Solving integral equations on piecewise smooth boundaries using the RCIP method: a tutorial

Helsing, Johan

Published in:
Abstract and Applied Analysis

DOI:
10.1155/2013/938167

2013

Solving integral equations on piecewise smooth boundaries using the RCIP method: a tutorial

Johan Helsing
Centre for Mathematical Sciences
Lund University, Box 118, SE-221 00 Lund, Sweden

February 18, 2013 (revised August 21, 2014)

Abstract

Recursively Compressed Inverse Preconditioning (RCIP) is a numerical method for obtaining highly accurate solutions to integral equations on piecewise smooth surfaces. The method originated in 2008 as a technique within a scheme for solving Laplace's equation in two-dimensional domains with corners. In a series of subsequent papers the technique was then refined and extended as to apply to integral equation formulations of a broad range of boundary value problems in physics and engineering. The purpose of the present paper is threefold: First, to review the RCIP method in a simple setting. Second, to show how easily the method can be implemented in MATLAB. Third, to present new applications of RCIP to integral equations of scattering theory on planar curves with corners.

1 Introduction

This paper is about an efficient numerical solver for elliptic boundary value problems in piecewise smooth domains. Such a solver is useful for applications in physics and engineering, where computational domains of interest often have boundaries that contain singular points such as corners and triple junctions. Furthermore, elliptic boundary value problems in non-smooth domains are difficult to solve irrespective of what numerical method is used. The reason being that the solution, or the quantity representing the solution, often exhibits a non-smooth behavior close to boundary singularities. That behavior is hard to resolve by polynomials, which underlie most approximation schemes. Mesh refinement is needed. This is costly and may lead to artificial ill-conditioning and the loss of accuracy.

The numerical solver we propose takes its starting point in an integral equation reformulation of the boundary value problem at hand. We assume that the problem can be modeled as a Fredholm second kind integral equation with integral operators that are compact away from singular boundary points and whose solution is a layer density representing the solution to the original problem. We then seek a discrete approximation to the layer density using a modification of the Nyström method [1, Chapter 4]. At the heart of the solver lie certain integral transforms whose inverses modify the kernels of the integral operators in such a way that the layer density becomes piecewise smooth and simple to resolve by polynomials. The discretizations of these inverses are constructed recursively on locally refined temporary meshes. Conceptually, this corresponds to applying a fast direct solver [29] locally to regions with troublesome geometry. Finally, a global iterative method is applied. This gives us many of the advantages of fast direct methods, for example the ability to deal with certain classes of operators whose spectra make them unsuitable for iterative methods. In addition, the approach is typically much faster than using only a fast direct solver.

Our method, or scheme, has previously been referred to as *recursive compressed inverse preconditioning* [14, 15, 21, 22] and there is a good reason for that name: the scheme relies on applying a relieving right inverse to the integral equation; on compressing this inverse to a low-dimensional subspace; and on carrying out the compression in a recursive manner. Still, the name *recursive(ly) compressed inverse preconditioning* is a bit awkward and we will here simply use the acronym RCIP.

A strong motivation for writing the present paper is that the original references [14, 15, 21, 22] are hard to read for non-experts and that efficient codes, therefore, could be hard to implement. Certain derivations in [14, 15, 21, 22] use complicated intermediary constructions, application specific issues obscure the general picture, and the notation has evolved from paper to paper. In the present paper we focus on the method itself, on how it works and how it can be implemented, and refer to the original research papers for details. Demo codes in Matlab, updated as of August 2014, are a part of the exposition and can be downloaded from:

`http://www.maths.lth.se/na/staff/helsing/Tutor/`

Section 2 provides some historical background. The basics of the RCIP method are then explained by solving a simple model problem in Sections 3–6. Sections 7–15 review various extensions and improvements. Some of this material is new, for example the simplified treatment of composed operators in Section 14. The last part of the paper, Sections 16–18, contain new applications to scattering problems.

2 Background

The line of research on fast solvers for elliptic boundary value problems in piecewise smooth domains, leading up to the RCIP method, grew out of work in computational fracture mechanics. Early efforts concerned finding efficient integral equation formulations. Corner singularities were either re-solved with brute force or by using special basis functions, see [13, 18, 24] for examples. Such strategies, in combination with fast multipole [11] accelerated iterative solvers, work well for simple small-scale problems.

Real world physics is more complicated and, for example, the study [10] on a high-order time-stepping scheme for crack propagation (a series of biharmonic problems for an evolving piecewise smooth surface) shows that radically better methods are needed. Special basis functions are too complicated to construct and brute force is not economical – merely storing the discretized solution becomes too costly in a large-scale simulation.

A breakthrough came in 2007, when a scheme was created that resolves virtually any problem for Laplace's equation in piecewise smooth two-dimensional domains in a way that is fully automatic, fast, stable, memory efficient, and whose computational cost scales linearly with the number of corners in the computational domain. The resulting paper [21] constitutes the origin of the RCIP method. Unfortunately, however, there are some flaws in [21]. The expressions in [21, Section 9] are not generally valid and the paper fails to apply RCIP in its entirety to the biharmonic problem of [21, Section 3], which was the ultimate goal.

The second paper on RCIP [22] deals with elastic grains. The part [22, Appendix B], on speedup, is particularly useful.

The third paper on RCIP [14] contains improvement relative to the earlier papers, both in the notation and in the discretization of singular operators. The overall theme is mixed boundary conditions, which pose similar difficulties as do piecewise smooth boundaries.

The paper [15], finally, solves the problem of [21, Section 3] in a broad setting and one can view the formalism of [15] as the RCIP method in its most general form, at least in two dimensions. Further work on RCIP has been devoted to more general boundary conditions [32], to problems in three dimension [23], and to solving elliptic boundary value problems in special situations, such as for aggregates of millions of grains, where special techniques are introduced to deal with problem specific issues [16, 17].

We end this retrospection by pointing out that several research groups in recent years have proposed numerical schemes for integral equations stemming from elliptic partial differential equations in piecewise smooth domains. See, for example, [2, 3, 4, 5, 6, 7]. There is also a widespread notion that a slight rounding of corners is a good idea for numerics. While rounding may work in particular situations, we do not believe it is a generally viable method. For one thing, how does one round a triple junction?

3 A model problem

Let Γ be the closed contour of Figure 1 with the parameterization

$$r(s) = \sin(\pi s)\left(\cos((s-0.5)\theta), \sin((s-0.5)\theta)\right), \quad s \in [0,1]. \tag{1}$$

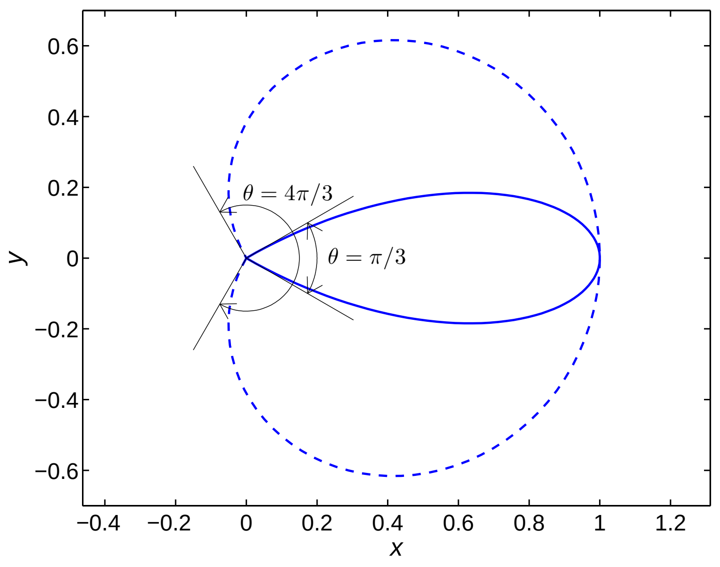

Figure 1: The contour Γ of (1) with a corner at the origin. The solid curve corresponds to opening angle $\theta = \pi/3$. The dashed curve has $\theta = 4\pi/3$.

Let $G(r, r')$ be the fundamental solution to Laplace's equation which in the plane can be written

$$G(r, r') = -\frac{1}{2\pi} \log |r - r'| \,. \tag{2}$$

We shall solve the integral equation

$$\rho(r) + 2\lambda \int_\Gamma \frac{\partial}{\partial \nu_r} G(r, r') \rho(r') \, \mathrm{d}\sigma_{r'} = 2\lambda \left(e \cdot \nu_r\right) \,, \quad r \in \Gamma \,, \tag{3}$$

numerically for the unknown layer density $\rho(r)$. Here ν is the exterior unit normal of the contour, $\mathrm{d}\sigma$ is an element of arc length, λ is a parameter, and e is a unit vector. The equation (3) models an electrostatic inclusion problem [21].

Using complex notation, where vectors r, r', and ν in the real plane $\mathbb{R}^2$ correspond to points z, τ, and n in the complex plane $\mathbb{C}$, one can express (3) as

$$\rho(z) + \frac{\lambda}{\pi} \int_\Gamma \rho(\tau) \Im \left\{ \frac{n_z \bar{n}_\tau \, \mathrm{d}\tau}{\tau - z} \right\} = 2\lambda \Re \left\{ \bar{e} n_z \right\} \,, \quad z \in \Gamma \,, \tag{4}$$

where the "bar" symbol denotes complex conjugation. Equation (4) is a simplification over (3) from a programming point of view.

From an algorithmic point of view it is advantageous to write (3) in the abbreviated form

$$\left(I + \lambda K\right) \rho(z) = \lambda g(z) \,, \quad z \in \Gamma \,, \tag{5}$$

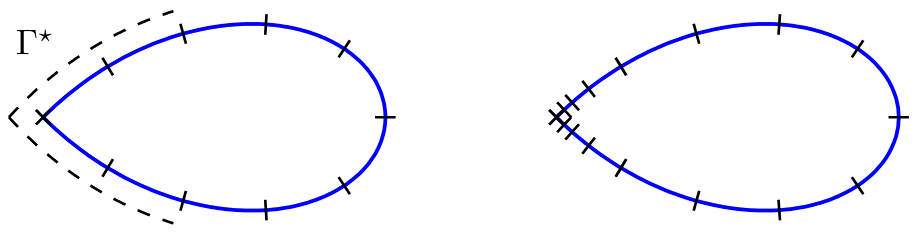

Figure 2: Left: A coarse mesh with ten panels on the contour Γ of (1) with opening angle $\theta = \pi/2$. A subset of Γ, called $\Gamma^\star$, covers four coarse panels as indicated by the dashed curve. Right: A fine mesh created from the coarse mesh by subdividing the panels closest to the corner $n_{\rm sub} = 3$ times.

where I is the identity. If Γ is smooth, then (5) is a Fredholm second kind integral equation with a compact, non-self-adjoint, integral operator K whose spectrum is discrete, bounded by one in modulus, and accumulates at zero.

We also need a way to monitor the convergence of solutions $\rho(z)$ to (5). For this purpose we introduce a quantity q, which corresponds to dipole moment or polarizability [23]

$$q = \int_\Gamma \rho(z) \Re\left\{\bar{e}z\right\} \mathrm{d}|z| \,. \tag{6}$$

Remark: Existence issues are important. Loosely speaking, the boundary value problem modeled by (3) has a unique finite energy solution for a large class of non-smooth Γ when λ is either off the real axis or when λ is real and $\lambda \in [-1,1)$. See [23] for sharper statements. The precise meaning of a *numerical solution* to an integral equation such as (3) also deserves comment. In this paper, a numerical solution refers to approximate values of $\rho(r)$ at a discrete set of points $r_i \in \Gamma$. The values $\rho(r_i)$ should, in a post-processor, enable the extraction of quantities of interest including values of $\rho(r)$ at arbitrary points $r \in \Gamma$, functionals of $\rho(r)$ such as q of (6), and the solution to the underlying boundary value problem at points in the domain where that problem was set.

4 Discretization on two meshes

We discretize (5) using the original Nyström method based on composite 16-point Gauss–Legendre quadrature on two different meshes: a *coarse mesh* with $n_{\rm pan}$ quadrature panels and a *fine mesh* which is constructed from the coarse mesh by $n_{\rm sub}$ times subdividing the panels closest to the corner in a direction toward the corner. The discretization is in parameter. The four panels on the coarse mesh that are closest to the corner should be equisized in parameter. These panels form a subset of Γ called $\Gamma^\star$. See Figure 2.

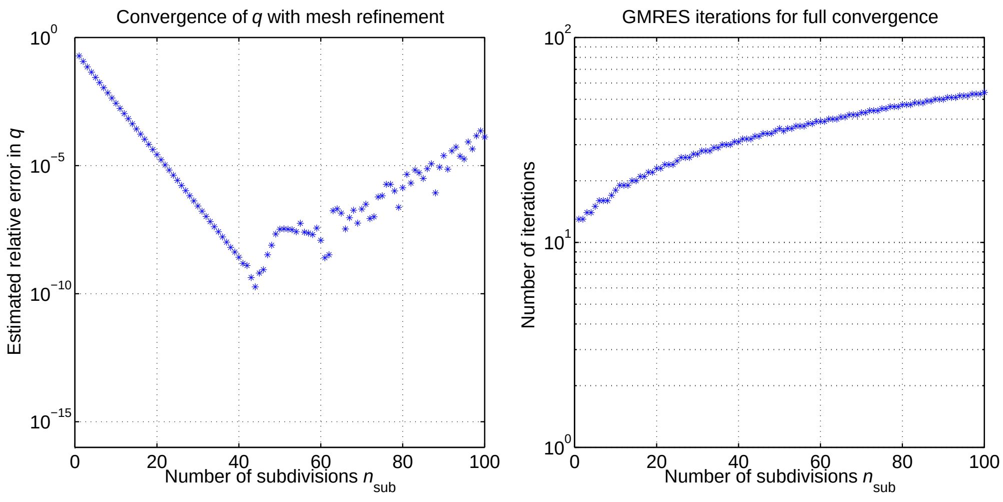

Figure 3: Convergence for q of (6) using (8) and the program `demo1b.m` (a loop of `demo1.m`) with `lambda=0.999`, `theta=pi/2`, `npan=10`, and `evec=1`. The reference value is $q = 1.1300163213105365$. There are $n_{\rm p} = 160 + 32n_{\rm sub}$ unknowns in the main linear system. Left: Convergence with $n_{\rm sub}$. Right: The number of iterations needed to meet an estimated relative residual of $\epsilon_{\rm mach}$.

