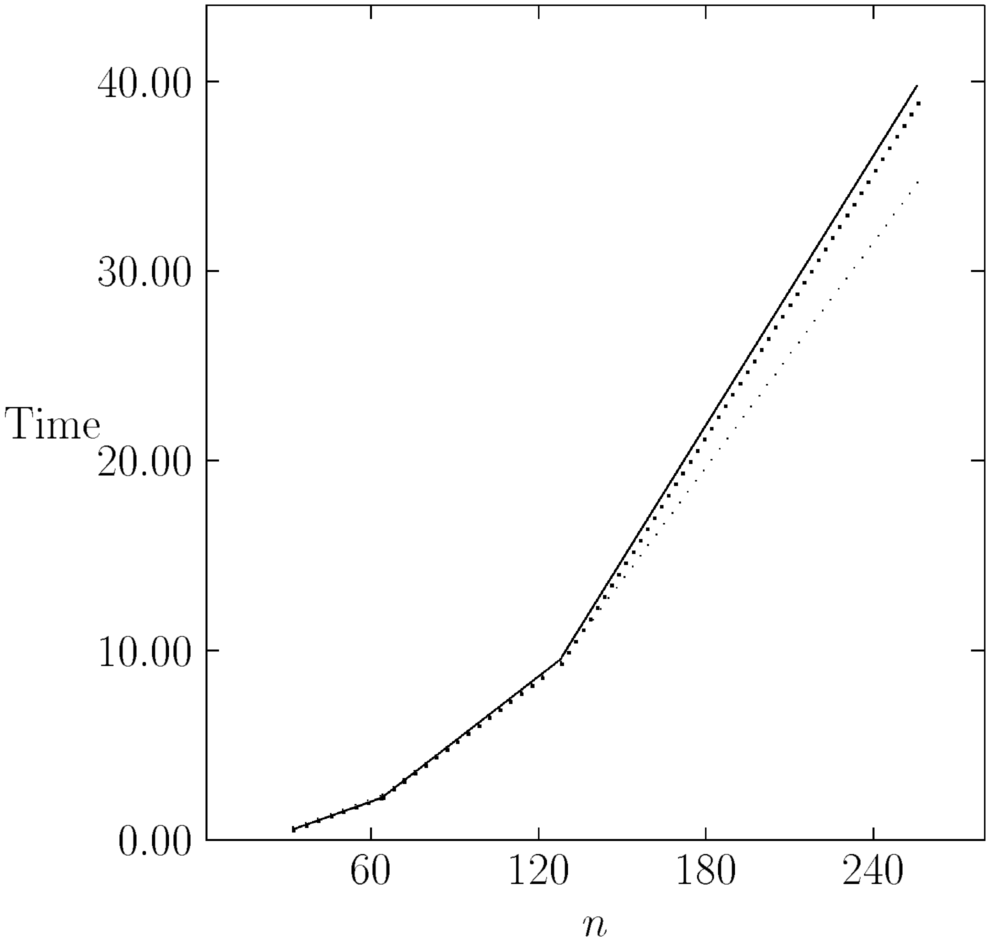

Figure 3.1: Plot of time in seconds for both steps of the QSC method versus the grid size n in one dimension for three solvers applied to the QSC equations corresponding to Problem 4. The solid line corresponds to GMRES-DH-FFT, the heavy dotted line to MRES-DH-FFT and the dotted line to GMRES-MGJ. The slopes of the three lines in a log-log plot, computed by a linear least squares fit to the data, are: GMRES-DH-FFT: 2.05, MRES-DH-FFT: 2.08 and GMRES-MGJ: 1.9.

Finally, consider Problem 5 of the Appendix. This problem has a non-zero cross-derivative term, which is of the same order as the dominant terms in the PDE, i.e., u_{xx} and u_{yy}. The discretisation of this problem by QSC gives rise to a highly non-symmetric and non-diagonally dominant matrix. Table 3.4 shows the number of iterations and time in seconds required for the convergence of GMRES with various preconditioners applied to the QSC equations arising from the discretisation of Problem 5. From the results of Table 3.4 it is noted that the convergence rate of GMRES remains invariant with the grid size n and that the DH and DHD preconditioners outperform the rest.

method	GMRES															
precond.	H		DHD		DH		HD		H		DHD		DH		HD	
	no. of iter.								time							
	step		step		step		step		step 1		step 1		step 1		step 1	
n	1	2	1	2	1	2	1	2	total	per iter.	total	per iter.	total	per iter.	total	per iter.
32	13	9	8	6	8	6	9	7	0.479	0.0369	0.301	0.0376	0.320	0.0400	0.345	0.03831
64	13	9	8	6	8	6	10	7	1.932	0.1486	1.301	0.1627	1.257	0.1571	1.581	0.1581
128	13	9	8	6	8	6	10	7	8.309	0.6392	5.473	0.6842	5.589	0.6986	6.543	0.6543
256	13	9	8	6	8	6	10	8	33.595	2.5842	22.796	2.8494	22.804	2.8505	27.438	2.7438

Table 3.4: Number of iterations and time in seconds required for the convergence of GMRES for the solution of the QSC equations arising from the discretisation of Problem 5 for several grid sizes n, and for both steps of the QSC method.

Chapter 4

Three-dimensional problems

In this chapter the FFT solution of QSC equations arising from the discretization of three-dimensional BVPs is studied. First, the case of differential operators with even order derivatives and constant coefficients is considered. Then, the study is extended to more general three-dimensional BVPs.

