Fast Algorithms for Boundary Integral Equations on Elliptic Domains and Related Inverse Problems

Weifu Fang\textsuperscript{2}, Fu-Rong Lin\textsuperscript{1,*} and Yan-Bo Ma\textsuperscript{1,3}

\textsuperscript{1}Department of Mathematics, Shantou University, Shantou Guangdong 515063, China.
\textsuperscript{2}Department of Mathematics and Statistics, Wright State University, Dayton, OH 45435, USA.
\textsuperscript{3}Department of Scientific Computing, Hanshan Normal University, Chaozhou, Guangdong 521041, China.

Received 13 September 2017; Accepted (in revised version) 17 August 2018.

Abstract. Fast algorithms for boundary integral equations connected with Robin boundary value problem for the Laplace equation in domains with ellipse or close to ellipse boundaries are developed. It is shown that the coefficient matrices of discretisation systems have a special structure. This fact is used to develop a fast algorithm for matrix vector multiplication and to implement it in the numerical methods used. Such an approach is especially helpful in numerical methods for inverse problems, since many methods of their solution repeatedly use forward solvers. The efficiency of the methods is illustrated by numerical examples.

AMS subject classifications: 65F22, 65R32

Key words: Robin boundary value problem, Robin inverse problem, circulant-like matrix, fast iterative methods.

1. Introduction

We consider a Robin boundary value problem for the Laplace equation in a smooth connected open domain $\Omega \subset \mathbb{R}^2$ with the boundary $\Gamma$ — viz.

\begin{equation}
\begin{aligned}
\Delta u &= 0 \quad \text{in} \quad \Omega, \\
\frac{\partial u}{\partial \nu} + pu &= g \quad \text{on} \quad \partial \Omega = \Gamma,
\end{aligned}
\end{equation}

(1.1)

where $\nu$ is the outward unit normal direction of the boundary and $p$ the Robin coefficient, nonnegative and not identically equal to zero on $\Gamma$.

\textsuperscript{*}Corresponding author. Email address: frlin@stu.edu.cn (F.-R. Lin)
The Robin boundary value problem (1.1) serves as a mathematical model for various physical phenomena and is widely used in applications, including heat distribution of thermal conducting materials and electrical potentials in semiconductors with contacts through the boundary. For known $\Gamma$, $p$ and $g$, the solution $u$ of (1.1) is uniquely determined. More interesting problems often arising in applications, consist in recovering the term $p$ or part of $\Gamma$ in (1.1) from additional data — e.g. from additional measurements of the solution $u$ on an accessible part of the boundary $\Gamma$. This extra information on the solution is to be used to extract various physical parameters of interest, such as the Robin coefficient $p$ that describes the quality of metal-to-silicon contact on the boundary in a transistor, or an inaccessible part of $\Gamma$ that represents a desirable material profile. Such inverse problems have been extensively studied in recent years both by analytical and numerical methods — cf. [1,3–6,8,13,15] and references therein.

One of popular approaches to the problem (1.1) consists in transforming the boundary value problem for the Laplace equation into an equivalent boundary integral equation (BIE) for $u$ on $\Gamma$ and applying suitable numerical methods to the resulting integral equation. This approach has been successfully used in solving inverse problems for (1.1) and we follow it here. Transforming (1.1) into a boundary integral equation on an ellipse and discretising it, we exploit the structure of the coefficient matrix of the resulting discrete system and construct a fast algorithm for the matrix-vector multiplication, which requires only $O(n \log n)$ operations instead of standard $O(n^2)$. This is also an essential ingredient in iterative methods for the forward problem. We adopt the idea to construct numerical methods, which work faster than conventional algorithms for boundaries $\Gamma$ close to an ellipse specified in Section 2.3. The solution of inverse problems often depends on repeated use of forward solvers, so that a faster forward solver can essentially speed up the corresponding numerical methods. We illustrate the efficiency of these fast algorithms by solving two inverse problems — viz. the recovery of the Robin coefficient $p$ on an ellipse and the recovery of a part of non-elliptic segment on $\Gamma$. In both cases, measurements of $u$ on another part of $\Gamma$ are used. Numerical examples show a reduced computation time in comparison with conventional algorithms that do not exploit special structure of arising matrices. This property is especially valuable if the size of discrete system grows.

The outline of the paper is as follows. In Section 2, the Robin problem (1.1) is reduced to an equivalent boundary integral equation. Assuming that $\Gamma$ is an ellipse, we then construct a fast algorithm for the discretisation of the linear system. This algorithm is also adopted in the case where a small part of $\Gamma$ differs from ellipse arc. In Section 3, we apply fast algorithms to two inverse problems: the recovery of the Robin coefficient and the recovery of a part of the boundary from the measurements of $u$ on another part of $\Gamma$. Numerical results presented in Section 4, illustrate the efficiency of the methods. Our concluding remarks are in Section 5.

2. Boundary Integral Equations and Fast Methods

Assume that the domain $\Omega$ has a $C^2$-boundary $\Gamma$, and let $p(x) \geq 0$ on $\Gamma$ with $\text{supp}(p) \neq \emptyset$. By Green’s formula, the solution $u$ of (1.1) in $\Omega$ can be represented via its boundary values
as
\[
  u(x) = -\int_{\Gamma} \left( \frac{\partial \Phi(x, y)}{\partial y} + \Phi(x, y) p(y) \right) u(y) ds_y + \int_{\Gamma} \Phi(x, y) g(y) ds_y, \quad x \in \Omega,
\]
where
\[
  \Phi(x, y) = \frac{1}{2\pi} \ln \frac{1}{|x - y|}, \quad x, y \in \Omega, \quad x \neq y
\]
is the fundamental solution of the Laplace equation \( \Delta u = 0 \) and \( |x - y| \) the distance between \( x \) and \( y \). Letting \( x \in \Omega \) tend to \( \Gamma \), one obtains that on \( \Gamma \) the function \( u \) satisfies the integral equation
\[
  \frac{1}{2} u(x) + \int_{\Gamma} \left( \frac{\partial \Phi(x, y)}{\partial y} + \Phi(x, y) p(y) \right) u(y) ds_y = \int_{\Gamma} \Phi(x, y) g(y) ds_y, \quad x \in \Gamma, \quad (2.1)
\]
where \( ds_y \) denotes the differential of the arc length [14]. Thus the boundary-value problem (1.1) is reduced to the integral equation (2.1).

Let \( \mathcal{D} \) and \( \mathcal{S} \), respectively, denote the double-layer and single-layer potential operators — i.e.
\[
  (\mathcal{D}u)(x) := \int_{\Gamma} \frac{\partial \Phi(x, y)}{\partial y} u(y) ds_y, \quad (\mathcal{S}u)(x) := \int_{\Gamma} \Phi(x, y) u(y) ds_y, \quad x \in \Gamma.
\]

Then the Eq. (2.1) can be written in the operator form as
\[
  \mathcal{A}(u) = f, \quad (2.2)
\]
where
\[
  \mathcal{A}(u) = \left( \frac{1}{2} \mathcal{D} + \mathcal{S} \right) u + \mathcal{S}(pu), \quad f = \mathcal{S} g,
\]
and \( \mathcal{I} \) refers to the identity operator.