The linear systems resulting from discretization on the coarse mesh and on the fine mesh can be written formally as

$$(\mathbf{I}_{\rm coa} + \lambda \mathbf{K}_{\rm coa})\,\boldsymbol{\rho}_{\rm coa} = \lambda \mathbf{g}_{\rm coa}\,, \tag{7}$$

$$(\mathbf{I}_{\rm fin} + \lambda \mathbf{K}_{\rm fin})\,\boldsymbol{\rho}_{\rm fin} = \lambda \mathbf{g}_{\rm fin}\,, \tag{8}$$

where $\mathbf{I}$ and $\mathbf{K}$ are square matrices and $\boldsymbol{\rho}$ and $\mathbf{g}$ are column vectors. The subscripts fin and coa indicate what type of mesh is used. Discretization points on a mesh are said to constitute a grid. The coarse grid has $n_{\rm p} = 16n_{\rm pan}$ points. The fine grid has $n_{\rm p} = 16(n_{\rm pan} + 2n_{\rm sub})$ points.

The discretization of (5) is carried out by first rewriting (4) as

$$\rho(z(s)) + \frac{\lambda}{\pi}\int_0^1 \rho(\tau(t))\Re\left\{\frac{n_{z(s)}|\tau'(t)|\,{\rm d}t}{\tau(t)-z(s)}\right\} = 2\lambda\Re\left\{\bar{e}n_{z(s)}\right\}\,,\quad s\in[0,1]\,. \tag{9}$$

Then Nyström discretization with $n_{\rm p}$ points z_i and weights w_i on Γ gives

$$\rho_i + \frac{\lambda}{\pi}\sum_{j=1}^{n_{\rm p}} \rho_j \Re\left\{\frac{n_i|z_j'|w_j}{z_j - z_i}\right\} = 2\lambda\Re\left\{\bar{e}n_i\right\}\,,\quad i = 1,2,\ldots,n_{\rm p}\,. \tag{10}$$

The program `demo1.m` sets up the system (8), solves it using the GMRES iterative solver [33] incorporating a low-threshold stagnation avoiding technique [20, Section 8], and computes q of (6). The user has to specify the opening angle θ, the parameter λ, the number $n_{\rm pan}$ of coarse panels on Γ, the unit vector e and the number of subdivisions n_{sub}. The opening angle should be in the interval $\pi/3 \le \theta \le 5\pi/3$. We choose $\lambda = 0.999$, $\theta = \pi/2$, $n_{\text{pan}} = 10$, and $e = 1$. The quantity q converges initially as n_{sub} is increased, but for $n_{\text{sub}} > 44$ the results start to get worse. See Figure 3. This is related to the fact, pointed out by Bremer [3], that the original Nyström method captures the L^∞ behavior of the solution ρ, while our ρ is unbounded. See, further, Appendix D.

5 Compressed inverse preconditioning

Let us split the matrices $\mathbf{K}_{\text{coa}}$ and $\mathbf{K}_{\text{fin}}$ of (7) and (8) into two parts each

$$\mathbf{K}_{\text{coa}} = \mathbf{K}^{\star}_{\text{coa}} + \mathbf{K}^{\circ}_{\text{coa}} \tag{11}$$

$$\mathbf{K}_{\text{fin}} = \mathbf{K}^{\star}_{\text{fin}} + \mathbf{K}^{\circ}_{\text{fin}}\,. \tag{12}$$

Here the superscript $\star$ indicates that only entries of a matrix K_{ij} whose indices i and j correspond to points z_i and z_j that both belong to the boundary segment $\Gamma^\star$ are retained. The remaining entries are zero.

Now we introduce two diagonal matrices $\mathbf{W}_{\text{coa}}$ and $\mathbf{W}_{\text{fin}}$ which have the quadrature weights w_i on the diagonal. Furthermore, we need a prolongation matrix $\mathbf{P}$ which interpolates functions known at points on the coarse grid to points on the fine grid. The construction of $\mathbf{P}$ relies on panelwise 15-degree polynomial interpolation in parameter using Vandermonde matrices. We also construct a weighted prolongation matrix $\mathbf{P}_W$ via

$$\mathbf{P}_W = \mathbf{W}_{\text{fin}}\mathbf{P}\mathbf{W}^{-1}_{\text{coa}}\,. \tag{13}$$

The matrices $\mathbf{P}$ and $\mathbf{P}_W$ share the same sparsity pattern. They are rectangular matrices, similar to the the identity matrix, but with one full $(4 + 2n_{\text{sub}})16 \times 64$ block. Let superscript T denote the transpose. Then

$$\mathbf{P}^T_W\mathbf{P} = \mathbf{I}_{\text{coa}} \tag{14}$$

holds exactly. See Appendix A and [15, Section 4.3].

Equipped with $\mathbf{P}$ and $\mathbf{P}_W$ we are ready to compress (8) on the fine grid to an equation essentially on the coarse grid. This compression is done without the loss of accuracy – the discretization error in the solution is unaffected and no information is lost. The compression relies on the variable substitution

$$\left(\mathbf{I}_{\text{fin}} + \lambda\mathbf{K}^{\star}_{\text{fin}}\right)\boldsymbol{\rho}_{\text{fin}} = \mathbf{P}\tilde{\boldsymbol{\rho}}_{\text{coa}}\,. \tag{15}$$

Here $\tilde{\boldsymbol{\rho}}_{\text{coa}}$ is the discretization of a piecewise smooth *transformed density.* The compression also uses the low-rank decomposition

$$\mathbf{K}^{\circ}_{\text{fin}} = \mathbf{P}\mathbf{K}^{\circ}_{\text{coa}}\mathbf{P}^T_W\,, \tag{16}$$

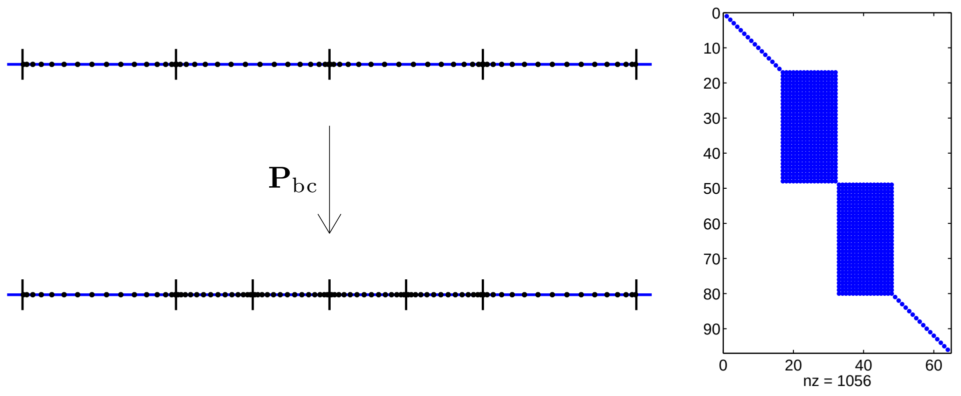

Figure 4: Left: The prolongation operator $\mathbf{P}_{\rm bc}$ performs panelwise interpolation from a grid on a four-panel mesh to a grid on a six-panel mesh. Right: The sparsity pattern of $\mathbf{P}_{\rm bc}$.

which should hold to about machine precision.

The compressed version of (8) reads

$$\left(\mathbf{I}_{\rm coa} + \lambda \mathbf{K}^{\circ}_{\rm coa}\mathbf{R}\right)\tilde{\boldsymbol{\rho}}_{\rm coa} = \lambda \mathbf{g}_{\rm coa}\,, \tag{17}$$

where the compressed weighted inverse $\mathbf{R}$ is given by

$$\mathbf{R} = \mathbf{P}_W^T\left(\mathbf{I}_{\rm fin} + \lambda \mathbf{K}^{\star}_{\rm fin}\right)^{-1}\mathbf{P}\,. \tag{18}$$

See Appendix B for details on the derivation. The compressed weighted inverse $\mathbf{R}$, for Γ of (1), is a block diagonal matrix with one full 64×64 block and the remaining entries coinciding with those of the identity matrix.

After having solved (17) for $\tilde{\boldsymbol{\rho}}_{\rm coa}$, the density $\boldsymbol{\rho}_{\rm fin}$ can easily be reconstructed from $\tilde{\boldsymbol{\rho}}_{\rm coa}$ in a post-processor, see Section 9. It is important to observe, however, that $\boldsymbol{\rho}_{\rm fin}$ is not always needed. For example, the quantity q of (6) can be computed directly from $\tilde{\boldsymbol{\rho}}_{\rm coa}$. Let $\boldsymbol{\zeta}_{\rm coa}$ be a column vector which contains the absolute values of the boundary derivatives z_i' multiplied with weights w_i, positions z_i, and the complex scalar $\bar{e}$. Then

$$q = \Re\left\{\boldsymbol{\zeta}_{\rm coa}\right\}^T \mathbf{R}\tilde{\boldsymbol{\rho}}_{\rm coa}\,. \tag{19}$$

6 The recursion for R

The compressed weighted inverse $\mathbf{R}$ is costly to compute from its definition (18). As we saw in Section 4, the inversion of large matrices $(\mathbf{I}+\mathbf{K})$ on highly refined grids could also be unstable. Fortunately, the computation of $\mathbf{R}$ can be greatly sped up and stabilized via a recursion. This recursion is derived in a roundabout way and using a refined grid that differs from that of the present tutorial in [21, Section 7.2]. A better derivation can be found in [15, Section 5], but there the setting is more general so that text could be hard to follow. Here we focus on results.

6.1 Basic prolongation matrices

Let $\mathbf{P}_{\text{bc}}$ be a prolongation matrix, performing panelwise 15-degree polynomial interpolation in parameter from a 64-point grid on a four-panel mesh to a 96-point grid on a six-panel mesh as shown in Figure 4. Let $\mathbf{P}_{W\text{bc}}$ be a weighted prolongation matrix in the style of (13). If `T16` and `W16` are the nodes and weights of 16-point Gauss–Legendre quadrature on the canonical interval $[-1, 1]$, then $\mathbf{P}_{\text{bc}}$ and $\mathbf{P}_{W\text{bc}}$ can be constructed as

```
  T32=[T16-1;T16+1]/2;
  W32=[W16;W16]/2;
  A=ones(16);
  AA=ones(32,16);
  for k=2:16
    A(:,k)=A(:,k-1).*T16;
    AA(:,k)=AA(:,k-1).*T32;
  end
  IP=AA/A;
  IPW=IP.*(W32*(1./W16)');
%
  Pbc=zeros(96,64);
  Pbc( 1:16, 1:16)=eye(16);
  Pbc(17:48,17:32)=IP;
  Pbc(49:96,33:64)=flipud(fliplr(Pbc(1:48,1:32)));
%
  PWbc=zeros(96,64);
  PWbc( 1:16, 1:16)=eye(16);
  PWbc(17:48,17:32)=IPW;
  PWbc(49:96,33:64)=flipud(fliplr(PWbc(1:48,1:32)));
```

See [20, Appendix A] for an explanation of why high-degree polynomial interpolation involving ill-conditioned Vandermonde systems gives accurate results for smooth functions.

6.2 Discretization on nested meshes

Let $\Gamma^\star_i$, $i = 1, 2, \ldots, n_{\text{sub}}$, be a sequence of subsets of $\Gamma^\star$ with $\Gamma^\star_{i-1} \subset \Gamma^\star_i$ and $\Gamma^\star_{n_{\text{sub}}} = \Gamma^\star$. Let there also be a six-panel mesh and a corresponding 96-point grid on each $\Gamma^\star_i$. The construction of the subsets and their meshes should be such that if $z(s)$, $s \in [-2, 2]$, is a local parameterization of $\Gamma^\star_i$, then the breakpoints (locations of panel endpoints) of its mesh are at $s \in$

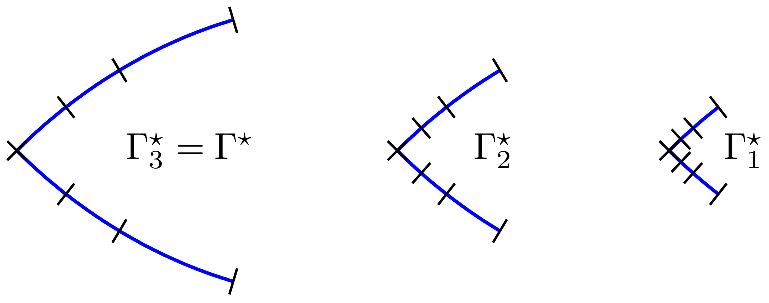

Figure 5: The boundary subsets $\Gamma_3^\star$, $\Gamma_2^\star$, and $\Gamma_1^\star$ along with their corresponding type b meshes for $n_{\rm sub} = 3$.

$\{-2, -1, -0.5, 0, 0.5, 1, 2\}$ and the breakpoints of the mesh on $\Gamma_{i-1}^\star$ are at $s = \{-1, -0.5, -0.25, 0, -0.25, 0.5, 1\}$. We denote this type of nested six-panel meshes *type b*. The index i is the *level*. An example of a sequence of subsets and meshes on $\Gamma^\star$ is shown in Figure 5 for $n_{\rm sub} = 3$. Compare [14, Figure 2] and [15, Figure 5.1].

Let $\mathbf{K}_{i\rm b}$ denote the discretization of K on a type b mesh on $\Gamma_i^\star$. In the spirit of (11,12) we write

$$\mathbf{K}_{i\rm b} = \mathbf{K}_{i\rm b}^\star + \mathbf{K}_{i\rm b}^\circ \,, \tag{20}$$

where the superscript $\star$ indicates that only entries with both indices corresponding to points on the four inner panels are retained.

6.3 The recursion proper

Now, let $\mathbf{R}_{n_{\rm sub}}$ denote the full 64×64 diagonal block of $\mathbf{R}$. The recursion for $\mathbf{R}_{n_{\rm sub}}$ is derived in Appendix C and it reads

$$\mathbf{R}_i = \mathbf{P}_{W\rm bc}^T \left(\mathbb{F}\{\mathbf{R}_{i-1}^{-1}\} + \mathbf{I}_{\rm b}^\circ + \lambda \mathbf{K}_{i\rm b}^\circ \right)^{-1} \mathbf{P}_{\rm bc} \,, \quad i = 1, \ldots, n_{\rm sub} \,, \tag{21}$$

$$\mathbb{F}\{\mathbf{R}_0^{-1}\} = \mathbf{I}_{\rm b}^\star + \lambda \mathbf{K}_{1\rm b}^\star \,, \tag{22}$$

where the operator $\mathbb{F}\{\cdot\}$ expands its matrix argument by zero-padding (adding a frame of zeros of width 16 around it). Note that the initializer (22) makes the recursion (21) take the first step

$$\mathbf{R}_1 = \mathbf{P}_{W\rm bc}^T \left(\mathbf{I}_{\rm b} + \lambda \mathbf{K}_{1\rm b} \right)^{-1} \mathbf{P}_{\rm bc} \,.$$

The program `demo2.m` sets up the linear system (17), runs the recursion (21,22), and solves the linear system using the same techniques as `demo1.m`, see Section 4. In fact, the results produced by the two programs are very similar, at least up to $n_{\rm sub} = 40$. This supports the claim of Section 5 that the discretization error in the solution is unaffected by compression.