4.1 Spectral properties of the three-dimensional QSC matrix

Consider the PDE

$$\mathbf{L}u(x,y,z) \equiv au_{xx} + cu_{yy} + pu_{zz} + fu = g(x,y,z)$$
$$\text{in } \Omega \equiv (0,1)\times(0,1)\times(0,1) \quad (4.1)$$

subject to *periodic* boundary conditions in the x direction

$$\mathbf{B}_1 u(x,y,z) \equiv u(0,y,z) = u(1,y,z), \text{ for } 0 \le y \le 1 \text{ and } 0 \le z \le 1 \text{ and} \quad (4.2)$$

$$\mathbf{B}_2 u(x,y,z) \equiv u_x(0,y,z) = u_x(1,y,z), \text{ for } 0 \le y \le 1 \text{ and } 0 \le z \le 1, \quad (4.3)$$

homogeneous Neumann boundary conditions in the y direction

$$\mathbf{B}_3 u(x,y,z) \equiv u_n = 0 \text{ on } y=0,\ y=1, \text{ for } 0 \le x \le 1 \text{ and } 0 \le z \le 1 \quad (4.4)$$

and *homogeneous Dirichlet* boundary conditions in the z direction

$$\mathbf{B}_4 u(x,y,z) \equiv u = 0 \text{ on } z=0,\ z=1,\ \text{for } 0 \le x \le 1 \text{ and } 0 \le y \le 1 \qquad (4.5)$$

where a, c, p and f are constants.

It is worth noting that the boundary conditions described by (4.2)-(4.5) are taken as an example of types of boundary conditions considered. The FFT solvers described in this section are applicable to PDEs with any combination of periodic, homogeneous Dirichlet and homogeneous Neumann boundary conditions.

A rectangular partition Δ of Ω with uniform step sizes $h_x = \frac{1}{n}$, $h_y = \frac{1}{m}$ and $h_z = \frac{1}{l}$ in the x, y and z directions respectively is considered. A set of basis functions for the tri-quadratic spline space constructed on the partition Δ of Ω is the tensor product of quadratic B-splines $\{\phi_i^x(x)\phi_j^y(y)\phi_k^z(z)\}_{i=0,j=0,k=0}^{n+1,m+1,l+1}$ with $\phi_i^x(x) = \frac{1}{2}\xi(\frac{x}{h_x} - i + 2)$, $\phi_j^y(y) = \frac{1}{2}\xi(\frac{y}{h_y} - j + 2)$ and $\phi_k^z(z) = \frac{1}{2}\xi(\frac{z}{h_z} - k + 2)$, where the quadratic spline function ξ is defined in (1.5).

Using the functions $\phi_j^y(y)$ we can construct a set of quadratic spline basis functions, $\{\theta_j^y(y)\}_{j=1}^m$, that satisfy the homogeneous Neumann boundary conditions (4.4). They are

$$\theta_1^y(y) = \phi_1^y(y) + \phi_0^y(y);\ \theta_j^y(y) = \phi_j^y(y),\ j = 2,\ldots,m-1;\ \theta_m^y(y) = \phi_m^y(y) + \phi_{m+1}^y(y).$$

Moreover, using the functions $\phi_k^z(z)$ we can construct a set of quadratic spline basis functions, $\{\psi_k^z(z)\}_{j=1}^l$, that satisfy the homogeneous Dirichlet boundary conditions (4.5). They are

$$\psi_1^z(z) = \phi_1^z(z) - \phi_0^z(z);\ \psi_k^z(z) = \phi_k^z(z),\ k = 2,\ldots,l-1;\ \psi_l^z(z) = \phi_l^z(z) - \phi_{l+1}^z(z).$$

Note also that the set of basis functions $\{\phi_i^x(x)\}_{i=1}^n$ satisfies the periodic conditions (4.2)-(4.3). Thus, the set of basis functions $\{\phi_i^x(x)\theta_j^y(y)\psi_k^z(z)\}_{i=1,j=1,k=1}^{n,m,l}$ for the tri-quadratic spline space satisfies the boundary conditions (4.2)-(4.5).

Using the above set of basis functions, the QSC method applied to the BVP

(4.1)-(4.5) results in a linear system of the form

$$A^{phnhd}\mathrm{x} \equiv \frac{1}{64}\left(a\frac{1}{h_x^2}T_{-2}^{p}\otimes T_6^{hn}\otimes T_6^{hd} + c\frac{1}{h_y^2}T_6^{p}\otimes T_{-2}^{hn}\otimes T_6^{hd} + \right.$$
$$\left. + p\frac{1}{h_z^2}T_6^{p}\otimes T_6^{hn}\otimes T_{-2}^{hd} + \frac{1}{8}fT_6^{p}\otimes T_6^{hn}\otimes T_6^{hd}\right)\mathrm{x} = \mathrm{g} \quad (4.6)$$

where T_{-2}^p and T_6^p are of size $n \times n$ and their form is shown in (1.15), T_{-2}^{hn} and T_6^{hn} are of size $m \times m$ and their form is shown in (1.23) and T_{-2}^{hd} and T_6^{hd} are of size $l \times l$ and their form is shown in (1.7). The right side vector of (4.6) is given by

$$\mathrm{g}_{k'} \equiv g\left(ih_x - \frac{h_x}{2}, jh_y - \frac{h_y}{2}, kh_z - \frac{h_z}{2}\right), \quad (4.7)$$

for $i = 1, \ldots, n$, $j = 1, \ldots, m$, $k = 1, \ldots, l$, with $k' = (i-1)ml + (j-1)l + k$.