Let
\[
  x = (\phi(t), \psi(t)), \quad 0 \leq t \leq 1
\]
be a parametrisation of \( \Gamma \) and \( \phi, \psi \in C^2[0, 1] \), where \( (\phi(0), \psi(0)) = (\phi(1), \psi(1)) \). Then the operators \( \mathcal{S} \) and \( \mathcal{D} \) can be rewritten as integral operators
\[
  (\mathcal{D}u)(t) = \int_0^1 K_1(t, s) u(s) ds, \quad (\mathcal{S}u)(t) = \int_0^1 K_2(t, s) u(s) ds
\]
with the kernels
\[
  K_1(t, s) = \frac{1}{2\pi} \frac{\psi'(s) (\phi(t) - \phi(s)) - \phi'(s) (\psi(t) - \psi(s))}{(\phi(t) - \phi(s))^2 + (\psi(t) - \psi(s))^2}, \quad (2.3)
\]
\[
  K_2(t, s) = (K_1(t, s) + K_2(t, s)) K_3(s), \quad (2.4)
\]
and

\[
K_{s1}(t, s) = \ln(2|\sin(\pi(t-s))|),
\]

\[
K_{s2}(t, s) = \begin{cases} 
\ln\left(\frac{\sqrt{(\phi(t) - \phi(s))^2 + (\psi(t) - \psi(s))^2}}{2|\sin(\pi(t-s))|}\right), & s \neq t, \\
\ln\left(\frac{\sqrt{\phi'(t)^2 + \psi'(t)^2}}{2\pi}\right), & s = t,
\end{cases}
\]

\[
K_{s3}(s) = -\frac{1}{2\pi}\sqrt{\phi'(s)^2 + (\psi'(s))^2}.
\]

Consequently, the operator \(\mathcal{A}\) in (2.2) takes the form

\[
(\mathcal{A}u)(t) = \frac{1}{2}u(t) + \int_0^1 (K_d(t, s) + K_s(t, s)p(s)) u(s)ds \quad \text{for} \quad t \in [0, 1].
\]

2.1. Discretisation

In order to discretise the integral equation (2.2), we apply the Nyström method based on simplest quadrature rules for integral operators with continuous kernels and trigonometric polynomial interpolation for ones with logarithm kernel — cf. [14, Section 12.3].

Let \((x_1, x_2) = (\phi(t), \psi(t)), \ t \in [0, 1]\) be a parametrisation of \(\Gamma\). Given an even integer \(n\), we subdivide the interval \([0, 1]\) into \(n\) equal subintervals \([i-1]h, ih]\), \(h = 1/n\), and choose quadrature points as \(t_i = (i-1)h\ (i = 1, 2, \cdots, n)\). Since the kernel function \(K_{s1}\) has a log-singularity along the diagonal \(t = s\), we apply the trigonometric polynomial interpolation [14, Section 12.3], thus obtaining

\[
S^{(1)} = \left[\begin{smallmatrix} s_{i-1}^{(1)} \\ \vdots \\ s_{i-1}^{(n)} \end{smallmatrix}\right]_{i=1}^n,
\]

where

\[
s_j^{(1)} = -\frac{1}{n} \sum_{l=1}^{n/2-1} \frac{1}{l} \cos \frac{2lj\pi}{n} + \frac{(-1)^j}{n}, \quad j = 0, \pm1, \cdots, \pm(n-1).
\]

Since kernel functions \(K_d(t, s), K_{s2}(t, s)\) are continuous, we use the rectangular rule, so that their discrete approximations are

\[
D = h \left[ [K_d(t_i, t_j)]_{i,j=1}^n \right], \quad S^{(2)} = h \left[ [K_{s2}(t_i, t_j)]_{i,j=1}^n \right].
\]

The corresponding discrete operators for the operator of multiplication by the function \(K_{s3}(s)\) have the form

\[
S^{(3)} = \text{diag}\left(s^{(3)}\right), \quad s^{(3)} = [K_{s3}(t_j)]_{j=1}^n,
\]

where \([v_j]_{j=1}^n = [v_1, \cdots, v_n]^T\) and diag denotes the diagonal matrix. Thus the kernel \(K_s(t, s)\) has the discrete representation

\[
S = (S^{(1)} + S^{(2)}) S^{(3)} = \left[ s_{i-j}^{(1)} + hK_{s2}(t_i, t_j) \right]_{i,j=1}^n \text{diag}\left( [K_{s3}(t_j)]_{j=1}^n \right).
\]
Let $u$, $p$ and $g$ be, respectively, the corresponding discrete representations of $u$, $p$ and $g$ — i.e.

$$u = [u(\phi(t_j), \psi(t_j))]_{j=1}^n, \quad p = [p(\phi(t_j), \psi(t_j))]_{j=1}^n, \quad g = [g(\phi(t_j), \psi(t_j))]_{j=1}^n,$$

and let $f = Sg$. Then

$$A = \frac{1}{2}I + D + \left(S^{(1)} + S^{(2)}\right) \text{diag}(s^{(3)} \circ p)$$

(2.10)

is the discrete version of the operator $\mathcal{A}$ in (2.2). As usual, $I$ refers to the corresponding identity matrix and $C \circ D := [c_{ij}d_{ij}]_{i=1;m}^{j=1;n}$ is the Hadamard product of the matrices $C = [c_{ij}]_{i=1;m}^{j=1:n}$ and $D = [d_{ij}]_{i=1;m}^{j=1:n}$. We recall that $C \circ D$ is the matrix formed by multiplication of the corresponding entries of $C$ and $D$. It is also called the Schur product of $C$ and $D$.

Thus discretising (2.2) leads to the system of linear equations

$$Au = f.$$

(2.11)

It is well-known that the Nyström method for BIEs depends on the choice of quadrature rules, so that there are various approaches to construct fast algorithms — [2, 9, 10, 12]. Our goal is to exploit the special structure of the matrix $A$ for elliptic domains in the construction of a fast algorithm for (2.11).

### 2.2. Elliptic domains

We recall that an $n \times n$ matrix $T$ of the form $T = [t_{i-j}]_{i,j=1}^n$ is called a Toeplitz matrix and a circulant if $t_j = t_{j-n}$ for $j = 1, 2, \ldots, n-1$. An $n \times n$ matrix $H$ of the form $H = [h_{i+j}]_{i,j=1}^n$ is called a Hankel matrix. It is easily seen that if $J$ is the exchange matrix (or reversal identity matrix) and $H$ is a Hankel one, then matrix, then $HJ = [h_{i+n-j+1}]_{i,j=1}^n$ is a Toeplitz matrix.

All circulant matrices can be diagonalised by a Fourier matrix [17, Section 5.2.3] — i.e. if $C = [c_{i-j}]_{i,j=1}^n$ and $c_j = c_{j-n}$, $j = 1, 2, \ldots, n-1$, then

$$C = F^* \cdot \text{diag} \lambda_C \cdot F,$$

where $F$ is the Fourier matrix of order $n$,

$$F = \frac{1}{\sqrt{n}} \left[ \exp \left( \frac{-2\pi \sqrt{-1}(i-1)(j-1)}{n} \right) \right]_{i,j=1}^n,$$

and $\lambda_C = [\lambda_i]_{i=1}^n$ such that

$$\lambda_i = \sum_{j=1}^{n-1} c_{j-i} \exp \left( \frac{-2\pi \sqrt{-1}(i-1)(j-1)}{n} \right).$$

Note that $\lambda_C$ can be obtained in $\Theta(n \log n)$ operations by applying the fast Fourier transform (FFT) to the first column of $C$. 


For circulant matrix $C$ and any vector $x \in \mathbb{C}^n$, the product $Cx$ can be calculated by

$$Cx = F^t(\lambda_C \circ (Fx)).$$

Thus one FFT and one IFFT (inverse FFT) of a vector of dimension $n$ have to be used to compute $Cx$. Let $\rho(n)$ refer to the cost of each of such an operation. It is known $\rho(n) = \mathcal{O}(n \log n)$ — cf. [17, §5.2.2].