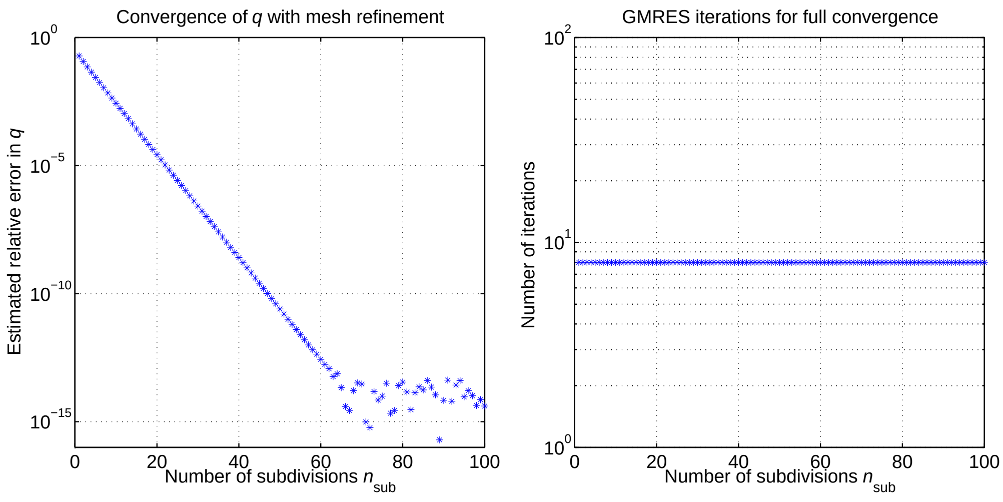

Figure 6: Same as Figure 3, but using (17) and the program `demo2b.m` (a loop of `demo2.m`). There are only $n_{\rm p} = 160$ unknowns in the main linear system.

Figure 6 demonstrates the power of RCIP: fewer unknowns and faster execution, better conditioning (the number of GMRES iterations does not grow), and higher achievable accuracy. Compare Figure 3. We emphasize that the number $n_{\rm sub}$ of recursion steps (levels) used in (21) corresponds to the number of subdivisions $n_{\rm sub}$ used to construct the fine mesh.

7 Schur–Banachiewicz speedup of the recursion

The recursion (21) can be sped up using the Schur–Banachiewicz inverse formula for partitioned matrices [25], which in this context can be written [22, Appendix B]

$$\begin{bmatrix} \mathbf{P}_W^{\star T} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A}^{-1} & \mathbf{U} \\ \mathbf{V} & \mathbf{D} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{P}^\star & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} =$$

$$\begin{bmatrix} \mathbf{P}_W^{\star T}\mathbf{A}\mathbf{P}^\star + \mathbf{P}_W^{\star T}\mathbf{A}\mathbf{U}(\mathbf{D}-\mathbf{V}\mathbf{A}\mathbf{U})^{-1}\mathbf{V}\mathbf{A}\mathbf{P}^\star & -\mathbf{P}_W^{\star T}\mathbf{A}\mathbf{U}(\mathbf{D}-\mathbf{V}\mathbf{A}\mathbf{U})^{-1} \\ -(\mathbf{D}-\mathbf{V}\mathbf{A}\mathbf{U})^{-1}\mathbf{V}\mathbf{A}\mathbf{P}^\star & (\mathbf{D}-\mathbf{V}\mathbf{A}\mathbf{U})^{-1} \end{bmatrix}, \tag{23}$$

where $\mathbf{A}$ plays the role of $\mathbf{R}_{i-1}$, $\mathbf{P}^\star$ and $\mathbf{P}_W^\star$ are submatrices of $\mathbf{P}_{\rm bc}$ and $\mathbf{P}_{W{\rm bc}}$, and $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{D}$ refer to blocks of $\lambda\mathbf{K}_{i{\rm b}}^\circ$.

The program `demo3.m` is based on `demo2.m`, but has (23) incorporated. Besides, the integral equation (3) is replaced with

$$\rho(r)+2\lambda\int_\Gamma \frac{\partial}{\partial\nu_r}G(r,r')\rho(r')\,{\rm d}\sigma_{r'}+\int_\Gamma \rho(r')\,{\rm d}\sigma_{r'} = 2\lambda\,(e\cdot\nu_r)\;, \quad r\in\Gamma\,, \tag{24}$$

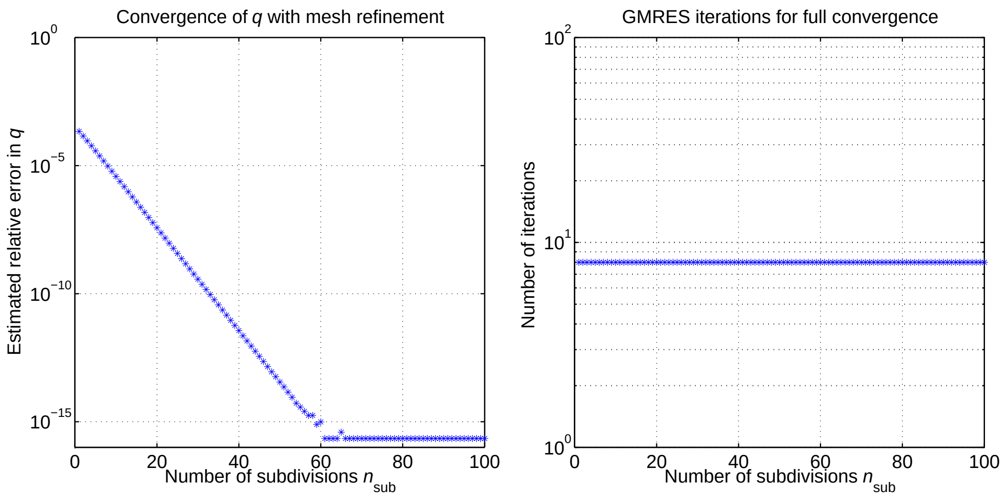

Figure 7: Same as Figure 6, but the program `demo3b.m` is used.

which has the same solution $\rho(r)$ but is more stable for λ close to one. For the discretization of (24) to fit the form (17), the last term on the left hand side of (24) is added to the matrix $\lambda\mathbf{K}^{\circ}_{\text{coa}}$ of (17).

The execution of `demo3.m` is faster than that of `demo2.m`. Figure 7 shows that a couple of extra digits are gained by using (24) rather than (3) and that full machine accuracy is achieved for $n_{\text{sub}} > 60$.

8 Various useful quantities

Let us introduce a new discrete density $\hat{\boldsymbol{\rho}}_{\text{coa}}$ via

$$\hat{\boldsymbol{\rho}}_{\text{coa}} = \mathbf{R}\tilde{\boldsymbol{\rho}}_{\text{coa}}. \tag{25}$$

Rewriting (17) in terms of $\hat{\boldsymbol{\rho}}_{\text{coa}}$ gives

$$\left(\mathbf{R}^{-1} + \lambda\mathbf{K}^{\circ}_{\text{coa}}\right)\hat{\boldsymbol{\rho}}_{\text{coa}} = \lambda\mathbf{g}_{\text{coa}}\,, \tag{26}$$

which resembles the original equation (7). We see that $\mathbf{K}^{\circ}_{\text{coa}}$, which is discretized using Gauss–Legendre quadrature, acts on $\hat{\boldsymbol{\rho}}_{\text{coa}}$. Therefore one can interpret $\hat{\boldsymbol{\rho}}_{\text{coa}}$ as pointwise values of the original density $\rho(r)$, multiplied with weight corrections suitable for integration against polynomials. We refer to $\hat{\boldsymbol{\rho}}_{\text{coa}}$ as a *weight-corrected density*. Compare [22, Section 5.4].

Assume now that there is a square matrix $\mathbf{S}$ which maps $\tilde{\boldsymbol{\rho}}_{\text{coa}}$ to discrete values $\boldsymbol{\rho}_{\text{coa}}$ of the original density on the coarse grid

$$\boldsymbol{\rho}_{\text{coa}} = \mathbf{S}\tilde{\boldsymbol{\rho}}_{\text{coa}}\,. \tag{27}$$

The matrix $\mathbf{S}$ allows us to rewrite (17) as a system for the original density

$$\left(\mathbf{S}^{-1} + \lambda\mathbf{K}^{\circ}_{\text{coa}}\mathbf{R}\mathbf{S}^{-1}\right)\boldsymbol{\rho}_{\text{coa}} = \lambda\mathbf{g}_{\text{coa}}\,. \tag{28}$$

We can interpret the composition $\mathbf{R}\mathbf{S}^{-1}$ as a matrix of multiplicative weight corrections that compensate for the singular behavior of $\rho(r)$ on $\Gamma^\star$ when Gauss–Legendre quadrature is used.

Let $\mathbf{Y}$ denote the rectangular matrix

$$\mathbf{Y} = \left(\mathbf{I}_{\text{fin}} + \lambda \mathbf{K}^\star_{\text{fin}}\right)^{-1} \mathbf{P} \,, \tag{29}$$

and let $\mathbf{Q}$ be a restriction operator which performs panelwise 15-degree polynomial interpolation in parameter from a grid on the fine mesh to a grid on a the coarse mesh. We see from (15) that $\mathbf{Y}$ is the mapping from $\tilde{\boldsymbol{\rho}}_{\text{coa}}$ to $\boldsymbol{\rho}_{\text{fin}}$. Therefore the columns of $\mathbf{Y}$ can be interpreted as discrete basis functions for $\rho(r)$. It holds by definition that

$$\mathbf{Q}\mathbf{P} = \mathbf{I}_{\text{coa}} \,, \tag{30}$$

$$\mathbf{Q}\mathbf{Y} = \mathbf{S} \,. \tag{31}$$

The quantities and interpretations of this section come in handy in various situations, for example in 3D extensions of the RCIP method [23]. An efficient scheme for constructing $\mathbf{S}$ will be presented in Section 10.

9 Reconstruction of $\boldsymbol{\rho}_{\text{fin}}$ from $\tilde{\boldsymbol{\rho}}_{\text{coa}}$

The action of $\mathbf{Y}$ on $\tilde{\boldsymbol{\rho}}_{\text{coa}}$, which gives $\boldsymbol{\rho}_{\text{fin}}$, can be obtained by, in a sense, running the recursion (21) backwards. The process is described in detail in [14, Section 7]. Here we focus on results.

The backward recursion on $\Gamma^\star$ reads

$$\vec{\boldsymbol{\rho}}_{\text{coa},i} = \left[\mathbf{I}_{\text{b}} - \lambda \mathbf{K}^\circ_{i\text{b}} \left(\mathbb{F}\{\mathbf{R}^{-1}_{i-1}\} + \mathbf{I}^\circ_{\text{b}} + \lambda \mathbf{K}^\circ_{i\text{b}}\right)^{-1}\right] \mathbf{P}_{\text{bc}} \tilde{\boldsymbol{\rho}}_{\text{coa},i} \,, \quad i = n_{\text{sub}}, \ldots, 1 \,. \tag{32}$$

Here $\tilde{\boldsymbol{\rho}}_{\text{coa},i}$ is a column vector with 64 elements. In particular, $\tilde{\boldsymbol{\rho}}_{\text{coa},n_{\text{sub}}}$ is the restriction of $\tilde{\boldsymbol{\rho}}_{\text{coa}}$ to $\Gamma^\star$, while $\tilde{\boldsymbol{\rho}}_{\text{coa},i}$ are taken as elements $\{17 : 80\}$ of $\vec{\boldsymbol{\rho}}_{\text{coa},i+1}$ for $i < n_{\text{sub}}$. The elements $\{1 : 16\}$ and $\{81 : 96\}$ of $\vec{\boldsymbol{\rho}}_{\text{coa},i}$ are the reconstructed values of $\boldsymbol{\rho}_{\text{fin}}$ on the outermost panels of a type b mesh on $\Gamma^\star_i$. Outside of $\Gamma^\star$, $\boldsymbol{\rho}_{\text{fin}}$ coincides with $\tilde{\boldsymbol{\rho}}_{\text{coa}}$.

When the recursion is completed, the reconstructed values of $\boldsymbol{\rho}_{\text{fin}}$ on the four innermost panels are obtained from

$$\mathbf{R}_0 \tilde{\boldsymbol{\rho}}_{\text{coa},0} \,. \tag{33}$$

Should one wish to interrupt the recursion (32) prematurely, at step $i = j$ say, then

$$\mathbf{R}_{j-1} \tilde{\boldsymbol{\rho}}_{\text{coa},(j-1)} \tag{34}$$

gives values of a weight-corrected density on the four innermost panels of a type b mesh on $\Gamma^\star_j$. That is, we have a part-way reconstructed weight-corrected density $\hat{\boldsymbol{\rho}}_{\text{part}}$ on a mesh that is $n_{\text{sub}} - j + 1$ times refined. This

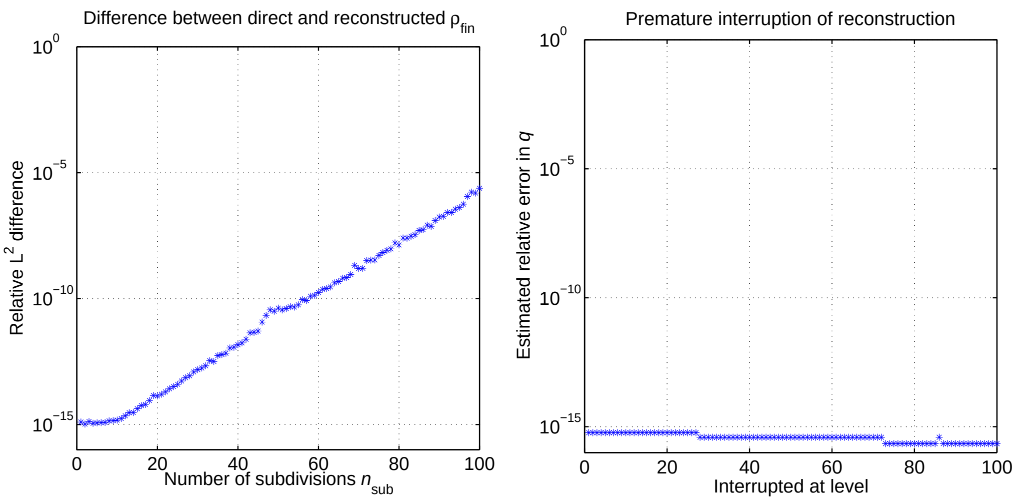

Figure 8: Output from `demo4.m` and `demo5.m`. Left: A comparison of $\boldsymbol{\rho}_{\rm fin}$ from the unstable equation (8) and $\boldsymbol{\rho}_{\rm fin}$ reconstructed from $\tilde{\boldsymbol{\rho}}_{\rm coa}$ of (17) via (32). Right: Relative accuracy in q of (6) from part-way reconstructed solutions $\hat{\boldsymbol{\rho}}_{\rm part}$.

observation is useful in the context of evaluating layer potentials close to their sources.