Let λ_i^p and μ_i^p, $i = 1, \ldots, n$, be the eigenvalues of the matrices T_{-2}^p and T_6^p (of size $n \times n$), respectively, and δ_i^p, $i = 1, \ldots, n$, the associated eigenvectors. Let λ_j^{hn} and μ_j^{hn}, $j = 1, \ldots, m$, be the eigenvalues of the matrices T_{-2}^{hn} and T_6^{hn} (of size $m \times m$), respectively, and δ_j^{hn}, $j = 1, \ldots, m$, the associated eigenvectors. Let also λ_k^{hd} and μ_k^{hd}, $k = 1, \ldots, l$, be the eigenvalues of the matrices T_{-2}^{hd} and T_6^{hd} (of size $l \times l$), respectively, and δ_k^{hd}, $k = 1, \ldots, l$, the associated eigenvectors. Then, the eigenvalues $\sigma_{k'}^{phnhd}$, $k' = 1, \ldots, nml$, of the matrix A^{phnhd} in (4.6) are given by

$$\sigma_{k'}^{phnhd} = \frac{1}{64}\left(a\frac{1}{h_x^2}\lambda_i^p\mu_j^{hn}\mu_k^{hd} + c\frac{1}{h_y^2}\mu_i^p\lambda_j^{hn}\mu_k^{hd} + p\frac{1}{h_z^2}\mu_i^p\mu_j^{hn}\lambda_k^{hd} + \frac{1}{8}f\mu_i^p\mu_j^{hn}\mu_k^{hd}\right) \quad (4.8)$$

and its eigenvectors $\delta_{k'}^{phnhd}$, $k' = 1, \ldots, nml$, by

$$\delta_{k'}^{phnhd} = \delta_i^p \otimes \delta_j^{hn} \otimes \delta_k^{hd} \quad (4.9)$$

for $i = 1, \ldots, n$, $j = 1, \ldots, m$, $k = 1, \ldots, l$, with $k' = (i-1)ml + (j-1)l + k$. Note that the eigenvectors $\delta_{k'}^{phnhd}$ are orthogonal. If we normalise the eigenvectors δ_i^p, δ_j^{hn} and δ_k^{hd} by choosing the constants κ_i^p, κ_j^{hn} and κ_k^{hd} as in (1.18), (1.26) and (1.10), respectively, then the eigenvectors $\delta_{k'}^{phnhd}$, $k' = 1, \ldots, nml$, are orthonormal. In the rest of the chapter, we assume that the eigenvectors $\delta_{k'}^{phnhd}$, $k' = 1, \ldots, nml$, are already normalised.

4.2 FFT solution of the three-dimensional QSC equations

Consider the BVP (4.1)-(4.5) and the respective discretised problem (4.6). Let Λ^{phnhd} be the diagonal matrix holding the eigenvalues of the matrix A^{phnhd} of (4.6) as defined in (4.8). Let also V^{phnhd} be the matrix whose columns are the normalised eigenvectors of A^{phnhd}, i.e.,

$$V_{qk'}^{phnhd} = \delta_{k'q}^{phnhd}, \text{ for } k' = 1, \ldots, nml, q = 1, \ldots, nml,$$

with $\delta_{k'q}^{phnhd}$ defined in (4.8). Note that $V^{phnhd} = V^p \otimes V^{hn} \otimes V^{hd}$, where V^p, V^{hn} and V^{hd} are the matrices defined in (2.3), (2.5) and (2.1), respectively.

The solution x to the linear system (4.6) can be written as

$$\begin{aligned} \mathrm{x} &= V^{phnhd}(\Lambda^{phnhd})^{-1}(V^{phnhd})^{-1}\mathrm{g} \\ &= (V^p \otimes V^{hn} \otimes V^{hd})(\Lambda^{phnhd})^{-1}(V^p \otimes V^{hn} \otimes V^{hd})^{-1}\mathrm{g} \\ &= (V^p \otimes V^{hn} \otimes V^{hd})(\Lambda^{phnhd})^{-1}((V^p)^{-1} \otimes (V^{hn})^{-1} \otimes (V^{hd})^{-1})\mathrm{g}. \end{aligned} \tag{4.10}$$

Using the properties of the matrices V^p , V^{hn} and V^{hd} as described in Section 2.1, x can be written as

$$\mathrm{x} = (\mathbb{F}_n^{-1} \otimes \mathrm{C}_m^{-1} \otimes \mathbb{S}_l^{-1})(\Lambda^{phnhd})^{-1}(\mathbb{F}_n \otimes \mathrm{C}_m \otimes \mathbb{S}_l)\mathrm{g}. \tag{4.11}$$

Taking into account property **Kron56** of Section 1.5, and applying it twice for each triple tensor product of (4.11), the computation involved in (4.11) can be performed in the following steps:

Step 1: Perform FFT to each of the ml columns of the vector $(\mathrm{g}_{ml\times n})^T$ to obtain $(\mathrm{g}^{(1)})_{n\times ml} = \mathbb{F}_n(\mathrm{g}_{ml\times n})^T$.

Step 2: Perform FCT-II to each of the nl columns of the vector $((\mathrm{g}^{(1)})_{nl\times m})^T$ to obtain $(\mathrm{g}^{(2)})_{m\times nl} = \mathrm{C}_m((\mathrm{g}^{(1)})_{nl\times m})^T$.