We now assume that $\Omega$ in (1.1) is an elliptic domain — i.e.

$$\Omega = \left\{(x_1, x_2): \frac{x_1^2}{a^2} + \frac{x_2^2}{b^2} \leq 1\right\},$$

where $a, b > 0$ and consider the standard parametrisation

$$x = x(t) = (\phi(t), \psi(t)) = (a \cos(2\pi t), b \sin(2\pi t)), \quad 0 \leq t \leq 1$$

of the boundary $\Gamma$. It follows from (2.3)-(2.7) that the kernels of the corresponding integral operators have the form

$$K^{(\text{ell})}_d(t, s) = -\frac{ab}{2(a^2 \sin^2(\pi t + s) + b^2 \cos^2(\pi t + s))},$$

$$K^{(\text{ell})}_s(t, s) = \ln \sqrt{a^2 \sin^2(\pi t + s) + b^2 \cos^2(\pi t + s)},$$

$$K^{(\text{ell})}_{s_3}(s) = -\sqrt{a^2 \sin^2(2\pi s) + b^2 \cos^2(2\pi s)}.$$

Therefore, the matrices $D$, $S^{(2)}$ and $s^{(3)}$ have the entries:

$$[D]_{ij} = hK^{(\text{ell})}_d(t_i, t_j) = d_{i+j},$$

$$d_j = -\frac{ab}{2(a^2 \sin^2((j-2)\pi h) + b^2 \cos^2((j-2)\pi h))}, \quad j = 2, 3, \ldots, 2n,$$

$$[S^{(2)}]_{ij} = hK^{(\text{ell})}_s(t_i, t_j) = s^{(2)}_{i+j},$$

$$s^{(2)}_j = -h \ln \sqrt{a^2 \sin^2((j-2)\pi h) + b^2 \cos^2((j-2)\pi h)}, \quad j = 2, 3, \ldots, 2n,$$

$$[s^{(3)}]_{ij} = K^{(\text{ell})}_{s_3}(t_j) = s^{(3)}_j,$$

$$s^{(3)}_j = -\sqrt{a^2 \sin^2(2j\pi h) + b^2 \cos^2(2j\pi h)}, \quad j = 1, 2, \ldots, n.$$

The structure of the matrices $D$, $S^{(2)}$ and $S^{(1)}$ is described in the following proposition.

**Proposition 2.1.** If $J$ is the exchange matrix of order $n$, then $DJ$, $S^{(2)}J$, and $S^{(1)}$ are circulant matrices.

**Proof.** The representations above show that $D$ and $S^{(2)}$ are Hankel matrices. Therefore, $DJ$ and $S^{(2)}J$ are Toeplitz matrices. Besides, we have

$$d_{j+n} = d_j, \quad s^{(2)}_{j+n} = s^{(2)}_j, \quad j = 1, 2, \ldots, n,$$

so that $DJ$ and $S^{(2)}J$ are circulants.
Considering the Eqs. (2.8)-(2.9), we note that \( s_j^{(1)} = s_{j-n}^{(1)} \) for \( j = 1, 2, \cdots, n \). Hence, \( S^{(1)} \) is also a circulant matrix of order \( n \).

Let us now evaluate the computational cost of matrix-vector multiplication of \( Ax \) for the discrete linear system (2.10)-(2.11) if \( \Gamma \) is an ellipse.

**Proposition 2.2.** For any \( x \in \mathbb{R}^n \) the matrix vector product \( Ax \) can be mainly computed by three \( \text{FFT}(n) \) and one \( \text{IFFT}(n) \). Therefore, the computational cost for this operation is about \( 4p(n) \).

**Proof.** Diagonalising circulant matrices \( DJ, S^{(2)}J \) and \( S^{(1)} \), we obtain
\[
D = DJ \cdot J = F^* \cdot \text{diag}(\lambda_{DJ}) \cdot F \cdot J,
\]
\[
S^{(2)} = S^{(2)}J \cdot J = F^* \cdot \text{diag}(\lambda_{S^{(2)}J}) \cdot F \cdot J,
\]
\[
S^{(1)} = F^* \cdot \text{diag}(\lambda_{S^{(1)})} \cdot F.
\]

Set \( z := s^{(3)} \circ p \circ x \). It follows from (2.10) that
\[
A x = \frac{1}{2} x + D x + \left( S^{(1)} + S^{(2)} \right) \text{diag}(s^{(3)} \circ p \circ x)
= \frac{1}{2} x + F^* (\lambda_{DJ} \circ (F(Jx))) + F^* (\lambda_{S^{(1)}J} \circ (Fz)) + F^* (\lambda_{S^{(2)}J} \circ (F(Jz)))
= \frac{1}{2} x + F^* [\lambda_{DJ} \circ (F(Jx)) + \lambda_{S^{(1)}J} \circ (Fz) + \lambda_{S^{(2)}J} \circ (F(Jz))].
\]

(2.12)

Therefore, the main computational operations needed to derive \( Ax \) include three \( \text{FFT} \) for finding \( Jx, z, \) and \( Jz \), and one \( \text{IFFT} \) for \([\lambda_{DJ} \circ (F(Jx)) + \lambda_{S^{(1)}J} \circ (Fz) + \lambda_{S^{(2)}J} \circ (F(Jz))] \).  

Let us note that matrix-vector multiplication is the main computational operation in the Krylov subspace iterative methods for linear systems. The discussion above shows that for well-structured matrices \( A \), the cost of this operation is \( \mathcal{O}(n \log n) \), considerably smaller than for conventional matrix-vector multiplication. Moreover, the Eq. (2.12) shows that to perform this operation one has to store four vectors only — viz. \( \lambda_{DJ}, \lambda_{S^{(1)}J}, \lambda_{DS^{(2)}J}, \) and \( s^{(3)} \circ p \). Therefore, solving the linear system (2.11) by a Krylov subspace iterative method such as GMRES, requires only \( \mathcal{O}(n \log n) \) operations per iteration.

### 2.3. Partially elliptic domains

If \( \Gamma \) is not entirely an ellipse, the above algorithm still can be efficient. Such domains often arise in applications. In the next section we will consider the case where a small non-elliptic segment of the boundary represents a corroded or unknown edge of the material in \( \Omega \).

Let \( \Omega \) be a connected domain with boundary \( \Gamma \). Assume that \( \Gamma \) is the ellipse \( x_1^2/a^2 + x_2^2/b^2 = 1 \) except for a small part. More exactly, if the ellipse is parameterised as before, then the non-elliptic part of \( \Gamma \) corresponds to the parameter range \( [\tau_3, \tau_4] \) as follows
\[
x = (a \cos(2\pi t), b \sin(2\pi t)) + (0, \beta(t)), \quad 0 \leq t \leq 1,
\]
where \( \beta(t) \in C^2[0, 1] \) and \( \text{supp}(\beta) \subseteq [\tau_3, \tau_4] \).
For simplicity, we choose a shifted parameter $t$ for $\Gamma$ — i.e. 
\[ x(t) = (\phi(t), \psi(t)) = (a \cos(2\pi(t+\tau_4)), b \sin(2\pi(t+\tau_4)))+(0, \beta(t)), \quad 0 \leq t \leq 1, \] (2.13)
so that the non-elliptic segment corresponds to the parameter $t \in [1 + \tau_3 - \tau_4, 1]$ and
\[ \beta(t) = 0, \quad 0 \leq t \leq 1 + \tau_3 - \tau_4. \] (2.14)