If the memory permits, one can store the matrices $\mathbf{K}_{ib}^{\circ}$ and $\mathbf{R}_i$ in the forward recursion (21) and reuse them in the backward recursion (32). Otherwise they may be computed afresh.

The program `demo4.m` builds on the program `demo3.m`, using (17) for (24). After the main linear system is solved for $\tilde{\boldsymbol{\rho}}_{\rm coa}$, a postprocessor reconstructs $\boldsymbol{\rho}_{\rm fin}$ via (32). Then a comparison is made with a solution $\boldsymbol{\rho}_{\rm fin}$ obtained by solving the un-compressed system (8). Figure 8 shows that for $n_{\rm sub} < 10$ the results are virtually identical. This verifies the correctness of (32). For $n_{\rm sub} > 10$ the result start to deviate. That illustrates the instabilities associated with solving (8) on a highly refined mesh. Compare Figure 3.

The program `demo5.m` investigates the effects of premature interruption of (32). The number of recursion steps is set to $n_{\rm sub} = 100$ and the recursion is interrupted at different levels. The density $\boldsymbol{\rho}_{\rm fin}$ is reconstructed on outer panels up to the level of interruption. Then a weight-corrected density is produced at the innermost four panels according to (34). Finally q of (6) is computed from this part-way reconstructed solution. The right image of Figure 8 shows that the quality of q is unaffected by the level of interruption.

10 The construction of S

This section discusses the construction of $\mathbf{S}$ and other auxiliary matrices. Note that in many applications, these matrices are not needed.

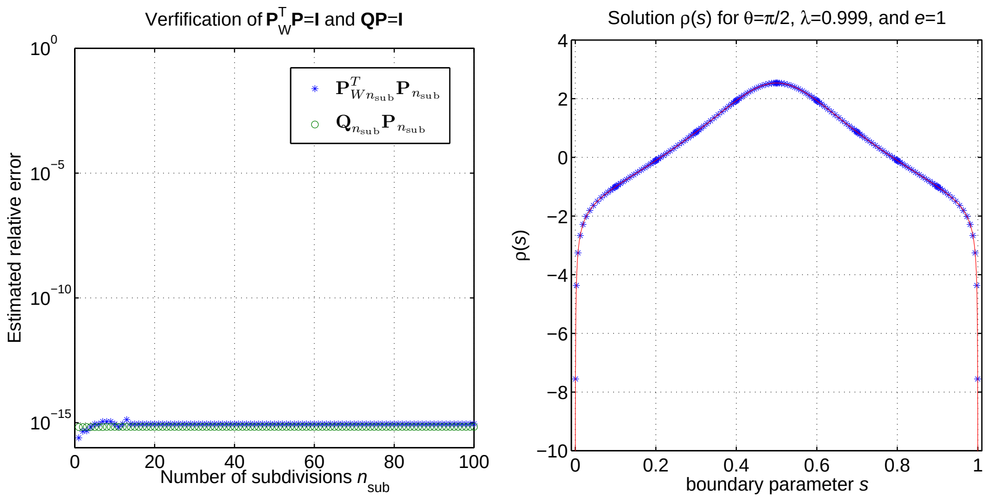

Figure 9: Left: The identities (14) and (30) hold to high accuracy in our implementation, irrespective of the degree of mesh refinement. Right: The solution ρ to (24) on (1) with parameters as specified in Section 4. The solution with RCIP (17) and (27), shown as blue stars, agrees with the solution from (8), shown as a red solid line. The solution diverges in the corner.

The entries of the matrices $\mathbf{P}$, $\mathbf{P}_W$, $\mathbf{Q}$, $\mathbf{R}$, $\mathbf{S}$, and $\mathbf{Y}$ can only differ from those of the identity matrix when both indices correspond to discretization points on $\Gamma^\star$. For example, the entries of $\mathbf{R}$ only differ from the identity matrix for the 64×64 block denoted $\mathbf{R}_{n_{\text{sub}}}$ in (21). In accordance with this notation we introduce $\mathbf{P}_{n_{\text{sub}}}$, $\mathbf{P}_{Wn_{\text{sub}}}$, $\mathbf{Q}_{n_{\text{sub}}}$, $\mathbf{S}_{n_{\text{sub}}}$ and $\mathbf{Y}_{n_{\text{sub}}}$ for the restriction of $\mathbf{P}$, $\mathbf{P}_W$, $\mathbf{Q}$, $\mathbf{S}$ and $\mathbf{Y}$ to $\Gamma^\star$. In the codes of this section we often use this restricted type of matrices, leaving the identity part out.

We observe that $\mathbf{S}_{n_{\text{sub}}}$ is a square 64×64 matrix; $\mathbf{P}_{n_{\text{sub}}}$, $\mathbf{P}_{Wn_{\text{sub}}}$ and $\mathbf{Y}_{n_{\text{sub}}}$ are rectangular $16(4+2n_{\text{sub}}) \times 64$ matrices; and $\mathbf{Q}_{n_{\text{sub}}}$ is a rectangular $64 \times 16(4+2n_{\text{sub}})$ matrix. Furthermore, $\mathbf{Q}_{n_{\text{sub}}}$ is very sparse for large n_{sub}. All columns of $\mathbf{Q}_{n_{\text{sub}}}$ with column indices corresponding to points on panels that result from more than eight subdivisions are identically zero.

The program `demo6.m` sets up $\mathbf{P}_{n_{\text{sub}}}$, $\mathbf{P}_{Wn_{\text{sub}}}$ and $\mathbf{Q}_{n_{\text{sub}}}$, shows their sparsity patterns, and verifies the identities (14) and (30). The implementations for $\mathbf{P}_{n_{\text{sub}}}$ and $\mathbf{P}_{Wn_{\text{sub}}}$ rely on repeated interpolation from coarser to finer intermediate grids. The implementation of $\mathbf{Q}_{n_{\text{sub}}}$ relies on keeping track of the relation between points on the original coarse and fine grids. Output from `demo6.m` is depicted in the left image of Figure 9. Note that the matrices $\mathbf{P}_{n_{\text{sub}}}$ and $\mathbf{P}_{Wn_{\text{sub}}}$ are never needed in applications.

We are now ready to construct $\mathbf{S}$. Section 9 presented a scheme for evaluating the action of $\mathbf{Y}_{n_{\text{sub}}}$ on discrete functions on the coarse grid on $\Gamma^\star$. The matrix $\mathbf{Y}_{n_{\text{sub}}}$, itself, can be constructed by applying this scheme to a 64×64 identity matrix. The matrix $\mathbf{Q}_{n_{\text{sub}}}$ was set up in `demo6.m`.

Composing these two matrices gives $\mathbf{S}_{n_{\rm sub}}$, see (31). This is done in the program `demo7.m`, where the identity part is added as to get the entire matrix $\mathbf{S}$. In previous work on RCIP we have found use for $\mathbf{S}$ in complex situations where (28) is preferable over (17), see [23, Section 9]. If one merely needs $\boldsymbol{\rho}_{\rm coa}$ from $\tilde{\boldsymbol{\rho}}_{\rm coa}$ in a post-processor, setting up $\mathbf{S}$ and using (27) is not worthwhile. It is cheaper to let $\mathbf{Y}$ act on $\tilde{\boldsymbol{\rho}}_{\rm coa}$ and then let $\mathbf{Q}$ act on the resulting vector. Anyhow, `demo7.m` builds on `demo4.m` and gives as output $\boldsymbol{\rho}_{\rm coa}$ computed via (27), see the right image of Figure 9. For comparison, $\boldsymbol{\rho}_{\rm fin}$, computed from the un-compressed system (8), is also shown.

11 Initiating R using fixed-point iteration

It often happens that $\Gamma_i^\star$ is wedge-like. A corner of a polygon, for example, has wedge-like $\Gamma_i^\star$ at all levels. If $\Gamma^\star$ is merely piecewise smooth, then the $\Gamma_i^\star$ are wedge-like to double precision accuracy for $i \ll n_{\rm sub}$.

Wedge-like sequences of $\Gamma_i^\star$ open up for simplifications and speedup in the recursion (21,22). Particularly so if the kernel of the integral operator K of (5) is scale invariant on wedges. Then the matrix $\mathbf{K}_{i{\rm b}}^\circ$ becomes independent of i. It can be denoted $\mathbf{K}_{\rm b}^\circ$ and needs only to be constructed once. Furthermore, the recursion (21,22) assumes the form of a fixed-point iteration

$$\mathbf{R}_i = \mathbf{P}_{W{\rm bc}}^T \left(\mathbb{F}\{\mathbf{R}_{i-1}^{-1}\} + \mathbf{I}_{\rm b}^\circ + \lambda \mathbf{K}_{\rm b}^\circ\right)^{-1} \mathbf{P}_{\rm bc}\,, \quad i = 1, \dots \tag{35}$$

$$\mathbb{F}\{\mathbf{R}_0^{-1}\} = \mathbf{I}_{\rm b}^\star + \lambda \mathbf{K}_{\rm b}^\star\,. \tag{36}$$

The iteration (35) can be run until $\mathbf{R}_i$ converges. Let $\mathbf{R}_*$ be the converged result. One need not worry about predicting the number of levels needed. Choosing the number $n_{\rm sub}$ of levels needed, in order to meet a beforehand given tolerance in $\mathbf{R}_*$, is otherwise a big problem in connection with (21,22) and non-wedge-like $\Gamma^\star$. This number has no general upper bound.

Assume now that the kernel of K is scale invariant on wedges. If all $\Gamma_i^\star$ are wedge-like, then (35,36) replaces (21,22) entirely. If $\Gamma^\star$ is merely piecewise smooth, then (35,36) can be run on a wedge with the same opening angle as $\Gamma^\star$, to produce an initializer to (21). That initializer could often be more appropriate than (22), which is plagued with a very large discretization error whenever (10) is used. This fixed-point recursion initializer is implemented in the program `demo8b.m`, which is a simple upgrading of `demo3b.m`, and produces Figure 10. A comparison of Figure 10 with Figure 7 shows that the number $n_{\rm sub}$ of levels needed for full convergence is halved compared to when using the initializer (22).

There are, generally speaking, several advantages with using the fixed-point recursion initializer, rather than (22), in (21) on a non-wedge-like $\Gamma^\star$: First, the number of different matrices $\mathbf{R}_i$ and $\mathbf{K}_{i{\rm b}}^\circ$ needed in (21) and in (32) is reduced as the recursions are shortened. This means savings in storage.

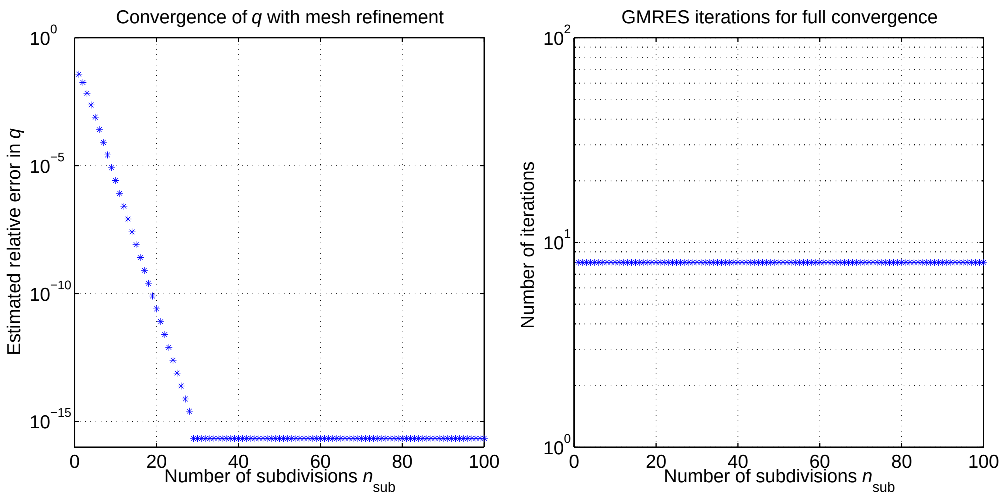

Figure 10: Same as Figure 7, but the program `demo8b.m` is used.

Second, the number n_{sub} of levels needed for full convergence in (21) seems to always be bounded. The hard work is done in (35). Third, Newton's method can be used to accelerate (35). That is the topic of Section 12.

12 Newton acceleration

When solving integral equations stemming from particularly challenging elliptic boundary value problems with solutions $\rho(r)$ that are barely absolutely integrable, the fixed-point iteration (35,36) on wedge-like $\Gamma^\star$ may need a very large number of steps to reach full convergence. See [17, Section 6.3] for an example where $2 \cdot 10^5$ steps are needed.

Fortunately, (35) can be cast as a non-linear matrix equation

$$\mathbf{G}(\mathbf{R}_*) \equiv \mathbf{P}^T_{W\text{bc}}\mathbf{A}(\mathbf{R}_*)\mathbf{P}_{\text{bc}} - \mathbf{R}_* = 0\,, \tag{37}$$

where $\mathbf{R}_*$, as in Section 11, is the fixed-point solution and

$$\mathbf{A}(\mathbf{R}_*) = \left(\mathbb{F}\{\mathbf{R}_*^{-1}\} + \mathbf{I}_{\text{b}}^\circ + \lambda\mathbf{K}_{\text{b}}^\circ\right)^{-1}\,. \tag{38}$$

The non-linear equation (37), in turn, can be solved for $\mathbf{R}_*$ with a variant of Newton's method. Let $\mathbf{X}$ be a matrix-valued perturbation of $\mathbf{R}_*$ and expand $\mathbf{G}(\mathbf{R}_* + \mathbf{X}) = 0$ to first order in $\mathbf{X}$. This gives a Sylvester-type matrix equation

$$\mathbf{X} - \mathbf{P}^T_{W\text{bc}}\mathbf{A}(\mathbf{R}_*)\mathbb{F}\{\mathbf{R}_*^{-1}\mathbf{X}\mathbf{R}_*^{-1}\}\mathbf{A}(\mathbf{R}_*)\mathbf{P}_{\text{bc}} = \mathbf{G}(\mathbf{R}_*) \tag{39}$$

for the Newton update $\mathbf{X}$. One can use the MATLAB built-in function `dlyap` for (39), but GMRES seems to be more efficient and we use that method. Compare [17, Section 6.2].