Step 3: Perform FST-II to each of the nm columns of the vector $((\mathrm{g}^{(2)})_{nm\times l})^T$ to obtain $(\mathrm{g}^{(3)})_{l\times nm} = \mathbb{S}_l((\mathrm{g}^{(2)})_{nm\times l})^T$, or, equivalently, $\mathrm{g}^{(3)} = (\mathbb{F}_n \otimes \mathrm{C}_m \otimes \mathbb{S}_l)\mathrm{g}$.

Step 4: Divide each of the components of $g^{(3)}$ by the respective eigenvalue of A^{phnhd} to obtain vector $g^{(4)} = (\Lambda^{phnhd})^{-1} g^{(3)}$.

Step 5: Perform iFST-II to each of the nm columns of the vector $((g^{(4)})_{nm \times l})^T$ to obtain $(g^{(5)})_{l \times nm} = \mathbb{S}_l^{-1}((g^{(4)})_{nm \times l})^T$.

Step 6: Perform iFCT-II to each of the nl columns of the vector $((g^{(5)})_{nl \times m})^T$ to obtain $(g^{(6)})_{m \times nl} = \mathrm{C}_m^{-1}((g^{(5)})_{nl \times m})^T$.

Step 7: Perform iFFT to each of the ml columns of the vector $((g^{(6)})_{ml \times n})^T$ to obtain $x_{n \times ml} = \mathbb{F}_n^{-1}((g^{(6)})_{ml \times n})^T$, or, equivalently:

$$x = (\mathbb{F}_n^{-1} \otimes \mathrm{C}_m^{-1} \otimes \mathbb{S}_l^{-1})(\Lambda^{phnhd})^{-1}(\mathbb{F}_n \otimes \mathrm{C}_m \otimes \mathbb{S}_l) g.$$

The above seven steps form an FFT technique for solving (4.6). Clearly, the seven steps involve $5nml \log_2 nml$ (+lower order terms of nml) flops.

4.3 The constant coefficient case

In this section, results from numerical experiments are presented that demonstrate the performance of the FFT solvers for the three-dimensional QSC equations.

For the implementation of the FFT, iFFT, FST-II, iFST-II, FCT-II and iFCT-II the same package [Mos92] was used as for the two-dimensional problems. The results shown in Table 4.1, correspond to Problem 6 of the Appendix. The error and convergence results are presented in order to verify that the performance of the QSC method on three-dimensional problems is similar to that for two-dimensional ones. For all computations we have used double precision. For each discretisation grid size shown in Table 4.1, the "global error" corresponds to the maximum absolute value of the error of the QSC approximation on a uniform grid of $20 \times 20 \times 20$ points, while the "error on grid points" corresponds to the maximum absolute value of the error of the QSC approximation on the discretisation grid. The "global error" is taken as an approximation of the uniform norm of the error. The results show that the QSC method is globally of third order, while on the grid points it is of fourth order, as expected.

The last column of Table 4.1 shows the time in seconds required for the FFT solution of the QSC equations corresponding to Problem 6 of the Appendix. The timing results were obtained on a Sparcstation20. If we use constant least-squares to approximate the timing data of Table 4.1 by a function of the form $\rho n^3 \log_2 n$ then $\rho = 2.61 \times 10^{-6}$. If we use linear least-squares to approximate the timing data of Table 4.1 by a function of the form $\rho_2 n^{\rho_1}$ then $\rho_1 = 3.1$ and $\rho_2 = 10^{-5}$. Therefore, the FFT solution of the three-dimensional QSC equations is asymptotically almost optimal.

	Error				Order of Convergence				Time
	First step		Second step		First step		Second step		
n	Global error	Error on grid	Global error	Error on grid	Global	On grid	Global	On grid	
8	$1.13E-02$	$1.25E-02$	$4.25E-03$	$3.36E-03$	2.94	2.89	3.68	4.15	0.007
16	$2.29E-03$	$2.53E-03$	$4.42E-04$	$2.07E-04$	2.30	2.31	3.27	4.02	0.047
32	$5.66E-04$	$4.61E-04$	$5.83E-05$	$1.00E-05$	2.01	2.46	2.92	4.36	0.351
64	$1.41E-04$	$4.85E-05$	$4.19E-06$	$2.70E-07$	2.01	3.25	3.80	5.22	3.350

Table 4.1: Errors and respective orders of convergence corresponding to Problem 6 discretised by the QSC method, for several grid sizes n and for both steps of the QSC method. Time in seconds for the FFT solution of the QSC equations corresponding to Problem 6 for several grid sizes n and for one step of the QSC method.

4.4 The variable coefficient case

In this section we consider the FFT solution of QSC equations arising from the discretisation of three-dimensional BVPs with variable coefficient differential operators.

The BVPs considered are defined by the PDE

$$\begin{aligned}\mathbf{L}u(x,y,z) \equiv a(x,y,z)u_{xx} + c(x,y,z)u_{yy} + p(x,y,z)u_{zz} + b(x,y,z)u_{xy}\\ + r(x,y,z)u_{xz} + s(x,y,z)u_{yz} + d(x,y,z)u_x + e(x,y,z)u_y\\ + q(x,y,z)u_z + f(x,y,z)u = g(x,y,z) \text{ in } \Omega \equiv (0,1)\times(0,1)\times(0,1)\end{aligned} \quad (4.12)$$

subject to the boundary conditions (4.2)-(4.5) on the boundary of the domain Ω. It is worth noting that the boundary conditions described by (4.2)-(4.5) are taken as an example of the types of boundary conditions considered. The FFT solvers described in this section are applicable to PDEs with any combination of periodic, homogeneous Dirichlet and homogeneous Neumann boundary conditions, as long as at each side of the boundary one type of these boundary conditions applies.