Using the Eqs. (2.3), (2.13), and (2.14), we note that now
\[ K_d(t,s) = K_d^{(\text{ell})}(t + \tau_4, s + \tau_4), \quad 0 \leq t,s \leq 1 + \tau_3 - \tau_4, \]
and we can write
\[ K_d(t,s) = K_d^{(\text{ell})}(t + \tau_4, s + \tau_4) + K_{d,2}(t,s), \quad 0 \leq t,s \leq 1, \] (2.15)
where
\[ K_{d,2}(t,s) = 0, \quad 0 \leq t,s \leq 1 + \tau_3 - \tau_4. \]

Analogously, using the Eqs. (2.6), (2.7), (2.13) and (2.14), we represent the kernels $K_{s,2}$ and $K_{s,3}(s)$ in the form
\[ K_{s,2}(t,s) = K_{s,2}^{(\text{ell})}(t + \tau_4, s + \tau_4) + K_{s,2,2}(t,s), \quad 0 \leq t,s \leq 1 \] (2.16)
and
\[ K_{s,3}(s) = K_{s,3}^{(\text{ell})}(s + \tau_4) + K_{s,3,2}(s), \quad 0 \leq s \leq 1, \]
where
\[ K_{s,2,2}(t,s) = 0, \quad 0 \leq t,s \leq 1 + \tau_3 - \tau_4, \]
\[ K_{s,3,2}(s) = 0, \quad 0 \leq s \leq 1 + \tau_3 - \tau_4. \]

Let $t_1, t_2, \cdots, t_n$ be quadrature points and $n'$ be an index such that $t_n' < 1 + \tau_3 - \tau_4 \leq t_{n'+1}$. Then the matrices $\tilde{D}$ and $\tilde{S}^{(2)}$ constructed for the kernels (2.15) and (2.16) can be represented in the form
\[ \tilde{D} = D + D_2, \quad \tilde{S}^{(2)} = S^{(2)} + S_2^{(2)}, \]
where $DJ$ and $S^{(2)}J$ are circulants of order $n$, and $D_2$ and $S_2^{(2)}$ are sparse matrices — viz.
\[ [D_2]_{i,j} = 0, \quad [S_2^{(2)}]_{i,j} = 0, \quad 1 \leq i,j \leq n'. \]

Therefore, matrix $A$ can be written as
\[ A = \frac{1}{2} I + D + (S^{(1)} + S^{(2)}) \text{diag}(s^{(3)} \circ p) + [D_2 + S_2^{(2)} \text{diag}(s^{(3)} \circ p)], \] (2.17)
where the matrix in the square brackets is related to the distorted part of the ellipse and other matrices in the left-hand-side of (2.17) are the same as in the previously considered case of ellipse boundary.

It is easily seen that the storage requirement for $A$ is approximately equal to $(n^2 - n')$ and according to Proposition 2.2, the computational cost of the multiplication of $A$ by a vector is $4\rho(n) + 2(n^2 - n')$. For example, if $n' \approx 0.7n$ — i.e. if about 30% of ellipse is distorted, the cost of the above method of matrix-vector multiplication is approximately equal to $n^2 + 4\rho(n)$. It is lower than $2n^2$ in conventional matrix-vector multiplication.
3. Inverse Problems

Here we incorporate the fast algorithm into numerical methods for two inverse problems based on (1.1).

3.1. Recovery of the Robin coefficient

Assume \( p(x) \geq 0 \) and is piecewise continuous on \( \Gamma \) and \( \text{supp}(p) \subseteq \Gamma_1 \subset \Gamma \) is nonempty. Consider the problem of recovering the Robin coefficient \( p \) in (1.1) from a given measurement of \( u \): \( u_0 = u|_{\Gamma_0} \), where \( \Gamma_0 \subset \Gamma \) and \( \Gamma_0 \cap \Gamma_1 = \emptyset \). Such a problem, also known as the inverse Robin problem, appears in various nondestructive testing techniques, including quality evaluation of inaccessible metal-to-silicon contacts on \( \Gamma_1 \) in a transistor—cf. [5, 15].

We consider numerical methods for the inverse problem mentioned, which are based on the integral equation (2.2). Although there are many similar studies—cf. [5, 15, 16], here we use a slightly modified approach. To underscore the dependence of the operator \( \mathcal{A} \) and its discrete representation on \( p \), we denote them by \( \mathcal{A}(p) \) and \( A(p) \), respectively. The restriction operator from \( \Gamma \) to \( \Gamma_0 \) is referred to as \( \mathcal{R}_0 \)—i.e. \( \mathcal{R}_0 u = u|_{\Gamma_0} \). The Robin inverse problem can be now formulated as follows: Find \( p \) such that the solution of \( \mathcal{A}(p)(u) = f \) satisfies the equation

\[
\mathcal{R}_0 u = u_0,
\]

where \( u_0 \) is a given measurement of \( u \) on \( \Gamma_0 \).

Since inverse problems are usually ill-posed, we reformulate the Eq. (3.1) as a minimisation problem with regularisation—viz.

\[
\min_{p,u} \frac{1}{2} \| \mathcal{R}_0 u - u_0 \|_{L^2(\Gamma_0)}^2 + \frac{\mu}{2} J(p), \quad \text{subject to} \quad \mathcal{A}(p)(u) = f,
\]

where \( J(p) \) is a regularisation functional for \( p \) and \( \mu > 0 \) a regularisation parameter. We use the \( H^1 \)-regularisation functional

\[
J(p) = \int_{\Gamma_1} \left| \frac{\partial p(x)}{\partial \gamma} \right|^2 ds_x,
\]

where \( \gamma \) is the unit tangent direction and \( ds_x \) the arc length differential on \( \Gamma_1 \). After that we employ a penalty method and convert the constrained optimisation problem (3.2) into the problem

\[
\min_{p,u} \frac{1}{2} \| \mathcal{A}(p)(u) - f \|_{L^2(\Gamma)}^2 + \frac{\alpha}{2} \| \mathcal{R}_0 u - u_0 \|_{L^2(\Gamma_0)}^2 + \frac{\mu}{2} J(p)
\]

with a scaling parameter \( \alpha > 0 \). The formulation (3.3) can also be considered as a nonlinear least squares method with regularisation for determining \( (p, u) \) from the following inconsistent system of equations

\[
\mathcal{A}(p)(u) = f,
\]

\[
\alpha \mathcal{R}_0 u = au_0.
\]
Considering the above parameterisation of $\Gamma$ and the same discretisation method, we assume that
\[ \{t_{m_1}, \cdots, t_{m_2}\} = \{t_1, t_2, \cdots, t_n\} \cap \{t : t \in [\tau_1, \tau_2]\}, \]
where subinterval $[\tau_1, \tau_2]$ defines the arc $\Gamma_1 := \{(\phi(t), \psi(t)) : \tau_1 \leq t \leq \tau_2\}$ containing the support of $p$.

For the sake of convenience, we continue to use the notation $p$ in the discrete representation of $p$ on $\Gamma_1$. Thus we have
\[ p = [p(\phi(t_{m_1}), \psi(t_{m_1})), \cdots, p(\phi(t_{m_2}), \psi(t_{m_2}))] = [p_1, \cdots, p_m], \]
where $m = m_2 - m_1 + 1$. According to the discretisation method, the functional $J(p)$ is approximated as follows
\[ J_H(p) = \sum_{i=0}^{m} (p_{i+1} - p_i)^2 = p_1^2 + \sum_{i=1}^{m-1} (p_{i+1} - p_i)^2 + p_m^2 \]
with the convention $p_0 = p_{m+1} = 0$. This representation can be also written as
\[ J_H(p) = p^T T p, \]
where $T = L^T L$ is the tridiagonal Toeplitz matrix generated by the matrix
\[
L = \begin{pmatrix}
1 & 0 & \cdots & 0 \\
-1 & 1 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 1 \\
0 & 0 & \cdots & -1
\end{pmatrix}_{(m+1) \times m}
\]
Thus $T$ is a symmetric Toeplitz matrix with the first row $[2, -1, 0, \cdots, 0]$.