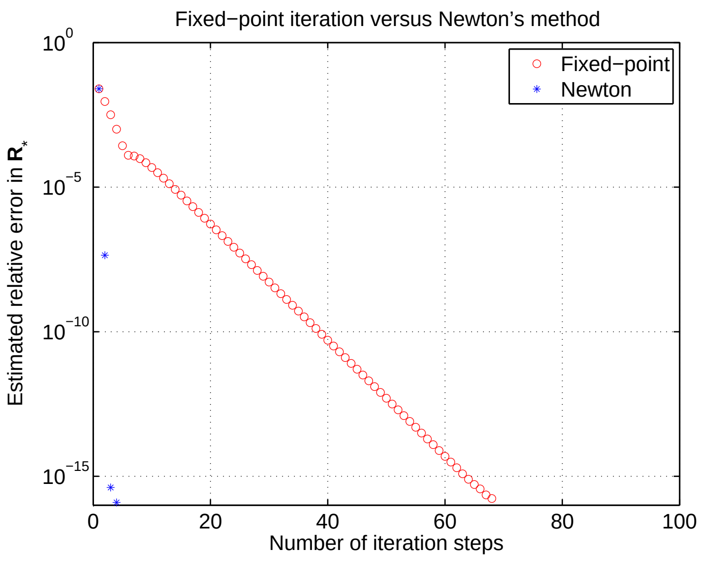

Figure 11: Output from the program `demo9.m`. The fixed-point iteration (35,36) is compared to Newton's method for (37).

Figure 11 shows a comparison between the fixed-point iteration (35,36) and Newton's method for computing the fixed-point solution $\mathbf{R}_*$ to (37) on a wedge-like $\Gamma^\star$. The program `demo9.m` is used and it incorporates Schur–Banachiewicz speedup in the style of Section 7. The wedge opening angle is $\theta = \pi/2$, The integral operator K is the same as in (5), and $\lambda = 0.999$. The relative difference between the two converged solutions $\mathbf{R}_*$ is $5.6 \cdot 10^{-16}$. Figure 11 clearly shows that (35,36) converges linearly (in 68 iterations), while Newton's method has quadratic convergence. Only four iterations are needed. The computational cost per iteration is, of course, higher for Newton's method than for the fixed-point iteration, but it is the same at each step. Recall that the underlying size of the matrix $(\mathbf{I}_{\text{fin}} + \lambda \mathbf{K}^\star_{\text{fin}})$, that is inverted according to (18), grows linearly with the number of steps needed in the fixed-point iteration. This example therefore demonstrates that one can invert and compress a linear system of the type (18) in sub-linear time.

13 On the accuracy of "the solution"

The integral equation (3) comes from a boundary value problem for Laplace's equation where the potential field $U(r)$ at a point r in the plane is related to the layer density $\rho(r)$ via

$$U(r) = (e \cdot r) - \int_\Gamma G(r, r')\rho(r')\, \mathrm{d}\sigma_{r'}\,, \tag{40}$$

see [21, Section 2.1].

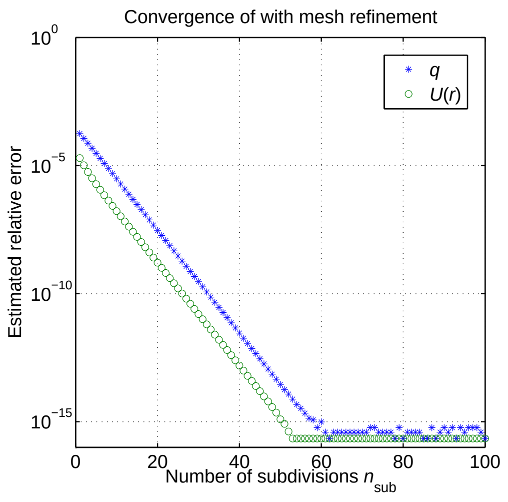

Figure 12: Similar as Figure 7, left image, but with the program `demo3c.m`. The potential $U(r)$ at $r = (0.4, 0.1)$ is evaluated via (40) and `npan=14` is used.

Figure 12 shows how the field solution $U(r)$ of (40) converges with mesh refinement at a point $r = (0.4, 0.1)$ inside the contour Γ of (1). We see that the accuracy in $U(r)$, typically, is one digit better than the accuracy of the dipole moment q of (6). One can therefore say that measuring the field error at a point some distance away from the corner is more forgiving than measuring the dipole moment error. It is possible to construct examples where the difference in accuracy between field solutions and moments of layer densities are more pronounced and this raises the question of how accuracy best should be measured in the context of solving integral equations. We do not have a definite answer, but think that higher moments of layer densities give more fair estimates of overall accuracy than do field solutions at points some distance away from the boundary. See, further, Appendix E.

14 Composed integral operators

Assume that we have a modification of (5) which reads

$$(I + MK)\,\rho_1(z) = g(z)\,, \quad z \in \Gamma\,. \tag{41}$$

Here K and g are as in (5), M is a new, bounded, integral operator, and ρ_1 is an unknown layer density to be solved for. This section shows how to apply RCIP to (41) using a simplified version of the scheme in [15].

Let us, temporarily, expand (41) into a system of equations by introduc-

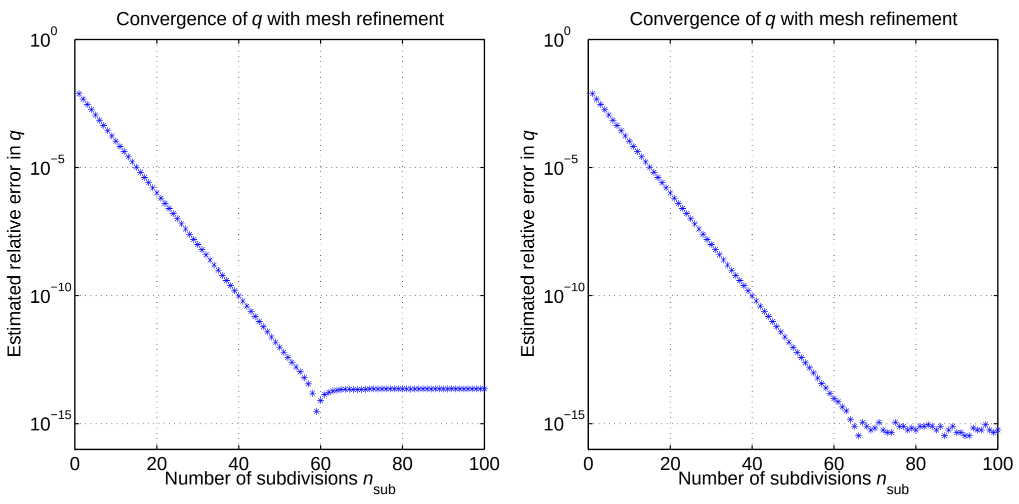

Figure 13: Convergence for q of (6) with $\rho = \rho_1$ from (41). The curve Γ is as in (1) and `theta=pi/2`, `npan=11`, and `evec=1`. The reference values is taken as $q = 1.95243329423584$. Left: Results from the inner product preserving scheme of Appendix D produced with `demo10.m`. Right: Results with RCIP according to (48,49) produced with `demo10b.m`

.

ing a new layer density $\rho_2(z) = K\rho_1(z)$. Then

$$\begin{aligned} \rho_1(z) + M\rho_2(z) &= g(z)\,, && (42)\\ -K\rho_1(z) + \quad \rho_2(z) &= 0\,, && (43) \end{aligned}$$

and after discretization on the fine mesh

$$\left(\begin{bmatrix} \mathbf{I}_{\text{fin}} & \mathbf{0}_{\text{fin}} \\ \mathbf{0}_{\text{fin}} & \mathbf{I}_{\text{fin}} \end{bmatrix} + \begin{bmatrix} \mathbf{0}_{\text{fin}} & \mathbf{M}_{\text{fin}} \\ -\mathbf{K}_{\text{fin}} & \mathbf{0}_{\text{fin}} \end{bmatrix} \right) \begin{bmatrix} \boldsymbol{\rho}_{1\text{fin}} \\ \boldsymbol{\rho}_{2\text{fin}} \end{bmatrix} = \begin{bmatrix} \mathbf{g}_{\text{fin}} \\ 0 \end{bmatrix} . \qquad (44)$$

Standard RCIP gives

$$\left(\begin{bmatrix} \mathbf{I}_{\text{coa}} & \mathbf{0}_{\text{coa}} \\ \mathbf{0}_{\text{coa}} & \mathbf{I}_{\text{coa}} \end{bmatrix} + \begin{bmatrix} \mathbf{0}_{\text{coa}} & \mathbf{M}^{\circ}_{\text{coa}} \\ -\mathbf{K}^{\circ}_{\text{coa}} & \mathbf{0}_{\text{coa}} \end{bmatrix} \begin{bmatrix} \mathbf{R}_1 & \mathbf{R}_3 \\ \mathbf{R}_2 & \mathbf{R}_4 \end{bmatrix} \right) \begin{bmatrix} \tilde{\boldsymbol{\rho}}_{1\text{coa}} \\ \tilde{\boldsymbol{\rho}}_{2\text{coa}} \end{bmatrix} = \begin{bmatrix} \mathbf{g}_{\text{coa}} \\ 0 \end{bmatrix} , \qquad (45)$$

where the compressed inverse $\mathbf{R}$ is partitioned into four equisized blocks.

Now we replace $\tilde{\boldsymbol{\rho}}_{1\text{coa}}$ and $\tilde{\boldsymbol{\rho}}_{2\text{coa}}$ with a single unknown $\tilde{\boldsymbol{\rho}}_{\text{coa}}$ via

$$\begin{aligned} \tilde{\boldsymbol{\rho}}_{1\text{coa}} &= \tilde{\boldsymbol{\rho}}_{\text{coa}} - \mathbf{R}_1^{-1}\mathbf{R}_3\mathbf{K}^{\circ}_{\text{coa}}\mathbf{R}_1\tilde{\boldsymbol{\rho}}_{\text{coa}}\,, && (46)\\ \tilde{\boldsymbol{\rho}}_{2\text{coa}} &= \mathbf{K}^{\circ}_{\text{coa}}\mathbf{R}_1\tilde{\boldsymbol{\rho}}_{\text{coa}}\,. && (47) \end{aligned}$$

The change of variables (46,47) is chosen so that the second block-row of (45) is automatically satisfied. The first block-row of (45) becomes

$$\begin{aligned} \big[\mathbf{I}_{\text{coa}} + \mathbf{M}^{\circ}_{\text{coa}}\left(\mathbf{R}_4 - \mathbf{R}_2\mathbf{R}_1^{-1}\mathbf{R}_3\right)\mathbf{K}^{\circ}_{\text{coa}}\mathbf{R}_1 + \mathbf{M}^{\circ}_{\text{coa}}\mathbf{R}_2 & \\ -\mathbf{R}_1^{-1}\mathbf{R}_3\mathbf{K}^{\circ}_{\text{coa}}\mathbf{R}_1\big]\,\tilde{\boldsymbol{\rho}}_{\text{coa}} = \mathbf{g}_{\text{coa}}\,. & \qquad (48) \end{aligned}$$

When (48) has been solved for $\tilde{\boldsymbol{\rho}}_{\rm coa}$, the weight-corrected version of the original density ρ_1 can be recovered as

$$\hat{\boldsymbol{\rho}}_{1{\rm coa}} = \mathbf{R}_1 \tilde{\boldsymbol{\rho}}_{\rm coa}\,. \tag{49}$$

Figure 13 shows results for (41) with M being the double layer potential

$$M\rho(z) = -2\int_\Gamma \frac{\partial}{\partial\nu_{r'}} G(r,r')\rho(r')\,{\rm d}\sigma_{r'} = \frac{1}{\pi}\int_\Gamma \rho(\tau)\Im\left\{\frac{{\rm d}\tau}{\tau - z}\right\}\,. \tag{50}$$

The left image shows the convergence of q of (6) with $n_{\rm sub}$ using the inner product preserving discretization scheme of Appendix D for (41) as implemented in `demo10.m`. The right image shows q produced with RCIP according to (48,49) as implemented in `demo10b.m`. The reference value for q is computed with the program `demo10c.m`, which uses inner product preserving discretization together with compensated summation [26, 27] in order to enhance the achievable accuracy. One can see that, in addition to being faster, RCIP gives and extra digit of accuracy. Actually, it seems as if the scheme in `demo10.m` converges to a q that is slightly wrong.

In conclusion, in this example and in terms of stability, the RCIP method is better than standard inner product preserving discretization and on par with inner product preserving discretization enhanced with compensated summation. In terms of computational economy and speed, RCIP greatly outperforms the two other schemes.

15 Nyström discretization of singular kernels

The Nyström scheme of Section 4 discretizes (5) using composite 16-point Gauss–Legendre quadrature. This works well as long as the kernel of the integral operator K and the layer density are smooth on smooth Γ. When the kernel is not smooth on smooth Γ, the quadrature fails and something better is needed. See [12] for a comparison of the performance of various modified high-order accurate Nyström discretizations for weakly singular kernels and [28] for a recent high-order general approach to the evaluation of layer potentials.

We are not sure what modified discretization is optimal in every situation. When logarithmic- and Cauchy-singular operators need to be discretized in Sections 16–18, below, we use two modifications to composite Gauss–Legendre quadrature called *local panelwise evaluation* and *local regularization*. See [14, Section 2] for a description of these techniques.