Consider a rectangular partition Δ of Ω with uniform step sizes $h_x = \dfrac{1}{n}$, $h_y = \dfrac{1}{m}$ and $h_z = \dfrac{1}{l}$ in the x, y and z directions respectively. Consider also the set of basis functions $\{\phi_i^x(x)\theta_j^y(y)\psi_k^z(z)\}_{i=1,j=1,k=1}^{n,m,l}$ as defined in Section 4.1, and recall that these basis functions satisfy the boundary conditions (4.2)-(4.5) by construction. The QSC method applied to the BVP using the above set of basis functions results in a sparse and almost block tridiagonal linear system

$$A\mathrm{x} = \mathrm{g} \quad (4.13)$$

of size $nml \times nml$. The eigenvalues of the matrix A are not known in analytic form, therefore the FFT solvers described in the previous section are not directly applicable to the linear system (4.13). We study preconditioned iterative methods for solving (4.13). The preconditioning techniques considered are similar to those described in Chapter 3 for the solution of the two-dimensional QSC equations.

A type of preconditioner considered is the matrix H arising from the discretisation of the Helmholtz operator

$$u_{xx} + u_{yy} + u_{zz} - u \tag{4.14}$$

subject to the same boundary conditions as the original BVP. We refer to this type of preconditioner as *Helmholtz (H) preconditioner*.

We also consider the following three forms of preconditioners P:

a) $P = \sqrt{D}H\sqrt{D}$,

b) $P = DH$,

c) $P = HD$,

where D is the diagonal matrix with entries the diagonal entries of A. We refer to the first type of preconditioner as the *DHD preconditioner*, to the second type of preconditioner as the *DH preconditioner*, and to the third type of preconditioner as the *HD preconditioner*. These preconditioners are tested with the iterative methods discussed in Section 3.2.

We present numerical results that demonstrate the convergence and overall performance of the Fourier preconditioned iterative solvers on the three-dimensional QSC equations. In all three-dimensional experiments, the stopping criterion, tolerance and initial guess vectors were chosen in a way similar to that for the two-dimensional experiments of Section 3.3. For the implementation of the acceleration methods, e.g., GMRES and BiCGStab, the KSP package [GrSm93] was used. For the rest of the methods we used our own code in FORTRAN. All results were obtained on a Sparcstation20 in double precision.

Problem 7 of the Appendix is a variable coefficient problem with the coefficients of u_{xx}, u_{yy} and u_{zz} being equal and ranging between 0.7 and 1.3 and the coefficients of u_x, u_y and u_z ranging between -0.3 and 0.3. Table 4.2 shows the number of iterations and the respective time required for the convergence of the GMRES, BiCGStab, MRES and RICH methods, with various preconditioners applied to the QSC equations arising from the discretisation of Problem 7.

The results of Table 4.2 lead us to conclusions similar to those made for the FFT solvers for the two-dimensional QSC equations.

- The rate of convergence of the preconditioned iterative methods is independent of the problem size n.

- BiCGStab requires fewer iterations (about half) than GMRES to converge, but takes about the same time, since it requires two multiplications per iteration, while GMRES requires one.

- MRES requires a few more iterations than GMRES to converge, and is overall slower than GMRES.

- The RICH method requires more iterations than MRES to converge and is the slowest method for this problem.

- The DH preconditioner outperforms the rest (except with RICH).

method	GMRES															
precond.	H		DHD		DH		HD		H		DHD		DH		HD	
	no. of iter.								time							
	step		step		step		step		step 1		step 1		step 1		step 1	
n	1	2	1	2	1	2	1	2	total	per iter.	total	per iter.	total	per iter.	total	per iter.
4	6	5	6	5	6	5	6	5	0.031	0.0051	0.028	0.0046	0.029	0.0048	0.027	0.0045
8	9	7	7	5	7	5	7	5	0.282	0.0313	0.221	0.0316	0.224	0.0320	0.218	0.0312
16	9	7	7	5	7	5	8	6	2.251	0.2501	1.800	0.2572	1.792	0.2560	2.016	0.2520
33	9	7	8	6	7	5	8	6	18.138	2.0153	16.926	2.1157	14.681	2.0973	16.479	2.0598
64	10	7	8	6	7	5	8	6	170.727	17.0727	142.996	17.8744	121.534	17.3620	139.950	17.4937

method	BiCGStab															
precond.	H		DHD		DH		HD		H		DHD		DH		HD	
	no. of iter.								time							
	step		step		step		step		step 1		step 1		step 1		step 1	
n	1	2	1	2	1	2	1	2	total	per iter.	total	per iter.	total	per iter.	total	per iter.
4	4	3	4	3	4	3	4	3	0.037	0.0092	0.037	0.0093	0.033	0.0084	0.034	0.0085
8	5	4	4	3	4	3	4	3	0.292	0.0584	0.249	0.0622	0.250	0.0626	0.237	0.0592
16	5	4	4	3	4	3	4	3	2.363	0.4727	2.016	0.5040	1.966	0.4916	1.952	0.4880
32	5	3	4	3	4	3	4	3	19.543	3.9087	16.474	4.1186	16.629	4.1572	15.861	3.9653
64	5	3	5	4	4	3	4	3	164.638	32.9276	170.408	34.0816	135.371	33.8428	134.817	33.7042