Let $\Gamma_0 = \{(\phi(t), \psi(t)) : \tilde{\tau}_1 \leq t \leq \tilde{\tau}_2\}$ be the part of $\Gamma$ where measurements of $u$ are collected and
\[ \{t_{\tilde{m}_1}, \cdots, t_{\tilde{m}_2}\} := \{t_1, t_2, \cdots, t_n\} \cap \{t : t \in [\tilde{\tau}_1, \tilde{\tau}_2]\}. \]
Moreover, we consider the discrete representation $R_0 = [O_{\tilde{m} \times (\tilde{m} - 1)} \; I_{\tilde{m}} \; O_{\tilde{m} \times (n - \tilde{m})}]$ of the restriction operator $\mathcal{R}_0$ with $\tilde{m} = \tilde{m}_2 - \tilde{m}_1 + 1$. Consequently, the discrete form of (3.3) is
\[ \min_{p, u} \frac{1}{2} \| A(p) u - f \|_2^2 + \frac{\alpha}{2} \| R_0 u - u_0 \|_2^2 + \frac{\mu}{2} p^T T p, \quad (3.4) \]
where $u_0$ refers to the discrete representation of the data $u_0$ on $\Gamma_0$.

In order to find the solution of the minimisation problem (3.4), we use the Gauss-Newton (GN) method
\[ p^{(k+1)} = p^{(k)} + \Delta p^{(k)}, \quad u^{(k+1)} = u^{(k)} + \Delta u^{(k)} \]
with the correction terms $\Delta p^{(k)}, \Delta u^{(k)}$ determined by the normal equations

$$
B\left( p^{(k)}, u^{(k)} \right) \begin{pmatrix} \Delta u^{(k)} \\ \Delta p^{(k)} \end{pmatrix} = v^{(k)},
$$

(3.5)

where

$$
B\left( p^{(k)}, u^{(k)} \right) = \begin{pmatrix}
A\left( p^{(k)} \right)^T A\left( p^{(k)} \right) + \alpha R_0^T R_0 & A\left( p^{(k)} \right)^T G\left( u^{(k)} \right) \\
G\left( u^{(k)} \right)^T A\left( p^{(k)} \right) & G\left( u^{(k)} \right)^T G\left( u^{(k)} \right) + \mu T
\end{pmatrix},
$$

and

$$
v^{(k)} = \begin{pmatrix}
A\left( p^{(k)} \right)^T e_1^{(k)} + \alpha R_0 e_2^{(k)} \\
G\left( u^{(k)} \right)^T e_1^{(k)} - \mu T p^{(k)}
\end{pmatrix}.
$$

Here $R_1 = [O_{m \times (m_1 - 1)} I_m O_{m \times (n - m_2)}]$ is the discrete representation of the restriction operator from $\Gamma$ to $\Gamma_1$.

Although this setup is applicable to curves of any shape, we now consider the case where $\Gamma$ is an ellipse in order to demonstrate the use of the fast algorithm above in GN algorithm for (3.5). Since $B(p^{(k)}, u^{(k)})$ is positive definite, we now apply the conjugate gradient (CG) method to (3.5).

In order to find the right hand side of (3.5) we shall first determine the terms $A(p^{(k)})u^{(k)}$, $A(p^{(k)})^T e_1^{(k)}$, $G(u^{(k)})^T e_1^{(k)}$ and $T p^{(k)}$. They can be evaluated by respective use of 3 FFT($n$) + 1 IFFT($n$), 3 FFT($n$) + 1 IFFT($n$), 2 FFT($n$) + 1 IFFT($n$) and 3n multiplications. Therefore, the total computational cost required is approximately equal to $11\rho(n)$. On the other hand, it would be $6n^2$ if the special structure of the relevant matrices is not taken into account.

For $r_1 \in \mathbb{R}^n$ and $r_2 \in \mathbb{R}^m$, one has

$$
B\left( p^{(k)}, u^{(k)} \right) \begin{pmatrix} r_1 \\ r_2 \end{pmatrix} = \begin{pmatrix}
A\left( p^{(k)} \right)^T \left[ A\left( p^{(k)} \right) r_1 + G\left( u^{(k)} \right) r_2 \right] + \alpha R_0^T R_0 r_1 \\
G\left( u^{(k)} \right)^T \left[ A\left( p^{(k)} \right) r_1 + G\left( u^{(k)} \right) r_2 \right] + \mu T r_2
\end{pmatrix},
$$

and it is easily seen that for any $n+m$ vector $r = (r_1^T, r_2^T)^T$, the fast algorithm in the matrix-vector multiplication of $B(p^{(k)}, u^{(k)})r$ requires about $14\rho(n)$ operations. On the other hand, if the structure of $B(p^{(k)}, u^{(k)})$ is not taken into account, at each GN step one has to form $A(p^{(k)})$ before starting the matrix-vector multiplication $B(p^{(k)}, u^{(k)})r$. It would require $2mn$ operations and the cost of the matrix-vector multiplication will be about $4n(m + n)$. Since matrix-vector multiplications is repeatedly used in solution of (3.4), the reduction of computational cost by fast matrix-vector multiplication can be very significant.

The application of the Gaussian elimination to (3.5) requires the construction of the matrix $B(p^{(k)}, u^{(k)})$ with approximate computational cost $n(n^2 + mn + m^2)$ operations and additional $(m + n)^3/3$ operations are needed to carry out the Gaussian elimination process itself.
3.2. Recovery of part of boundary

Now we assume that \( p(x) \) is given on \( \Gamma \) and consider the problem of recovering a part \( \Gamma_1 \subseteq \text{supp}(p) \) of the boundary from known values of the solution \( u \) of (1.1) on \( \Gamma_0 \), where \( \Gamma_0 \cap \Gamma_1 = \emptyset \). This problem appears in thermal detection of corrosion of an accessible part of the material boundary \([3, 6]\). In order to recover \( \Gamma_1 \) via integral equation (2.2), we will follow the approach in \([6]\), but focus on the fast algorithm from Subsection 2.3 to improve the efficiency of the method.

More exactly, we consider a parametrisation

\[
\Gamma = \{(\phi(t), \psi(t)) = (\phi_0(t), \psi_0(t) + \beta(t)) : 0 \leq t \leq 1\}
\]  

(3.6)
of \( \Gamma \) such that \( \text{supp}(\beta) \subseteq [1 - \tau, 1] \) for some \( 0 < \tau < 1 \) — i.e. the parameter range for \( \Gamma_1 \) is \([1 - \tau, 1]\) and the function \( \beta(t) \) is responsible for the distorted part \( \Gamma_1 \). Therefore, in order to describe \( \Gamma_1 \) one has to determine the function \( \beta(t) \). To underscore the dependence of \( \Gamma_1 \) on \( \beta \), the corresponding operators in (2.2) are written as \( \mathcal{A}(\beta) \), \( \mathcal{D}(\beta) \), \( \mathcal{S}(\beta) \). Thus

\[
\mathcal{A}(\beta)u := \left( \frac{1}{2}\mathcal{I} + \mathcal{D}(\beta) \right)u + \mathcal{S}(\beta)(pu), \quad f = \mathcal{S}(\beta)g.
\]

Following \([6]\) and similar formulations here, the solution of the inverse Robin problem is determined from the optimisation problem

\[
\min_{\beta, u} \frac{1}{2} \| \mathcal{A}(\beta)u - \mathcal{S}(\beta)g \|_{L^2(\Gamma)}^2 + \frac{\alpha}{2} \| \mathcal{R}_0 u - u_0 \|_{L^2(\Gamma_0)}^2 + \frac{\mu}{2} \| \beta' \|^2_2.
\]  