16 The exterior Dirichlet Helmholtz problem

Let D be the domain enclosed by the curve Γ and let E be the exterior to the closure of D. The exterior Dirichlet problem for the Helmholtz equation

$$\Delta U(r) + \omega^2 U(r) = 0\,, \quad r \in E\,, \tag{51}$$

$$\lim_{E \ni r \to r^\circ} U(r) = g(r^\circ)\,, \quad r^\circ \in \Gamma\,, \tag{52}$$

$$\lim_{|r|\to\infty} \sqrt{|r|} \left(\frac{\partial}{\partial |r|} - \mathrm{i}\omega \right) U(r) = 0\,, \tag{53}$$

has a unique solution $U(r)$ under mild assumptions on Γ and $g(r)$ [31] and can be modeled using a combined integral representation [9, Chapter 3]

$$U(r) = \int_\Gamma \frac{\partial}{\partial \nu_{r'}} \Phi_\omega(r,r')\rho(r')\,\mathrm{d}\sigma_{r'} - \frac{\mathrm{i}\omega}{2} \int_\Gamma \Phi_\omega(r,r')\rho(r')\,\mathrm{d}\sigma_{r'}\,, \quad r \in \mathbb{R}^2 \setminus \Gamma\,, \tag{54}$$

where $\Phi_\omega(r,r')$ is the fundamental solution to the Helmholtz equation in two dimensions

$$\Phi_\omega(r,r') = \frac{\mathrm{i}}{4} H_0^{(1)}(\omega|r-r'|)\,. \tag{55}$$

Here $H_0^{(1)}(\cdot)$ is the zeroth order Hankel function of the first kind. Insertion of (54) into (52) gives the combined field integral equation

$$\left(I + K_\omega - \frac{\mathrm{i}\omega}{2} S_\omega \right) \rho(r) = 2g(r)\,, \quad r \in \Gamma\,, \tag{56}$$

where

$$K_\omega \rho(r) = 2 \int_\Gamma \frac{\partial}{\partial \nu_{r'}} \Phi_\omega(r,r')\rho(r')\,\mathrm{d}\sigma_{r'}\,, \tag{57}$$

$$S_\omega \rho(r) = 2 \int_\Gamma \Phi_\omega(r,r')\rho(r')\,\mathrm{d}\sigma_{r'}\,. \tag{58}$$

Figure 14 shows the performance of RCIP applied to (56) for 1000 different values of $\omega \in [1, 10^3]$. The program `demo11.m` is used. The boundary Γ is as in (1) with $\theta = \pi/2$ and the boundary conditions are chosen as $g(r) = H_0^{(1)}(r - r')$ with $r' = (0.3, 0.1)$ inside Γ. The error in $U(r)$ of (54) is evaluated at $r = (-0.1, 0.2)$ outside Γ. Since the magnitude of $U(r)$ varies with ω, peaking at about unity, the absolute error is shown rather than the relative error. The number of panels on the coarse mesh is chosen as `npan=0.6*omega+18` rounded to the nearest integer.

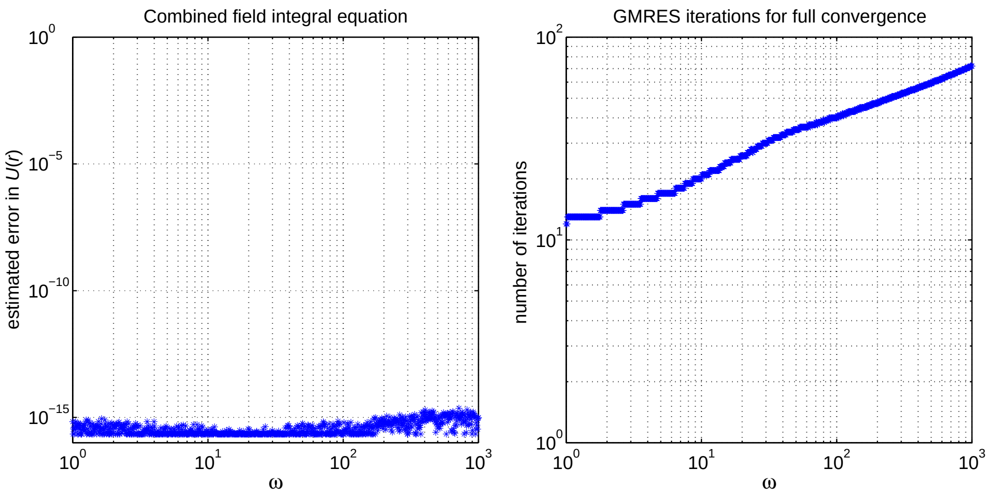

Figure 14: The exterior Dirichlet problem for Helmholtz equation with RCIP applied to (56). The program `demo11.m` is used with Γ as in (1) and $\theta = \pi/2$. The boundary condition $g(r)$ of (52) is generated by a point source at $(0.3, 0.1)$. Left: the absolute error in $U(r)$ at $r = (-0.1, 0.2)$. Right: the number of GMRES iterations needed to meet an estimated relative residual of $\epsilon_{\rm mach}$.

17 The exterior Neumann Helmholtz problem

The exterior Neumann problem for the Helmholtz equation

$$\Delta U(r) + \omega^2 U(r) = 0\,, \quad r \in E\,, \tag{59}$$

$$\lim_{E \ni r \to r^\circ} \frac{\partial U(r)}{\partial \nu_r} = g(r^\circ)\,, \quad r^\circ \in \Gamma\,, \tag{60}$$

$$\lim_{|r| \to \infty} \sqrt{|r|} \left(\frac{\partial}{\partial |r|} - {\rm i}\omega \right) U(r) = 0\,, \tag{61}$$

has a unique solution $U(r)$ under mild assumptions on Γ and $g(r)$ [31] and can be modeled as an integral equation in several ways. We shall consider two options: an “analogy with the standard approach for Laplace's equation”, which is not necessarily uniquely solvable for all ω, and a “regularized combined field integral equation” which is always uniquely solvable. See, further, [2, 8].

17.1 An analogy with the standard Laplace approach

Let K'_ω be the adjoint to the double-layer integral operator K_ω of (57)

$$K'_\omega \rho(r) = 2 \int_\Gamma \frac{\partial}{\partial \nu_r} \Phi_\omega(r, r') \rho(r')\, {\rm d}\sigma_{r'}\,. \tag{62}$$

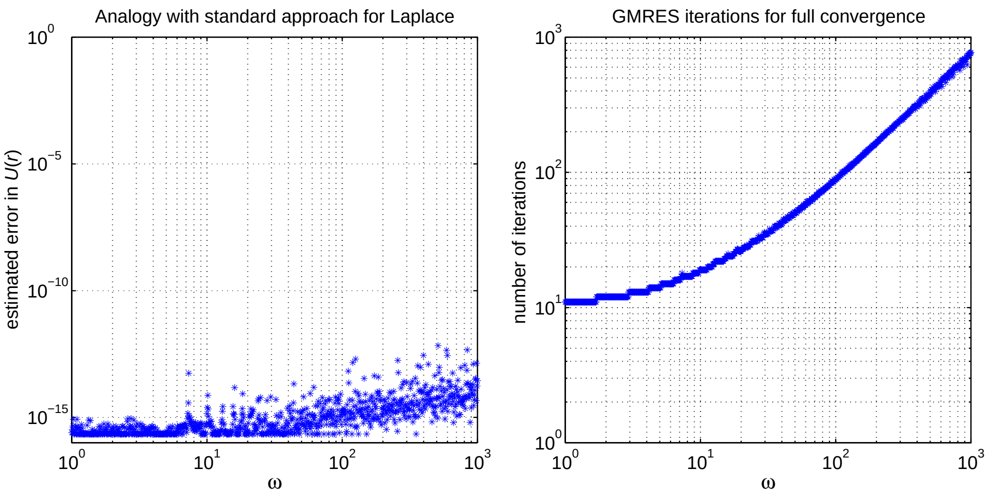

Figure 15: The exterior Neumann problem for Helmholtz equation with RCIP applied to (64). The program `demo12.m` is used with Γ as in (1) and $\theta = \pi/2$. The boundary condition $g(r)$ of (60) is generated by a point source at $(0.3, 0.1)$. Left: the absolute error in $U(r)$ at $r = (-0.1, 0.2)$. Right: the number of GMRES iterations needed to meet an estimated relative residual of $\epsilon_{\rm mach}$.

Insertion of the integral representation

$$U(r) = \int_\Gamma \Phi_\omega(r, r')\rho(r')\,{\rm d}\sigma_{r'}\,, \quad r \in \mathbb{R}^2 \setminus \Gamma \tag{63}$$

into (60) gives the integral equation

$$\left(I - K'_\omega\right)\rho(r) = -2g(r)\,, \quad r \in \Gamma\,. \tag{64}$$

Figure 15 shows the performance of RCIP applied to (64). The program `demo12.m` is used and the setup is the same as that for the Dirichlet problem in Section 16. A comparison between Figure 15 and Figure 14 shows that the number of GMRES iterations needed for full convergence now grows much faster with ω. Furthermore, the relative error in the solution to the Neumann problem is larger, particularly so when ω happens to be close to values for which the operator $I - K'_\omega$ in (64) has a nontrivial nullspace. Recall that (56) is always uniquely solvable while (64) is not.

17.2 A regularized combined field integral equation

The literature on regularized combined field integral equations for the exterior Neumann problem is rich and several formulations have been suggested. We shall use the representation [8]

$$U(r) = \int_\Gamma \Phi_\omega(r, r')\rho(r')\,{\rm d}\sigma_{r'} + {\rm i}\int_\Gamma \frac{\partial}{\partial \nu_{r'}}\Phi_\omega(r, r')\left(S_{{\rm i}\omega}\rho\right)(r')\,{\rm d}\sigma_{r'}\,, \quad r \in \mathbb{R}^2 \setminus \Gamma\,, \tag{65}$$

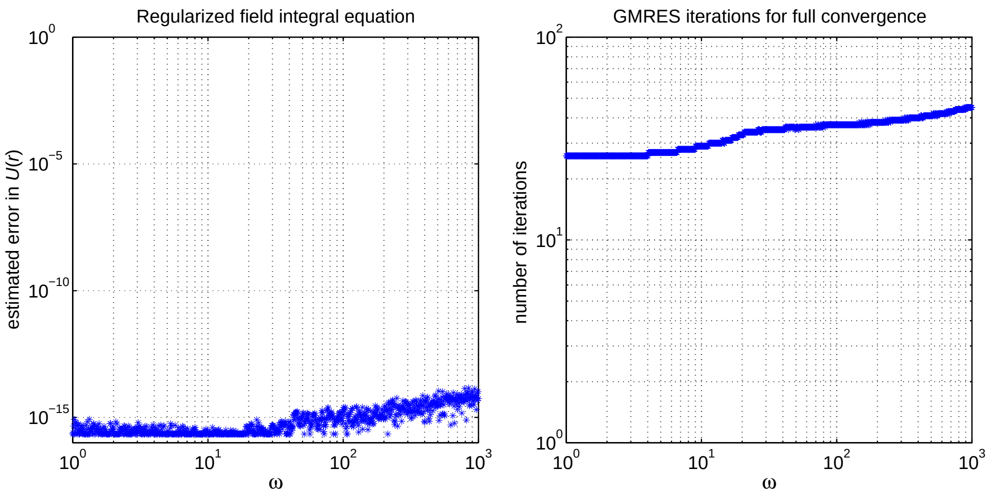

Figure 16: The same exterior Neumann problem for Helmholtz equation as in Figure 15, but RCIP is now applied to (69). The program `demo13b.m` is used.

which after insertion into (60) gives the integral equation

$$\left(I - K'_\omega - \mathrm{i}T_\omega S_{\mathrm{i}\omega}\right)\rho(r) = -2g(r)\,, \quad r \in \Gamma\,, \tag{66}$$

where

$$T_\omega \rho(r) = 2\frac{\partial}{\partial \nu_r}\int_\Gamma \frac{\partial}{\partial \nu_{r'}}\Phi_\omega(r,r')\rho(r')\,\mathrm{d}\sigma_{r'}\,. \tag{67}$$

The hypersingular operator T_ω of (67) can be expressed as a sum of a simple operator and an operator that requires differentiation with respect to arc length only [30]

$$T_\omega \rho(r) = 2\omega^2\int_\Gamma \Phi_\omega(r,r')(\nu_r\cdot\nu_{r'})\rho(r')\,\mathrm{d}\sigma_{r'} + 2\frac{\mathrm{d}}{\mathrm{d}\sigma_r}\int_\Gamma \Phi_\omega(r,r')\frac{\mathrm{d}\rho(r')}{\mathrm{d}\sigma_{r'}}\,\mathrm{d}\sigma_{r'}\,. \tag{68}$$

This makes it possible to write (66) in the form

$$\left(I + A + B_1B_2 + C_1C_2\right)\rho(r) = -2g(r)\,, \quad r \in \Gamma\,, \tag{69}$$

where $A = -K'$, $B_2 = S_{\mathrm{i}\omega}$, and the action of the operators B_1, C_1, and C_2 is given by

$$B_1\rho(r) = -2\mathrm{i}\omega^2\int_\Gamma \Phi_\omega(r,r')(\nu_r\cdot\nu_{r'})\rho(r')\,\mathrm{d}\sigma_{r'}\,, \tag{70}$$

$$C_1\rho(r) = -2\mathrm{i}\frac{\mathrm{d}}{\mathrm{d}\sigma_r}\int_\Gamma \Phi_\omega(r,r')\rho(r')\,\mathrm{d}\sigma_{r'}\,, \tag{71}$$

$$C_2\rho(r) = 2\frac{\mathrm{d}}{\mathrm{d}\sigma_r}\int_\Gamma \Phi_{\mathrm{i}\omega}(r,r')\rho(r')\,\mathrm{d}\sigma_{r'}\,. \tag{72}$$

All integral operators in (69) are such that their discretizations admit the low-rank decomposition (16). We use the temporary expansion technique of Section 14 for (69), with two new layer densities that are later eliminated, to arrive at a single compressed equation analogous to (48). That equation involves nine equisized blocks of the compressed inverse $\mathbf{R}$.

Solving the problem in the example of Section 17.1 again, we now take the number of panels on the coarse mesh as `npan=0.6*omega+48` rounded to the nearest integer. Figure 16 shows results from the program `demo13b.m`. The resonances, visible in Figure 15, are now gone. It is interesting to observe in Figure 16 that, despite the presence of several singular operators and compositions in (69), the results produced with RCIP are essentially fully accurate and the number of GMRES iterations needed for convergence grows very slowly with ω.

The program `demo13c.m` differs from `demo13b.m` in that it uses local regularization for the Cauchy-singular operators of (71) and (72) rather than local panelwise evaluation. The results produced by the two programs are virtually identical, so we do not show yet another figure.

18 Field evaluations

Strictly speaking, a boundary value problem is not properly solved until its solution can be accurately evaluated in the entire computational domain. The program `demo11b.m` is a continuation of `demo11.m` which, after solving (56) for $\tilde{\boldsymbol{\rho}}_{\rm coa}$ with RCIP and forming $\hat{\boldsymbol{\rho}}_{\rm coa}$ via (25), computes the solution $U(r)$ via (54) using three slightly different discretizations:

(i) When r is away from Γ, 16-point Gauss–Legendre quadrature is used in (54) on all quadrature panels.

(ii) When r is close to Γ, but not close to a panel neighboring a corner, 16-point Gauss–Legendre quadrature is used in (54) on panels away from r and local panelwise evaluation is used for panels close to r.

(iii) When r is close to a panel neighboring a corner, the density $\tilde{\boldsymbol{\rho}}_{\rm coa}$ is first used to reconstruct $\hat{\boldsymbol{\rho}}_{\rm part}$ according to Section 9. Then 16-point Gauss–Legendre quadrature is used in (54) on panels away from r and local panelwise evaluation is used for panels close to r.

The first two discretizations only use the coarse grid on Γ. The third discretization needs a grid on a partially refined mesh on Γ.

The program `demo13d.m` is a continuation of `demo13b.m` which, after solving (66) with RCIP as described in Section 17.2, computes the solution $U(r)$ via (65) using the three discretizations of the previous paragraph.