Table 4.2: Number of iterations and time in seconds required for the convergence of various preconditioned iterative techniques for the solution of the QSC equations arising from the discretisation of Problem 7 for several grid sizes n, and for both steps of the QSC method.

method	MRES															
precond.	H		DHD		DH		HD		H		DHD		DH		HD	
	no. of iter.								time							
	step		step		step		step		step 1		step 1		step 1		step 1	
n	1	2	1	2	1	2	1	2	total	per iter.	total	per iter.	total	per iter.	total	per iter.
4	11	8	8	6	8	6	8	6	0.050	0.0045	0.037	0.0047	0.036	0.0044	0.034	0.0043
8	12	9	9	6	9	6	9	7	0.408	0.0340	0.309	0.0344	0.299	0.0332	0.292	0.0324
16	12	8	10	7	9	6	10	8	3.284	0.2737	2.812	0.2812	2.407	0.2675	2.625	0.2625
32	12	8	12	8	9	6	11	8	25.935	2.1612	26.717	2.2264	19.865	2.2072	23.844	2.1677
64	12	8	14	10	9	7	12	10	195.480	16.2900	239.198	17.0856	161.796	17.9774	208.523	17.3769

Table 4.2: Continued.

method	RICH			
precond.	H			
	no. of iter.		time	
	step		step 1	
n	1	2	total	per iter.
4	15	11	0.053	0.0035
8	14	10	0.374	0.0267
16	14	10	3.047	0.2177
32	14	10	24.816	1.7726
64	14	10	208.684	14.9060

Table 4.2: Continued.

Chapter 5

Conclusions

This thesis deals with the FFT solution of QSC equations arising from the discretisation of linear elliptic BVPs in one, two and three dimensions. The boundary conditions considered are homogeneous Dirichlet, periodic, homogeneous Neumann and any combination of these.

Based on the analytic forms of the eigenvalues and eigenvectors of the QSC matrix as described in [Chr94], we develop FFT solvers for the QSC equations arising from the discretisation of PDEs with constant coefficients and even order derivatives. The performance of the FFT solvers on the QSC equations is almost optimal. More specifically the computation time is proportional to the number of equations multiplied by the logarithm of the number of equations.

Diagonal scalings of these solvers are then used as preconditioners for the solution of the QSC equations arising from general elliptic PDEs. It is experimentally shown that the convergence rate of various one-step methods and acceleration techniques integrated with the above preconditioners and applied to the QSC equations is independent of the problem size. Thus, the performance of the FFT solvers for QSC equations arising from general elliptic PDEs is almost optimal. The FFT solvers when compared with other iterative solvers, e.g. multigrid solvers, are shown to be effective alternatives, with similar asymptotic behaviour.

A question that arises is how effective the FFT solvers would be in a parallel environment. It is well known that the FFT computations are appropriate for parallel computation. A difficulty arises, though, in the case of two- and three-dimensional problems. The transposition of matrices required, for example, between Steps 1 and 2 of Section 2.2 is not straighforward parallelisable and the technique used with the FFT package [Mos92] is not applicable in a distributed memory environment. Therefore, an interesting future problem is to test the performance of the FFT solvers on various parallel machines and to develop methods to overcome the transposition problem on specific interconnection networks among processors.

Appendix A

Problems

Problem 1.

Determine $u = u(x, y)$ such that

$$2u_{xx} + 3u_{yy} - u = g \text{ in } (0, 2\pi) \times (0, 1)$$

subject to

$$\begin{cases} u(x,0) = 0 & \text{for} \quad 0 \le x \le 2\pi \\ u(x,1) = 0 & \text{for} \quad 0 \le x \le 2\pi \\ u(0,y) = u(2\pi, y) & \text{for} \quad 0 \le y \le 1 \\ u_x(0,y) = u_x(2\pi, y) & \text{for} \quad 0 \le y \le 1 \end{cases}$$

The function $g = g(x, y)$ is chosen so that the solution to the problem is $u(x, y) = \cos(x)(y - y^2)$.

Problem 2.

Determine $u = u(x, y)$ such that

$$a(x,y)u_{xx} + c(x,y)u_{yy} + f(x,y)u = g \text{ in } (0, 2\pi) \times (0, \pi)$$

where $a(x, y), c(x, y), f(x, y)$ are given by

$$\begin{cases} a(x,y) &=& 1.0/(2.0 + \sin(x)\cos(y)) \\ c(x,y) &=& 1.0/(2.0 - \sin(x)\cos(y)) \\ f(x,y) &=& -1.0 \end{cases}$$

subject to

$$\begin{cases} u(x,0) = 0 & \text{for} \quad 0 \le x \le 2\pi \\ u(x,\pi) = 0 & \text{for} \quad 0 \le x \le 2\pi \\ u(0,y) = u(2\pi,y) & \text{for} \quad 0 \le y \le \pi \\ u_x(0,y) = u_x(2\pi,y) & \text{for} \quad 0 \le y \le \pi \end{cases}$$

The function $g = g(x,y)$ is chosen so that the solution to the problem is $u(x,y) = \cos(x)\sin(y)$.

Problem 3.