(3.7)

We represent the function \( \beta(t) \) in the form

\[
\beta(t) = \sum_{j=1}^{\tilde{n}} q_j \xi_j(t),
\]

where \( \xi_j(t) \in C^2[0, 1] \) are basis functions such that \( \xi_j(t) = 0 \). Setting

\[
\xi(t) := [\xi_1(t), \cdots, \xi_{\tilde{n}}(t)]^T, \quad q := [q_1, \cdots, q_{\tilde{n}}]^T,
\]

we write \( \beta \) as

\[
\beta(t) = \xi(t)^T q.
\]

Thus \( \Gamma_1 \) is now represented by the vector \( q \). The operators \( \mathcal{A} \) and \( \mathcal{S} \) also depend on \( q \) and so are their discrete representations. Consequently, problem (3.7) has the discrete representation

\[
\min_{q, u} \frac{1}{2} \left\| A(q)u - \left( S^{(1)} + S^{(2)}(q) \right) \left( S^{(3)}(q) \circ g \right) \right\|_2^2 + \frac{\alpha}{2} \| R_0 u - u_0 \|_2^2 + \frac{\mu}{2} \| Bi u \|_2^2,
\]  

(3.8)

where \( B = [\xi_j'(t_i)]_{i=1:n} \) and other matrices are defined in (2.10).
Using the Gauss-Newton iteration method to solve (3.8), we arrive at the following normal system for the correction terms

\[
\begin{pmatrix}
A(q^{(k)})^T A(q^{(k)}) + \alpha R_0^T R_0 & A(p^{(k)})^T Q(q^{(k)}, u^{(k)}) \\
Q(q^{(k)}, u^{(k)})^T A(q^{(k)}) & Q(q^{(k)}, u^{(k)})^T Q(q^{(k)}, u^{(k)}) + \mu B^T B
\end{pmatrix} \times
\begin{pmatrix}
\Delta u^{(k)} \\
\Delta q^{(k)}
\end{pmatrix} =
\begin{pmatrix}
A(q^{(k)})^T e_1^{(k)} + \alpha R_0^T e_2^{(k)} \\
Q(q^{(k)}, u^{(k)})^T e_1^{(k)} - \mu B^T B q^{(k)}
\end{pmatrix}
\tag{3.9}
\]

with the vectors

\[
e_1^{(k)} = (s^{(1)} + s^{(2)}(q^{(k)}))(s^{(3)}(q^{(k)}) \circ \varphi) - A(q^{(k)}) u^{(k)}, \quad e_2^{(k)} = u_0 - R_0 u^{(k)}.
\]

We note that \(Q(q^{(k)}, u^{(k)})\) is an \(n \times \bar{n}\) matrix determined by the derivatives of \(D(q)\) and \(S(q)\) and the basis functions — cf. [6].

Let \(C(q^{(k)}, u^{(k)})\) denote the coefficient matrix of (3.9). It is clear that for full column rank matrices \(B\), the matrix \(C(q^{(k)}, u^{(k)})\) is positive definite and CG method is applicable to (3.9).

In what follows, we consider the case where \(\{ (\phi_0(t), \psi_0(t)) : 0 \leq t \leq 1 \}\) in (3.6) is an ellipse and apply the fast algorithm of Section 2.3 to the system (3.9). In this case, the cost of computing \(A(q^{(k)}) r\) is approximately equal to \(4\rho(n) + 2(n^2 - n^2)\). During the \(k\)-th iteration in GN, we have to determine \(Q(q^{(k)}, u^{(k)})\). Each column of this matrix is obtained by computing and multiplying two matrices by various vectors. These matrices correspond to the derivatives of \(D(q)\) and \(S(q)\) and are as sparse as \(D_2\) and \(S_2^{(2)}\) at least. The sparsity of the matrices yields that each column can be obtained in \(O(nl)\) operations, where \(l\) is determined by the support of \(\xi_i, i = 1, 2, \cdots, \bar{n}\). It follows that the total cost for establishing \(Q(q^{(k)}, u^{(k)})\) is \(O(nl)\). Since \(\text{supp}(\xi_i) \subset \text{supp}(\beta) \subset [1 - \tau, 1]\), then \(l \leq n - n'\) and \(l \ll n\) in general. For instance, in a forthcoming example, we choose \(\tau = 0.3\) and \(\bar{n} = 12\), so that the length of each \(\text{supp}(\xi_i)\) is \(4\tau/(\bar{n} + 3) \approx 0.0193\) and \(l \approx 0.0193n\).

Let \(r\) be an \(n + \bar{n}\) vector. One can check that with the fast algorithm using special matrix structures, the matrix-vector multiplication \(C(q^{(k)}, u^{(k)}) r\) will require about \(8\rho(n) + 4n\bar{n} + 4(n^2 - n^2)\) operations instead of \(4n^2 + 4n\bar{n}\) ones for the standard approach. For example, if \(n' \approx 0.7n\) — i.e. if about 30% of \(\Gamma\) differ from ellipse boundary, the fast algorithm reduces the cost from \(4n^2 + 4n\bar{n}\) to about \(2n^2 + 8\rho(n) + 4n\bar{n}\) operations.

The Gaussian elimination method requires \(n(n^2 + n\bar{n} + 2n^2)\) operations to form the matrix \(C(q^{(k)}, u^{(k)})\) and additional \((n + \bar{n})^3/3\) ones in the elimination procedure.

4. Numerical Examples

We now consider numerical examples to show the efficiency of the above algorithms. In particular, we compare the computation time with the time required by the methods not using special structures of discrete systems. Computations are carried out in Matlab environment on a Lenovo computer with Intel(R) Core(TM) i7-4790U CPU@3.60GHz 3.60GHz,
W. F. ang, F.-R. Lin and Y.-B. Ma

RAM 4.0GB. Computation time is given in seconds for various number of quadrature points \( n \).

Considering the forward problem (2.11), we compare three methods — viz.

- **GMRES(f):** solve linear system (2.11) by a GMRES method with fast matrix-vector multiplication of Section 2.
- **GMRES(c):** solve linear system (2.11) by a GMRES method with conventional matrix-vector multiplication.
- **GE:** solve linear system (2.11) by Gaussian elimination.

In the GMRES methods, we choose zero vector as the initial guess and the stopping criterion

\[
\frac{\|r^{(k)}\|_2}{\|r^{(0)}\|_2} < 10^{-7},
\]

where \( r^{(k)} = f - Au^{(k)} \) is the residual vector for the \( k \)-th iteration.

**Example 4.1** (Forward problem for elliptic domain). Consider the Robin boundary value problem (1.1) in the integral equation form (2.2), where \( \Gamma \) is an ellipse with the parametrisation

\[
x = (\cos(2\pi t), 0.3 \sin(2\pi t)), \quad 0 \leq t \leq 1.
\]

The functions \( g \) and \( p \) have the form

\[
g(t) := \begin{cases} 1, & t \in [0.5, 0.6], \\ 0, & \text{otherwise,} \end{cases}
\]

\[
p(t) := \begin{cases} c(t - 0.1)^3(0.4 - t)^3, & t \in [0.1, 0.4], \\ 0, & \text{otherwise,} \end{cases}
\]

where \( c \) is a coefficient such that \( \max\{p(t) : 0 \leq t \leq 1\} = 10 \).