Figure 17 and Figure 18 show that RCIP in conjunction with the modified quadrature rules of [14, Section 2] is capable of producing very accurate

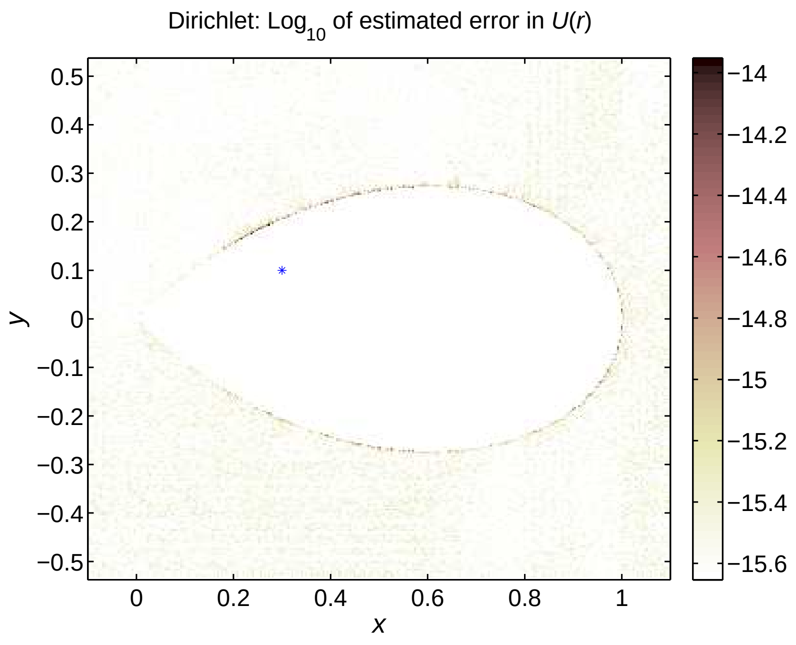

Figure 17: The error in the solution $U(r)$ to the exterior Dirichlet Helmholtz problem of Section 16 evaluated at 62392 points on a cartesian grid using `demo11b.m` with $\omega = 10$. The source at $r' = (0.3, 0.1)$ is shown as a blue star.

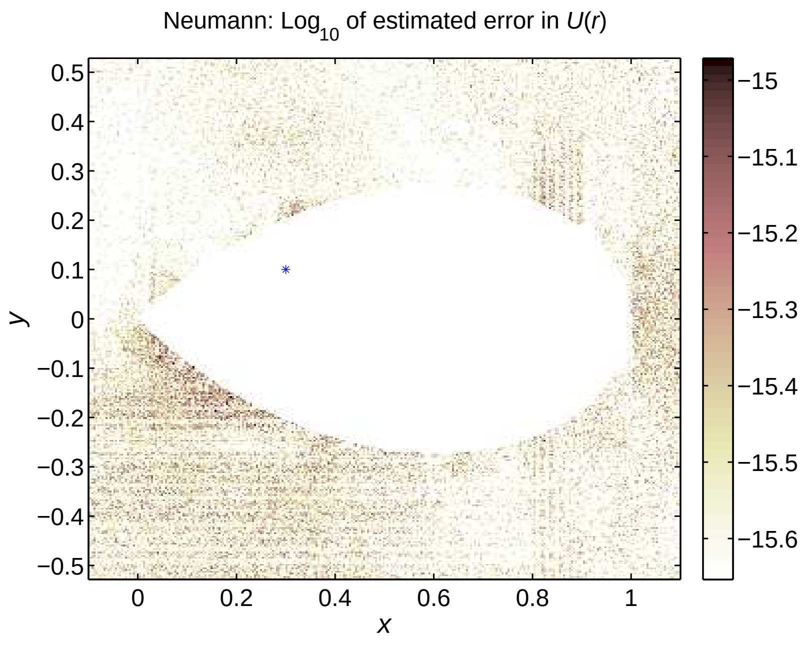

Figure 18: Same as Figure 17, but the exterior Neumann Helmholtz problem is solved using `demo13d.m`. The accuracy is even higher than in Figure 17.

solutions to exterior Helmholtz problems in, essentially, the entire computational domain. Further examples involving the Helmholtz equation in non-smooth domains, along with more details on the discretization of Hankel kernels, are given in [19].

Acknowledgements

The idea to make this tutorial came up during a discussion with Alex Barnett and Adrianna Gillman of 5/20/12 at the FACM'12 conference at NJIT. Input and feedback from these colleagues and from Rikard Ojala has been of great value. The work was supported by the Swedish Research Council under contract 621-2011-5516.

*** Appendicies ***

Appendix A. Proof that $\mathbf{P}_W^T\mathbf{P} = \mathbf{I}_{\rm coa}$

Let $\mathbf{f}_{\rm coa}$ and $\mathbf{g}_{\rm coa}$ be two column vectors, corresponding to the discretization of two panelwise polynomials with panelwise degree 15 on the coarse mesh of Γ. Then

$$\mathbf{f}_{\rm coa}^T\mathbf{W}_{\rm coa}\mathbf{g}_{\rm coa} = (\mathbf{P}\mathbf{f}_{\rm coa})^T\,\mathbf{W}_{\rm fin}\,(\mathbf{P}\mathbf{g}_{\rm coa}) = \mathbf{f}_{\rm coa}^T\mathbf{P}^T\mathbf{W}_{\rm fin}\mathbf{P}\mathbf{g}_{\rm coa}\,, \tag{A.1}$$

because composite 16-point Gauss–Legendre quadrature has panelwise polynomial degree 31. The diagonal matrix $\mathbf{W}_{\rm coa}$ has size $16n_{\rm pan} \times 16n_{\rm pan}$.

Since there are $16n_{\rm pan}$ linearly independent choices of $\mathbf{f}_{\rm coa}$ and of $\mathbf{g}_{\rm coa}$ it follows from (A.1) that

$$\mathbf{W}_{\rm coa} = \mathbf{P}^T\mathbf{W}_{\rm fin}\mathbf{P}\,, \tag{A.2}$$

which, using (13), can be rewritten

$$\mathbf{I}_{\rm coa} = \mathbf{W}_{\rm coa}^{-1}\mathbf{P}^T\mathbf{W}_{\rm fin}\mathbf{P} = \mathbf{P}_W^T\mathbf{P}\,. \tag{A.3}$$

Appendix B. Derivation of the compressed equation

The compression of (8), leading up to (17), was originally described in [21, Section 6.4]. Here we give a summary.

The starting point is (5) which, using the operator split analogous to (11,12)

$$K = K^\star + K^\circ \tag{B.1}$$

and the variable substitution

$$\rho(z) = (I + \lambda K^\star)^{-1}\,\tilde{\rho}(z)\,, \tag{B.2}$$

gives the right preconditioned equation

$$\tilde{\rho}(z) + \lambda K^\circ (I + \lambda K^\star)^{-1}\tilde{\rho}(z) = \lambda g(z)\,, \quad z \in \Gamma\,. \tag{B.3}$$

Now, let us take a close look at (B.3). We observe that $K^\circ(I + \lambda K^\star)^{-1}$ is an operator whose action on any function gives a function that is smooth on the innermost two panels of the coarse grid on $\Gamma^\star$. This is so since K° is constructed so that its action on any function gives a function that is smooth on the innermost two panels of the coarse grid on $\Gamma^\star$. Furthermore, the right hand side $\lambda g(z)$ of (B.3) is assumed to be panelwise smooth. Using an argument of contradiction we see that $\tilde{\rho}(z)$ has to be a panelwise smooth function on the innermost two panels of the coarse grid on $\Gamma^\star$.

Having concluded that $\tilde{\rho}(z)$ is panelwise smooth close to the corner we can write

$$\tilde{\boldsymbol{\rho}}_{\text{fin}} = \mathbf{P}\tilde{\boldsymbol{\rho}}_{\text{coa}}. \tag{B.4}$$

We also have

$$\mathbf{g}_{\text{fin}} = \mathbf{P}\mathbf{g}_{\text{coa}}\,, \tag{B.5}$$

the discrete version of (B.2) on the fine grid

$$\boldsymbol{\rho}_{\text{fin}} = \left(\mathbf{I}_{\text{fin}} + \lambda \mathbf{K}^\star_{\text{fin}}\right)^{-1}\tilde{\boldsymbol{\rho}}_{\text{fin}}\,, \tag{B.6}$$

and the relations (12) and (16) which we now repeat:

$$\mathbf{K}_{\text{fin}} = \mathbf{K}^\star_{\text{fin}} + \mathbf{K}^\circ_{\text{fin}}\,, \tag{B.7}$$

$$\mathbf{K}^\circ_{\text{fin}} = \mathbf{P}\mathbf{K}^\circ_{\text{coa}}\mathbf{P}^T_W\,. \tag{B.8}$$

Substitution of (B.4,B.5,B.6,B.7,B.8) into (8), which we now repeat:

$$\left(\mathbf{I}_{\text{fin}} + \lambda \mathbf{K}_{\text{fin}}\right)\boldsymbol{\rho}_{\text{fin}} = \lambda \mathbf{g}_{\text{fin}}\,, \tag{B.9}$$

gives

$$\mathbf{P}\tilde{\boldsymbol{\rho}}_{\text{coa}} + \lambda \mathbf{P}\mathbf{K}^\circ_{\text{coa}}\mathbf{P}^T_W\left(\mathbf{I}_{\text{fin}} + \lambda \mathbf{K}^\star_{\text{fin}}\right)^{-1}\mathbf{P}\tilde{\boldsymbol{\rho}}_{\text{coa}} = \mathbf{P}\mathbf{g}_{\text{coa}}\,. \tag{B.10}$$

Applying $\mathbf{P}^T_W$ (or $\mathbf{Q}$) to the left in (B.10) and using the identities (14) (or (30)) gives the compressed equation (17).

Appendix C. Derivation of the recursion

The recursion (21) for the rapid construction of the diagonal blocks of the compressed weighted inverse $\mathbf{R}$ was originally derived in [21, Section 7] using different notation and different meshes than in the present tutorial. The recursion was derived a second time in [22, Section 7] using new meshes. Better notation was introduced in [14, Section 6]. A third derivation, in a general setting, takes place in [15, Section 5] and it uses the same notation and meshes as in the present tutorial.

A problem when explaining the derivation of (21) is that one needs to introduce intermediate meshes and matrices whose appearance may cause enervation at a first glance. Particularly so since these meshes and matrices are not needed in the final expression (21). We emphasize that the underlying matrix property that permits the recursion is the low rank of certain off-diagonal blocks in discretizations of K° of (B.1) on nested meshes.

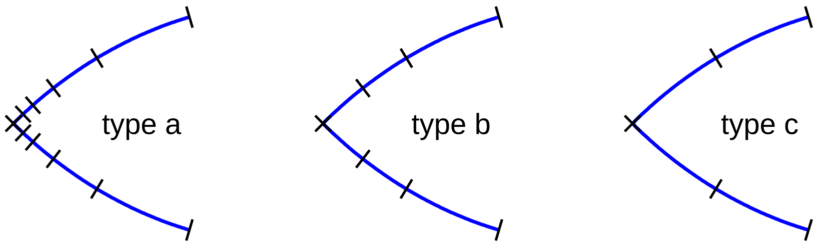

Figure 19: Meshes of type a, type b, and type c on the boundary subset $\Gamma_i^\star$ for $i = 3$ and $n_{\rm sub} = 3$. The type a mesh has $4 + 2i$ panels. The type b mesh has six panels. The type c mesh has four panels. The type a mesh is the restriction of the fine mesh to $\Gamma_i^\star$. For $i = n_{\rm sub}$, the type c mesh is the restriction of the coarse mesh to $\Gamma^\star$. The type a mesh and the type b mesh coincide for $i = 1$.

The recursion (21) only uses uses one type of mesh explicitly – the type b mesh of Figure 5. On each $\Gamma_i^\star$ there is a type b mesh and a corresponding discretization of K° denoted $\mathbf{K}_{i{\rm b}}^\circ$. Here we need two new types of meshes denoted type a and type c, along with corresponding discrete operators. For example, $\mathbf{K}_{i{\rm a}}$ is the discretization of K on a type a mesh on $\Gamma_i^\star$. The three types of meshes are depicted in Figure 19. Actually, a straight type c mesh was already introduced in Figure 4.

Now we define $\mathbf{R}_i$ as

$$\mathbf{R}_i \equiv \mathbf{P}_{Wi{\rm ac}}^T \left(\mathbf{I}_{i{\rm a}} + \lambda \mathbf{K}_{i{\rm a}}\right)^{-1} \mathbf{P}_{i{\rm ac}}\,, \tag{C.1}$$

where $\mathbf{P}_{Wi{\rm ac}}$ and $\mathbf{P}_{i{\rm ac}}$ are prolongation operators (in parameter) from a grid on a type c mesh on $\Gamma_i^\star$ to a grid on a type a mesh on $\Gamma_i^\star$. Note that $\mathbf{R}_i$ for $i = n_{\rm sub}$, according to the definition (C.1), is identical to the full diagonal 64×64 block of $\mathbf{R}$ of (18). Note also that $\mathbf{R}_1$ comes cheaply. The rest of this appendix is about finding an expression for $\mathbf{R}_i$ in terms of $\mathbf{R}_{i-1}$ that is cheap to compute.