Determine $u = u(x,y)$ such that

$$a(x,y)u_{xx} + c(x,y)u_{yy} + f(x,y)u = g \text{ in } (0,2\pi) \times (0,\pi)$$

where $a(x,y), c(x,y), f(x,y)$ are given by

$$\begin{cases} a(x,y) &=& 1.0 + \delta xy \\ c(x,y) &=& 1.0 + \delta xy \\ f(x,y) &=& -1.0 \end{cases}$$

subject to

$$\begin{cases} u(x,0) = 0 & \text{for} \quad 0 \le x \le 2\pi \\ u(x,\pi) = 0 & \text{for} \quad 0 \le x \le 2\pi \\ u(0,y) = u(2\pi,y) & \text{for} \quad 0 \le y \le \pi \\ u_x(0,y) = u_x(2\pi,y) & \text{for} \quad 0 \le y \le \pi \end{cases}$$

The function $g = g(x,y)$ is chosen so that the solution to the problem is $u(x,y) = \sin(x)\sin(y)$.

Problem 4.

Determine $u = u(x,y)$ such that

$$a(x,y)u_{xx} + c(x,y)u_{yy} + d(x,y)u_x + e(x,y)u_y + f(x,y)u = g \text{ in } (0,1) \times (0,1)$$

where $a(x,y), c(x,y), f(x,y)$ are given by

$$\begin{cases} a(x,y) & = & \exp(xy) \\ c(x,y) & = & 1.0 + \sin(xy) \\ d(x,y) & = & y\exp(xy) \\ e(x,y) & = & x\exp(xy) \\ f(x,y) & = & 0.0 \end{cases}$$

subject to

$$\begin{cases} u(x,0) = 0 & \text{for} \quad 0 \le x \le 1 \\ u(x,1) = 0 & \text{for} \quad 0 \le x \le 1 \\ u(0,y) = 0 & \text{for} \quad 0 \le y \le 1 \\ u(1,y) = 0 & \text{for} \quad 0 \le y \le 1 \end{cases}$$

The function $g = g(x,y)$ is chosen so that the solution to the problem is $u(x,y) = \exp(x+y)(x^2-x)(y^2-y)$.

Problem 5.

Determine $u = u(x,y)$ such that

$$a(x,y)u_{xx} + b(x,y)u_{xy} + c(x,y)u_{yy} + d(x,y)u_x + e(x,y)u_y + f(x,y)u = g$$

in $(0,2\pi) \times (0,\pi)$ where $a(x,y), b(x,y), c(x,y), f(x,y)$ are given by

$$\begin{cases} a(x,y) & = & 1.0 + 0.5\sin(x)\cos(y) \\ b(x,y) & = & 0.2\cos(x+y) \\ c(x,y) & = & 1.0 + \sin(xy) \\ d(x,y) & = & 0.5\cos(x)\cos(y) \\ e(x,y) & = & -0.5\sin(x)\sin(y) \\ f(x,y) & = & 0.0 \end{cases}$$

subject to

$$\begin{cases} u(x,0) = 0 & \text{for} \quad 0 \le x \le 2\pi \\ u(x,\pi) = 0 & \text{for} \quad 0 \le x \le 2\pi \\ u(0,y) = u(2\pi,y) & \text{for} \quad 0 \le y \le \pi \\ u_x(0,y) = u_x(2\pi,y) & \text{for} \quad 0 \le y \le \pi \end{cases}$$

The function $g = g(x,y)$ is chosen so that the solution to the problem is $u(x,y) = \sin(x)\sin(y)$.

Problem 6.

Determine $u = u(x,y,z)$ such that

$$au_{xx} + cu_{yy} + pu_{zz} + fu = g$$
$$\text{in } (0,2\pi)\times(0,\pi)\times(0,\pi) \text{ where } a=5, c=3, p=2, f=-1$$

subject to

$$\begin{cases} u(0,y,z) = u(2\pi,y,z) & \text{for } 0 \le y \le \pi \text{ and } 0 \le z \le \pi, \\ u_x(0,y,z) = u_x(2\pi,y,z) & \text{for } 0 \le y \le \pi \text{ and } 0 \le z \le \pi, \\ u_y(x,0,z) = 0 & \text{for } 0 \le x \le 2\pi \text{ and } 0 \le z \le \pi, \\ u_y(x,\pi,z) = 0 & \text{for } 0 \le x \le 2\pi \text{ and } 0 \le z \le \pi, \\ u(x,y,0) = 0 & \text{for } 0 \le x \le 2\pi \text{ and } 0 \le y \le \pi, \\ u(x,y,\pi) = 0 & \text{for } 0 \le x \le 2\pi \text{ and } 0 \le y \le \pi. \end{cases}$$

The function $g = g(x,y,z)$ is chosen so that the solution to the problem is $u(x,y,z) = \sin(x)\cos(y)\sin(z)$.

Problem 7.

Determine $u = u(x,y,z)$ such that

$$a(x,y,z)u_{xx} + c(x,y,z)u_{yy} + p(x,y,z)u_{zz} +$$
$$+\, d(x,y,z)u_x + e(x,y,z)u_y + q(x,y,z)u_z = g$$
$$\text{in } (0,2\pi)\times(0,\pi)\times(0,\pi)$$

where $a(x,y,z), c(x,y,z), d(x,y,z), e(x,y,z), p(x,y,z), q(x,y,z)$ are given by

$$\begin{cases} a(x,y,z) &=& 1.0 + 0.3\cos(x)\cos(y)\sin(z) \\ c(x,y,z) &=& 1.0 + 0.3\cos(x)\cos(y)\sin(z) \\ d(x,y,z) &=& -0.3\sin(x)\cos(y)\sin(z) \\ e(x,y,z) &=& -0.3\cos(x)\sin(y)\sin(z) \\ p(x,y,z) &=& 1.0 + 0.3\cos(x)\cos(y)\sin(z) \\ q(x,y,z) &=& 0.3\cos(x)\cos(y)\cos(z) \end{cases}$$