We first find an approximate solution \( \hat{u} \) of (2.11) with \( n = 8192 \), and will use it as the benchmark solution to estimate the errors in methods with lower numbers of quadrature points. The relative error is measured as

\[
err_{n, \infty} = \frac{\max_{j=1,\ldots,n} |u_j^* - \hat{u}_{1+(j-1)8192/n}|}{\max_{j=1,\ldots,n} |\hat{u}_{1+(j-1)8192/n}|},
\]

where \( u^* \) refers to approximate solutions for \( n \) quadrature points.

The computation time for all methods with different number of quadrature points is shown in Table 1. We observe that GMRES(f) with the fast algorithm from Section 2.2, requires considerably less computation time, especially for large \( n \). Moreover, the algorithm converges linearly and a very large \( n \) is needed to derive accurate numerical solutions.
Example 4.2 (Forward problem for partially elliptic domain). Let the boundary $\Gamma$— cf. Fig. 1, has the parametrisation

$$x = (\cos(2\pi(t + 0.4)), 0.3 \sin(2\pi(t + 0.4)) + \beta(t)), \quad 0 \leq t \leq 1,$$

where

$$\beta(t) = \begin{cases} 
0, & \text{if } t \in (0, 0.7), \\
\text{B-spline,} & \text{if } t \in [0.7, 1].
\end{cases}$$

The functions $g(t)$ and $p(t)$ have the form

$$g(t) = \begin{cases} 
1, & t \in [0.1, 0.2], \\
0, & \text{elsewhere},
\end{cases}$$

$$p(t) = \begin{cases} 
10, & t \in [0.7, 1], \\
0, & \text{elsewhere}.
\end{cases}$$

The computation time for three methods with different number of quadrature points is shown in Table 2. Although for small $n$, the Gaussian elimination is the fastest method, for $n \geq 512$ the GMRES(f) with the fast algorithm from Subsection 2.3 in GMRES(f) with $n' \approx 0.7n$ becomes the fastest one. The relative errors are also shown in Table 2. Similar to Example 4.1, the algorithm converges linearly and a very large $n$ is needed to achieve a high accuracy.

Considering the inverse problems from Section 3, we compare three methods for solving the normal systems at each CN step — viz.

- CG(f): Form the coefficient matrix implicitly by computing matrices and vectors needed for matrix-vector multiplications in (3.5) or (3.9) and solve the corresponding linear systems by the CG method with fast matrix-vector multiplication from Section 2.
• CG(c): Form the coefficient matrix of (3.5) or (3.9) explicitly and solve the corresponding linear systems by the CG method with conventional matrix-vector multiplication.

• GE: Form the coefficient matrix in (3.5) or (3.9) explicitly, and solve the corresponding linear systems by the Gaussian elimination.

The GN method is terminated after 30 iterations or if the norm of the correction vector does not exceed $10^{-5}$. In CG methods the zero vector is taken as the initial guess. Since the linear system (3.5) and (3.9) may be very ill-conditioned, the stopping criterion is set as

$$\text{either } \frac{\|r^{(j)}\|_2}{\|r^{(0)}\|_2} < 10^{-5} \text{ or } j \geq 500,$$

where $r^{(j)}$ is the residual vector at the $j$-th iterate.

**Example 4.3** (Recovery of the Robin coefficient). Let $\Gamma$, $g(t)$ and $p(t)$ be the same as in Example 4.1. Assume we know the values of $u$ at certain points of the arc $\Gamma_0 = \{(\cos(2\pi t), 0.3\sin(2\pi t)) : 0.6 \leq t \leq 0.9\}$ and data $u_0$ are generated by adding white noise to the solution $u$, thus obtaining

$$u_0 = u(m_1 : m_2) + \delta [\text{rand}(m, 1) - 0.5],$$

where $m = m_2 - m_1 + 1$ and rand denotes random numbers of the uniform distribution of the interval $(0, 1)$. If $\hat{u}$ is the approximate solution of (2.11) with $n = 8192$, then $u_{j} = \hat{u}_{1+(j-1)8192/n}$ is $j$-th entry of $u$.

The optimisation problem (3.4) has two unknown positive parameters $\alpha$ and $\mu$. We determine them by the following procedure. Starting with certain $\alpha$ and $\mu$, we find an
approximate solution of (3.4) by CG(f) and obtain a pair \((p^*, u^*)\). This pair is then used to adjust the parameters \(\alpha\) and \(\mu\) using the following method:

- The scaling parameter \(\alpha\) is in fact a weight for the data fidelity term \(\|R_0 u - u_0\|_2\). Therefore, if the relative error 
  \[
  \frac{\|A(p^*)u^* - f\|_2}{\|f\|_2}
  \]
  is not sufficiently small, say greater than \(10^{-6}\), we choose a smaller \(\alpha\). On the other hand, if the term \(\|A(p^*)u^* - f\|_2/\|f\|_2\) is too small, we replace \(\alpha\) by a slightly larger one.

- The regularisation parameter \(\mu\) is selected according to the normalised cumulative periodogram (NCP) method \([11]\). If the residual vector \(R_0 u^* - u_0\) is dominated by white noise, then \(\mu\) is a suitable choice. However, if \(R_0 u^* - u_0\) is dominated by high frequency noise, the current parameter \(\mu\) shall be replaced by a larger one. On the other hand, if \(R_0 u^* - u_0\) is dominated by a low frequency signal, \(\mu\) shall be reduced to a smaller value \([7]\).

We start with the initial guess 
\[p_0 = \text{ones}(m, 1) \in \mathbb{R}^m.\]

The computation time for the whole procedure, the number of steps in GN, and the average number of iterations for solving (3.5) with \(n = 1024\) and \(n = 2048\) are, respectively, shown in Tables 3 and 4. The problem (3.5) is severely ill-conditioned, so that the GM method needs a large number of iterations and does not allow to achieve a high accuracy. As a result, more GN steps have to be carried out. Therefore, CG(c) requires more computation time than the Gaussian elimination method. However, for large \(n\) the use of the special structure of coefficient matrices and the Fourier transform technique — i.e. CG(f) leads to a smaller total computation time than in the Gaussian elimination.

We now consider the accuracy for the recovery of the Robin coefficient with respect to various noise levels and the numbers of quadrature points. Analogously to Examples 4.1 and 4.2, to measure the accuracy we use relative errors 
\[
\frac{\|p^* - p\|_\infty}{\|p\|_\infty},
\]
where \(p^*\) and \(p\) are the vectors of recovered Robin and exact Robin coefficients, respectively. The relative errors are displayed in Table 5 and the regularisation parameters \(\alpha\) and \(\mu\) in Table 6. The exact Robin coefficient and the recovered ones for \(n = 1024\) and noise level 0.01 and for \(n = 2048\) and noise level 0.001 are presented in Fig. 2.

As Table 5 shows, for a high noise level 0.1, the method using more quadrature points may not provide better results. On the other hand, for low noise level, the method with a larger number of quadrature points delivers better results.
Table 3: Example 4.3. Approximate solution of (3.5). $n = 1024$ and noise level is 0.01.

<table>
<thead>
<tr>
<th>CPU time</th>
<th>GN Steps</th>
<th>Average number of iterations for CG</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG(f)</td>
<td>1.5617</td>
<td>24</td>
</tr>
<tr>
<td>CG(c)</td>
<td>9.4140</td>
<td>30</td>
</tr>
<tr>
<td>GE</td>
<td>1.5902</td>
<td>9</td>
</tr>
</tbody>
</table>

Table 4: Example 4.3. Approximate solution of (3.5). $n = 2048$ and noise level is 0.01.