Let us split $\mathbf{K}_{ia}$ into two parts

$$\mathbf{K}_{i\mathrm{a}} = \mathbf{K}^{\star}_{i\mathrm{a}} + \mathbf{K}^{\circ}_{i\mathrm{a}}\,, \tag{C.2}$$

where $\mathbf{K}^{\star}_{i\mathrm{a}} = \mathbb{F}\{\mathbf{K}_{(i-1)\mathrm{a}}\}$ and $\mathbf{K}^{\circ}_{i\mathrm{a}}$ is such that

$$\mathbf{K}^{\circ}_{i\mathrm{a}} = \mathbf{P}_{i\mathrm{ab}}\mathbf{K}^{\circ}_{i\mathrm{b}}\mathbf{P}^{T}_{Wi\mathrm{ab}} \tag{C.3}$$

holds to about machine precision, compare (16). The prolongation operators $\mathbf{P}_{i\mathrm{ab}}$ and $\mathbf{P}_{Wi\mathrm{ab}}$ act from a grid on a type b mesh to a grid on a type a mesh. It holds that

$$\mathbf{P}_{i\mathrm{ac}} = \mathbf{P}_{i\mathrm{ab}}\mathbf{P}_{\mathrm{bc}}\,, \tag{C.4}$$

$$\mathbf{P}_{Wi\mathrm{ac}} = \mathbf{P}_{Wi\mathrm{ab}}\mathbf{P}_{W\mathrm{bc}}\,. \tag{C.5}$$

Summing up, we can rewrite (C.1) as

$$\mathbf{R}_i = \mathbf{P}^{T}_{W\mathrm{bc}}\mathbf{P}^{T}_{Wi\mathrm{ab}}\left(\mathbf{I}_{i\mathrm{a}} + \mathbb{F}\{\lambda\mathbf{K}_{(i-1)\mathrm{a}}\} + \lambda\mathbf{P}_{i\mathrm{ab}}\mathbf{K}^{\circ}_{i\mathrm{b}}\mathbf{P}^{T}_{Wi\mathrm{ab}}\right)^{-1}\mathbf{P}_{i\mathrm{ab}}\mathbf{P}_{\mathrm{bc}}\,. \tag{C.6}$$

The subsequent steps in the derivation of (21) are to expand the term within parenthesis in (C.6) in a Neumann series, multiply the terms in this series with $\mathbf{P}^{T}_{Wi\mathrm{ab}}$ from the left and with $\mathbf{P}_{i\mathrm{ab}}$ from the right, and bring the series back in closed form. The result is

$$\mathbf{R}_i = \mathbf{P}^{T}_{W\mathrm{bc}}\left[\left(\mathbf{P}^{T}_{Wi\mathrm{ab}}\left(\mathbf{I}_{i\mathrm{a}} + \mathbb{F}\{\lambda\mathbf{K}_{(i-1)\mathrm{a}}\}\right)^{-1}\mathbf{P}_{i\mathrm{ab}}\right)^{-1} + \lambda\mathbf{K}^{\circ}_{i\mathrm{b}}\right]^{-1}\mathbf{P}_{\mathrm{bc}}\,, \tag{C.7}$$

which, in fact, is (21) in disguise. To see this, recall from (C.1) that

$$\mathbf{R}_{(i-1)} \equiv \mathbf{P}^{T}_{W(i-1)\mathrm{ac}}\left(\mathbf{I}_{(i-1)\mathrm{a}} + \lambda\mathbf{K}_{(i-1)\mathrm{a}}\right)^{-1}\mathbf{P}_{(i-i)\mathrm{ac}}\,. \tag{C.8}$$

Then

$$\begin{aligned}\mathbb{F}\{\mathbf{R}_{(i-1)}\} &= \mathbb{F}\{\mathbf{P}^{T}_{W(i-1)\mathrm{ac}}\left(\mathbf{I}_{(i-1)\mathrm{a}} + \lambda\mathbf{K}_{(i-1)\mathrm{a}}\right)^{-1}\mathbf{P}_{(i-i)\mathrm{ac}}\}\\ &= \mathbf{P}^{T}_{Wi\mathrm{ab}}\left(\mathbf{I}_{i\mathrm{a}} + \mathbb{F}\{\lambda\mathbf{K}_{(i-1)\mathrm{a}}\}\right)^{-1}\mathbf{P}_{i\mathrm{ab}} - \mathbf{I}^{\circ}_{\mathrm{b}}\,, \end{aligned} \tag{C.9}$$

where the second equality uses $\mathbf{P}^{T}_{Wi\mathrm{ab}}\mathbf{P}_{i\mathrm{ab}} = \mathbf{I}_{\mathrm{b}}$, see Appendix A. Substitution of (C.9) in (C.7) gives the recursion in the familiar form

$$\mathbf{R}_i = \mathbf{P}^{T}_{W\mathrm{bc}}\left(\mathbb{F}\{\mathbf{R}^{-1}_{i-1}\} + \mathbf{I}^{\circ}_{\mathrm{b}} + \lambda\mathbf{K}^{\circ}_{i\mathrm{b}}\right)^{-1}\mathbf{P}_{\mathrm{bc}}\,. \tag{C.10}$$

Appendix D. An inner product preserving scheme

In [3], Bremer describes a scheme that stabilizes the solution to the discretized system (8) on the fine mesh. The scheme can be interpreted as an inner product preserving discretization. In practice it corresponds to making a similarity transformation of the system matrix. While inner product preserving Nyström discretization elegantly solves problems related to stability (the condition number of the system matrix is improved) it does not reduce the number of discretization points (unknowns) needed to achieve a given precision in the solution. Neither does it affect the spectrum of the system matrix (similarity transformations preserve eigenvalues) and hence it does not in any substantial way improve the convergence rate of the GMRES iterative method [34, Lecture 35].

For completeness, we have implemented inner product preserving Nyström discretization in the program `demo1d.m`. The program is a continuation of `demo1b.m` where we also have replaced (3) with the more stable integral equation (24). This should facilitate comparison with the program `demo3b.m` and the results shown in Figure 7.

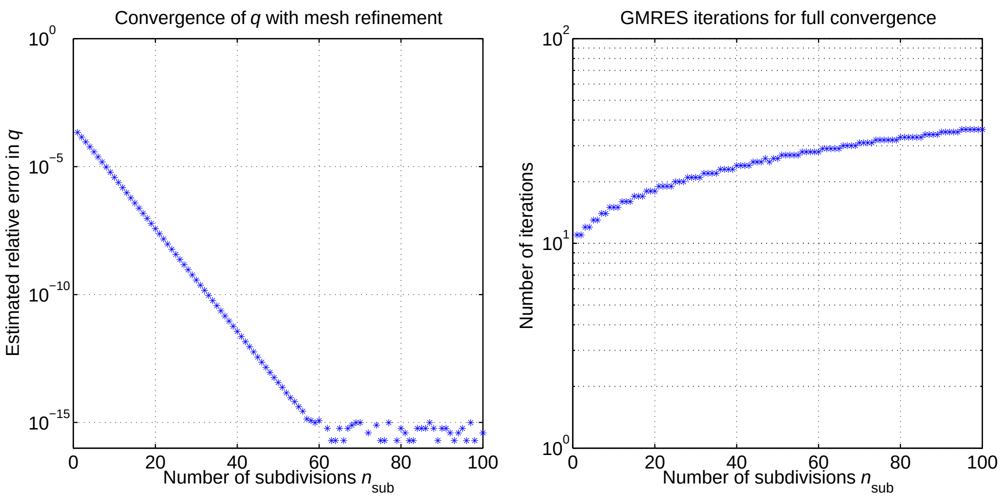

Figure 20: Same as Figure 7, but the program `demo1d.m` is used.

Figure 20 shows results produced by `demo1d.m`. Beyond $n_{\text{sub}} = 60$ one now achieves essentially full machine precision in q of (6). Despite this success, inner product preserving Nyström discretization can perhaps not quite compete with the RCIP method in this example. The differences in performance relate to issues of memory and speed. The RCIP method uses a much smaller linear system ($16n_{\text{pan}}$ unknowns) than does inner product preserving Nyström discretization ($16(n_{\text{pan}} + 2n_{\text{sub}})$ unknowns). Besides, the RCIP method converges in only eight GMRES iterations, irrespective of n_{sub}. See Figure 7.

Appendix E. The nature of layer densities in corners

It is considered difficult to solve integral equations on piecewise smooth boundaries. One reason being that solutions (layer densities) often have complicated asymptotics close to corner vertices. Also stable numerical schemes, such as that of Appendix D, are burdened with the task of resolution. It takes a lot of mesh refinement to resolve non-polynomial-like layer densities with polynomial basis functions.

Sometimes layer densities diverge, sometimes they stay bounded, but they are seldom polynomial-like close to corner vertices. While we refrain from giving a characterization of what properties of Fredholm second kind integral equations give rise to unbounded solutions we observe the following:

- The solution $\rho(r)$ to (3) diverges in corners whenever $\Re\{\lambda\} > 0$ and λ is such that $\rho(r)$ exists. Figure 9 illustrates this for $\lambda = 0.999$.
- The size of the zone where the asymptotic shape of a layer density is visible depends on the smoothness of the solution to the PDE that underlies the integral equation. See [14, Figure 3].
- The divergence of layer densities close to corner vertices can be very strong. In classical materials science applications it is not unusual with layer densities that barely lie in L^2 [21, Section 2]. In metamaterial applications layer densities may barely lie in L^1 [17, Section 3]. Furthermore, it is likely that L^p spaces are not the most natural function spaces for the characterization of layer densities [23, Section 5].

Variable separation is often used to predict the asymptotic behavior of layer densities close to corner vertices. See, for example, [21, Section 2] and [13, Section 2]. But perhaps asymptotic studies of layer densities are not that important? Assume that there are two different Fredholm second kind integral equations for the same underlying PDE: one where the layer density is bounded and one where it is unbounded. If the RCIP method is to be used, it really does not matter which equation is chosen. The recursion (21,22) might need fewer steps for the equation with a bounded solution. The achievable accuracy for functionals of the layer density, on the other hand, is often slightly higher if computed from the unbounded solution. The reason being, loosely speaking, that more information can be contained in a rapidly varying function than in a slowly varying function.

References

[1] K.E. Atkinson, *The Numerical Solution of Integral Equations of the Second Kind*, Cambridge University Press, Cambridge, 1997.

[2] J. Bremer, ‘A fast direct solver for the integral equations of scattering theory on planar curves with corners’, J. Comput. Phys., **231**, 1879–1899 (2012).

[3] J. Bremer, ‘On the Nyström discretization of integral equations on planar curves with corners’, Appl. Comput. Harmon. Anal., **32**, 45–64 (2012).

[4] J. Bremer, Z. Gimbutas, and V. Rokhlin, A nonlinear optimization procedure for generalized Gaussian quadratures, SIAM J. Sci. Comput., **32** (2010) 1761–1788.

[5] J. Bremer and V. Rokhlin, ‘Efficient discretization of Laplace boundary integral equations on polygonal domains’, J. Comput. Phys., **229**, 2507–2525 (2010).

[6] J. Bremer, V. Rokhlin, and I. Sammis, ‘Universal quadratures for boundary integral equations on two-dimensional domains with corners’, J. Comput. Phys., **229**, 8259–8280 (2010).

[7] O.P. Bruno, J.S. Ovall, and C. Turc, ‘A high-order integral algorithm for highly singular PDE solutions in Lipschitz domains’, Computing, **84**, 149–181 (2009).

[8] O.P. Bruno, T. Elling, and C. Turc, ‘Regularized integral equations and fast high-order solvers for sound-hard acoustic scattering problems’, Int. J. Numer. Meth. Eng., **91**, 1045–1072 (2012).

[9] D. Colton and R. Kress, *Inverse Acoustic and Electromagnetic Scattering Theory*, 2nd ed., Springer, Berlin, 1998.

[10] J. Englund, ‘A higher order scheme for two-dimensional quasi-static crack growth simulations’, Comp. Meth. Appl. Mech. Engrg., **196**, 2527–2538 (2007).

[11] L. Greengard and V. Rokhlin, ‘A fast algorithm for particle simulations’, J. Comput. Phys., **73**, 325–348 (1987).

[12] S. Hao, A.H. Barnett, P.G. Martinsson, and P. Young, ‘High-order accurate methods for Nyström discretization of integral equations on smooth curves in the plane’, Adv. Comput. Math., **40**, 245–272 (2014).

[13] J. Helsing, ‘Corner singularities for elliptic problems: special basis functions versus "brute force"’, Comm. Numer. Methods Engrg., **16**, 37–46, (2000).

[14] J. Helsing, ‘Integral equation methods for elliptic problems with boundary conditions of mixed type’, J. Comput. Phys., **228**, 8892–8907 (2009).

[15] J. Helsing, ‘A fast and stable solver for singular integral equations on piecewise smooth curves’, SIAM J. Sci. Comput., **33**, 153–174 (2011).

[16] J. Helsing, ‘The effective conductivity of random checkerboards’, J. Comput. Phys., **230**, 1171–1181 (2011).

[17] J. Helsing, ‘The effective conductivity of arrays of squares: large random unit cells and extreme contrast ratios’, J. Comput. Phys., **230**, 7533–7547 (2011).

[18] J. Helsing and A. Jonsson, ‘On the computation of stress fields on polygonal domains with V-notches’, Int. J. Numer. Meth. Eng., **53**, 433–454 (2002).

[19] J. Helsing and A. Karlsson, ‘An accurate boundary value problem solver applied to scattering from cylinders with corners’, IEEE Trans. Antennas Propag., **61**, 3693–3700 (2013).

[20] J. Helsing and R. Ojala, ‘On the evaluation of layer potentials close to their sources’, J. Comput. Phys., **227**, 2899–2921 (2008).

[21] J. Helsing and R. Ojala, ‘Corner singularities for elliptic problems: Integral equations, graded meshes, quadrature, and compressed inverse preconditioning’, J. Comput. Phys., **227**, 8820–8840 (2008).

[22] J. Helsing and R. Ojala, ‘Elastostatic computations on aggregates of grains with sharp interfaces, corners, and triple-junctions’, Int. J. Solids Struct., **46**, 4437–4450 (2009).

[23] J. Helsing and K-.M. Perfekt, ‘On the polarizability and capacitance of the cube’, Appl. Comput. Harmon. Anal., **34**, 445–468 (2013).

[24] J. Helsing and G. Peters, ‘Integral equation methods and numerical solutions of crack and inclusion problems in planar elastostatics’, SIAM J. Appl. Math., **59**, 965–982 (1999).

[25] H.V. Henderson and S.R. Searle, ‘On deriving the inverse of a sum of matrices’, SIAM Rev. **23**, 53–60 (1981).

[26] N.J. Higham, *Accuracy and stability of numerical algorithms*, SIAM, Philadelphia, 1996, 92–97.

[27] W. Kahan, ‘Further remarks on reducing truncation errors’, Comm. ACM, **8**, 40 (1965).

[28] A. Klöckner, A. Barnett, L. Greengard, and M. O’Neil, ‘Quadrature by expansion: A new method for the evaluation of layer potentials’, J. Comput. Phys., **252**, 332–349 (2013).

[29] W.Y. Kong, J. Bremer, and V. Rokhlin, “An adaptive fast direct solver for boundary integral equations in two dimensions”, Appl. Comput. Harmon. Anal., **31** (2011) 346–369.

[30] R. Kress, ‘On the numerical solution of a hypersingular integral equation in scattering theory’, J. Comput. Appl. Math., **61**, 345–360 (1995).

[31] M. Mitrea, ‘Boundary value problems and Hardy spaces associated to the Helmholtz equation in Lipschitz domains’, J. Math. Anal. Appl., **202**, 819–842 (1996).

[32] R. Ojala, ‘A robust and accurate solver of Laplace’s equation with general boundary conditions on general domains in the plane’, J. Comp. Math., **30**, 433–448 (2012).

[33] Y. Saad and M.H. Schultz, ‘GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems’, SIAM J. Sci. Stat. Comp., **7**, 856–869 (1986).

[34] L.N. Trefethen and D. Bau, III, *Numerical Linear Algebra*, SIAM, Philadelphia, PA, 1997.