subject to

$$\begin{cases} u(0,y,z) = u(2\pi,y,z) & \text{for } 0 \le y \le \pi \text{ and } 0 \le z \le \pi, \\ u_x(0,y,z) = u_x(2\pi,y,z) & \text{for } 0 \le y \le \pi \text{ and } 0 \le z \le \pi, \\ u_y(x,0,z) = 0 & \text{for } 0 \le x \le 2\pi \text{ and } 0 \le z \le \pi, \\ u_y(x,\pi,z) = 0 & \text{for } 0 \le x \le 2\pi \text{ and } 0 \le z \le \pi, \\ u(x,y,0) = 0 & \text{for } 0 \le x \le 2\pi \text{ and } 0 \le y \le \pi, \\ u(x,y,\pi) = 0 & \text{for } 0 \le x \le 2\pi \text{ and } 0 \le y \le \pi. \end{cases}$$

The function $g = g(x,y,z)$ is chosen so that the solution to the problem is $u(x,y,z) = \sin(x)\cos(y)\sin(z)$.

Bibliography

[Arc73] D. A. Archer. *Some Collocation Methods for Differential Equations.* PhD thesis, Rice University, Houston, TX, U.S.A., 1973.

[Ban77] R. E. Bank. Marching algorithms for elliptic boundary value problems. II: The variable coefficient case. *SIAM J. Numer. Anal.*, 14:950–970, 1977.

[BFB92] B. Bialecki, G. Fairweather, and K. Bennett. Fast direct solvers for piecewise Hermite bicubic orthogonal spline collocation equations. *SIAM J. Numer. Anal.*, 29:156–173, 1992.

[Bia94] B. Bialecki. Preconditioned Richardson and minimal residual iterative methods for piecewise Hermite bicubic orthogonal spline collocations equations. *SIAM J. Sci. Comput.*, 13:668–680, 1994.

[CoGo73] P. Concus and G. H. Golub. Use of fast direct methods for the efficient numerical solution of nonseperable elliptic equations. *SIAM J. Numer. Anal.*, 10:1103–1120, 1973.

[Chr88] C. C. Christara. *Spline Collocation Methods, Software and Architectures for Linear Elliptic Partial Differential Equations.* PhD thesis, Purdue University, IN, U.S.A., 1988.

[Chr94] C. C. Christara. Quadratic spline collocation methods for elliptic partial differential equations. *BIT*, 34:1:33–61, 1994.

[Chr96] C. C. Christara. Parallel solvers for spline collocation equations, to appear in *Computing Systems in Engineering*, 1996.

[ChSm95] C. C. Christara and B. Smith. Multilevel methods for quadratic spline collocation, submitted to *BIT*, 1995.

[DaSw75] J. W. Daniel and B. K. Swartz. Extrapolated collocation for two point boundary value problems using cubic splines. *J. Inst. Math. Appl.*, 16:161–174, 1975.

[HVR88] E. N. Houstis, E. A. Vavalis and J. R. Rice. Convergence of an $O(h^4)$ cubic spline collocation method for elliptic partial differential equations. *SIAM J. Numer. Anal.*, 25:54–74, 1988.

[HRCV88] E. N. Houstis, J. R. Rice, C. C. Christara and E. A. Vavalis. Performance of scientific software. In J. R. Rice, editor, *Mathematical Aspects of Scientific Software*, pages 123–156. Springer Verlag, 1988.

[ElSc86] H. C. Elman and M. H. Schultz. Preconditioning by fast direct methods for nonselfadjoint nonseperable elliptic equations. *SIAM J. Numer. Anal.*, 23:44–57, 1986.

[Fyf68] D. J. Fyfe. The use of cubic splines in the solution of two-point boundary value problems. *Comp. J.*, 17:188–192, 1968.

[Gre92] A. Greenbaum. Diagonal scalings of the Laplacian as preconditioners for other elliptic differential operators. *SIAM J. Matrix Anal. Appl.*, 13:826–846, 1992.

[GrSm93] W. Gropp and B. Smith. *Users Manual for KSP: Data-Structure-Neutral Codes Implementing Krylov Space Methods.* Argonne National Laboratory, August 1993.

[HCR88] E. N. Houstis, C. C. Christara and J. R. Rice. Quadratic-spline collocation methods for two-point boundary value problems. *International Journal for Numerical Methods in Engineering*, 26:935–952, 1988.

[Iro87] M. Irodotou-Ellina. *Optimal Spline Collocation Methods for High Degree Two Point Boundary Value Problems.* PhD thesis, Aristotle University of Thessaloniki, Greece, 1987.

[Loa92] C. Van Loan. *Computational Frameworks for the Fast Fourier Transform.* Society for Industrial and Applied Mathematics, 1992.

[Mos92] B. Mossberg. Documentation for the FFT subroutines available in the four libraries. NEC Systems Laboratory, Inc., 1992.

[PaSo87] N. Papamichael and M. J. Soares. A class of cubic and quintic spline modified collocation methods for the solution of two-point boundary value problems. Technical report, Brunel University, 1987.

[EES83] S. C. Eisenstat, H. C. Elman and M. H. Schultz. Variational iterative methods for nonsymmetric systems of linear equation. *SIAM J. Numer. Anal.*, 20:345–357, 1983.

[SaSc86] Y. Saad and M. H. Schultz. GMRES: A generalised minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7:856–869, 1986.

[vdV92] H. van der Vorst. Bi-CGStab: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 13:631–644, 1992.