<table>
<thead>
<tr>
<th>CPU time</th>
<th>GN Steps</th>
<th>Average number of iterations for CG</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG(f)</td>
<td>7.8606</td>
<td>30</td>
</tr>
<tr>
<td>CG(c)</td>
<td>63.4653</td>
<td>30</td>
</tr>
<tr>
<td>GE</td>
<td>9.5619</td>
<td>9</td>
</tr>
</tbody>
</table>

Table 5: Example 4.3. Robin coefficient recovery: Errors.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Noise Level = 0.1</th>
<th>Noise Level = 0.01</th>
<th>Noise Level = 0.001</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>0.1990</td>
<td>0.0706</td>
<td>0.0675</td>
</tr>
<tr>
<td>1024</td>
<td>0.1848</td>
<td>0.0663</td>
<td>0.0564</td>
</tr>
<tr>
<td>2048</td>
<td>0.1816</td>
<td>0.0444</td>
<td>0.0301</td>
</tr>
</tbody>
</table>

Table 6: Example 4.3. Parameters $(\alpha, \mu)$.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Noise Level = 0.1</th>
<th>Noise Level = 0.01</th>
<th>Noise Level = 0.001</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>$(1.0e^{-5}, 1.0e^{-9})$</td>
<td>$(1.0e^{-5}, 1.0e^{-10})$</td>
<td>$(1.0e^{-5}, 5.0e^{-11})$</td>
</tr>
<tr>
<td>1024</td>
<td>$(1.0e^{-5}, 1.0e^{-10})$</td>
<td>$(2.0e^{-5}, 9.0e^{-11})$</td>
<td>$(1.0e^{-5}, 6.0e^{-11})$</td>
</tr>
<tr>
<td>2048</td>
<td>$(1.0e^{-5}, 5.0e^{-9})$</td>
<td>$(2.0e^{-5}, 7.0e^{-10})$</td>
<td>$(1.0e^{-5}, 6.0e^{-11})$</td>
</tr>
</tbody>
</table>

Figure 2: Example 4.3. Exact and recovered profiles. Left: $n = 1024$, noise level 0.01. Right: $n = 2048$, noise level 0.001.
Example 4.4 (Recovery of a boundary part). Choosing the same functions \( g(t) \) and \( p(t) \) as in Example 4.2, we will try to recover the part \( \Gamma_1 \) of the boundary \( \Gamma \). Measurements of \( u \) are available for the arc

\[
\Gamma_0 = \{ \cos(2\pi(t + 0.4)), 0.3 \sin(2\pi(t + 0.4)) : t \in [0.2, 0.5] \}
\]

and the data vector \( u_0 \) is generated similar to Example 4.3.

The function \( \beta(t) \) is a combination of five B-splines. In numerical experiments we set \( \tilde{n} = 12 \) and start with \( q_0 = \text{zeros}(12, 1) \in \mathbb{R}^{12} \) as an initial guess. The computation time of the whole solution process, the number of GN steps and the average number of iterations for the Eq. (3.9) for \( n = 1024 \) and \( n = 2048 \) are shown in Tables 7 and 8, respectively. Thus fast algorithm based on special matrix structures of the coefficient matrices reduces computation time especially for large \( n \).

Finally, we evaluate the quality of the boundary recovering with respect to various levels of noise and different number of quadrature points. The quality is measured by relative errors. If \((x, y)\) and \((x, y^*)\) are exact and recovered boundaries, the relative error is defined by

\[
\frac{\|y^* - y\|_\infty}{\|y\|_\infty}.
\]

The relative errors for the boundary recovery are demonstrated in Table 9 and the regularisation parameters \( \alpha \) and \( \mu \) in Table 10.

Table 7: Example 4.4. Computation time, number of GN steps and average number of iterations for the Eq. (3.9), \( n = 1024 \), noise level 0.01.

<table>
<thead>
<tr>
<th>CPU time</th>
<th>GN Steps</th>
<th>Average number of iterations for CG</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG(f)</td>
<td>10.4477</td>
<td>30</td>
</tr>
<tr>
<td>CG(c)</td>
<td>11.7719</td>
<td>30</td>
</tr>
<tr>
<td>GE</td>
<td>11.2914</td>
<td>30</td>
</tr>
</tbody>
</table>

Table 8: Example 4.4. Computation time, number of GN steps and average number of iterations for the Eq. (3.9), \( n = 2048 \), noise level 0.01.

<table>
<thead>
<tr>
<th>CPU time</th>
<th>GN Steps</th>
<th>Average number of iterations for CG</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG(f)</td>
<td>58.9262</td>
<td>30</td>
</tr>
<tr>
<td>CG(c)</td>
<td>67.3127</td>
<td>30</td>
</tr>
<tr>
<td>GE</td>
<td>65.4905</td>
<td>30</td>
</tr>
</tbody>
</table>

Table 9: Example 4.4. Boundary recovery. Errors, noise and number of quadrature points.

<table>
<thead>
<tr>
<th>( n )</th>
<th>Noise Level = 0.1</th>
<th>Noise Level = 0.01</th>
<th>Noise Level = 0.001</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>0.3812</td>
<td>0.2049</td>
<td>0.1425</td>
</tr>
<tr>
<td>1024</td>
<td>0.1915</td>
<td>0.1352</td>
<td>0.1070</td>
</tr>
<tr>
<td>2048</td>
<td>0.1421</td>
<td>0.1006</td>
<td>0.0308</td>
</tr>
</tbody>
</table>
Table 10: Example 4.4. Boundary recovery. Parameters $(\alpha, \mu)$.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Noise Level = 0.1</th>
<th>Noise Level = 0.01</th>
<th>Noise Level = 0.001</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>$(5.0e^{-6}, 2.0e^{-15})$</td>
<td>$(1.0e^{-5}, 2.0e^{-15})$</td>
<td>$(5.0e^{-5}, 2.0e^{-15})$</td>
</tr>
<tr>
<td>1024</td>
<td>$(1.0e^{-5}, 1.0e^{-14})$</td>
<td>$(1.0e^{-5}, 3.0e^{-15})$</td>
<td>$(1.0e^{-5}, 2.0e^{-15})$</td>
</tr>
<tr>
<td>2048</td>
<td>$(1.0e^{-5}, 1.0e^{-14})$</td>
<td>$(1.0e^{-5}, 1.0e^{-15})$</td>
<td>$(5.0e^{-5}, 1.0e^{-17})$</td>
</tr>
</tbody>
</table>

Figure 3: Example 4.4. Exact and recovered boundaries. Left: $n = 1024$. noise level 0.01. Right: $n = 2048$ noise level 0.001.

It is clearly seen that for low noise level and large number of quadrature points the recovery results are much better. For a fixed noise level, the increase of the number of quadrature points leads to better results. Exact and recovered boundaries are shown in Fig. 3.

5. Concluding Remarks

The Robin boundary value problem for the Laplace equation arises in various applications. Approximation methods using equivalent boundary integral equations have a number of advantages and can be employed for solving related inverse problems. We show that for ellipse boundary, the coefficient matrices of discretisation systems have a special structure. This fact is used to develop fast numerical algorithms. Such an approach is especially helpful in numerical methods for inverse problems, since many methods of their solution repeatedly use forward solvers. The efficiency of the methods is illustrated by numerical examples. This approach can be also of interest for the Robin boundary value problem in special three-dimensional domains.

Acknowledgments

The authors thank the anonymous referees for their valuable comments and suggestions which improved the quality of the paper.
This work was supported by the National Natural Science Foundation of China under Grant No. 11271238 and by the Guangdong Provincial Department of Education Innovating Strong School Project under Grant No. 2016KTSCX085. Parts of this work were completed when one of us (F.-R.L.) visited the Department of Mathematics and Statistics at Wright State University, the hospitality of which is greatly appreciated.

References