Theory & Computation of Singularly Perturbed Differential Equations

IIT (BHU) Varanasi, Dec 2017

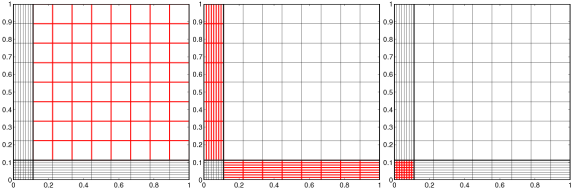
https://skumarmath.wordpress.com/gian-17/singular-perturbation-problems/

http://www.maths.nuigalway.ie/~niall/TCSPDEs2017

Niall Madden, NUI Galway

§10 Preconditioning for SPDEs

Version 07.12.17



Presentation version: not for printing

Outline

Monday, 4 December		
09:30 – 10.30	Registration and Inauguration	
10:45 – 11.45	1. Introduction to singularly perturbed problems	NM
12:00 – 13:00	2. Numerical methods and uniform convergence	NM
14:30 – 15:30	Tutorial (Convection diffusion problems)	NM
15:30 – 16:30	Lab 1 (Simple FEMs in MATLAB)	NM

Tuesday, 5 December		
09:30 – 10:30	3. Finite difference methods and their analyses	NM
10:45 – 11:45	4. Coupled systems of SPPDEs	NM
14:00 – 16:00	Lab 2 (Fitted mesh methods for ODEs)	NM

Thursday, 7 December		
09:00 – 10:00	8. Singularly perturbed elliptic PDEs	NM
10:15 – 11:15	9. Finite Elements in two and three dimensions	NM
01:15 – 15:15	Lab 4 (Singularly perturbed PDEs)	NM

Friday, 8 December		
09:00 – 10:00	10. Preconditioning for SPPs	NM

§10 Preconditioning for SPDEs

(≈ 60 minutes)

All this week we have studied how to convert boundary value problems into systems of linear algebraic equations.

In this lecture, we will focus on how to solve such systems.

We'll see that direct solvers are not always suitable for singularly perturbed problems.

That will encourage us to consider the topic of *preconditioning*.

1. Outline
2. From DEs to linear systems
   - 1D
   - 2D
3. Singularly perturbed problems
   - Fitted meshes
4. Are these methods really "robust"?
5. A direct solver
6. Preconditioners
   - Diagonal Preconditioned CG
7. A boundary layer preconditioner
   - Interior
   - Edge
   - Corner
8. Specifying the preconditioner
   - Spectral Equivalence
   - Multigrid in Corners
9. Preconditioner results
10. Summary

Outline

This lecture is concerned with two aspects of the solution of large linear systems of equations that arise in the solution of two-dimensional *singularly perturbed* problems by finite difference methods. In particular, I aim to:

(a) explain the curious **poor behaviour** of direct solvers for these problems;

(b) describe some robust preconditioners based on standard approaches, based on;

(c) outline the design of a new **boundary layer preconditioner** based on multigrid methods.

Unlike other lectures in this series, it based largely on my own work, with collaborators Scott MacLachlan [MacLachlan and Madden, 2013], and Tháí Anh Nhan [Nhan and Madden, 2015b, Nhan and Madden, 2015a]. In particular numerous details are from Nhan's PhD thesis.

From DEs to linear systems

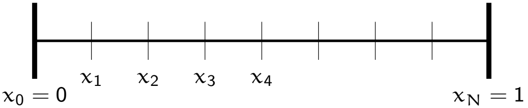
Suppose we want to find a numerical solution to the one-dimensional differential equation

$$-u''(x) + b(x)u(x) = f(x) \quad \text{on } (0,1),$$

with boundary conditions $u(0) = u(1) = 0$, and where $b > 0$, and f are continuous functions.

AS we now know, the simplest numerical scheme one could apply to this problem is a 2^{nd} **order finite difference scheme on a uniform mesh**: choose a number of intervals N, set $h = 1/N$, and generate mesh points

$$\{x_0, x_1, x_2, \dots, x_N\} = \{0, h, 2h, \dots, 1\}$$



We'll find approximations for $u(x)$ at each of these points:

$$\left\{u(x_0), u(x_1), u(x_2), \dots, u(x_N)\right\} = \left\{U_0, U_1, U_2, \dots, U_N\right\}$$

From DEs to linear systems

To construct these approximations, we replace the *differential* equation with a *difference* equation, using

$$u''(x_i) = \underbrace{\frac{1}{h^2}\big(u(x_{i-1} - 2u(x_i) + u(x_{i+1})\big)}_{\delta^2 u(x_i)} + \underbrace{\|\frac{d^4}{dx^4}u\|_\infty N^{-2}}_{\text{truncation error}}$$

Then construct and solve the linear system

$$\begin{gathered} \mathbf{U}_0 = 0, \\ -\delta^2 U_i + b(x_i)U_i = f(x_i), \qquad i = 1, \dots, N-1 \\ \mathbf{U}_N = 0. \end{gathered}$$

Using some standard techniques, such as maximum principles and the properties of M-matrices, one can prove that error $|u(x_i) - U_i|$ is bounded by the truncation error.

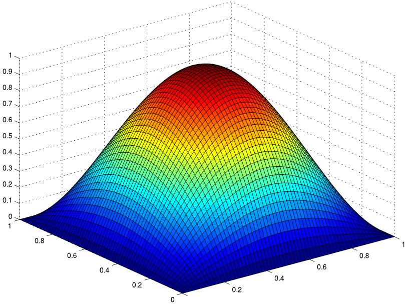
Implementing the method involves solving the tridiagonal linear system, which can be done with optimal complexity using, say, the Thomas Algorithm.

From DEs to linear systems 2D

The simplest two-dimensional partial differential equation we'll consider is

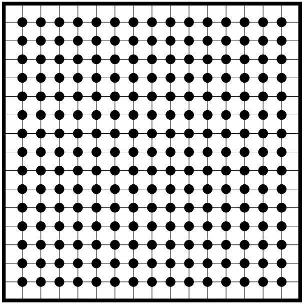
$$-(u_{xx} + u_{yy}) + b(x, y)u = f(x, y)$$

posed on the unit square, with $u = 0$ on the boundary.



To solve this by a finite difference method, we partition $(0, 1)^2$ into an $(N + 1) \times (N + 1)$ grid, and again approximate the differential operator by finite differences...

From DEs to linear systems



$$[u_{xx}+u_{yy}](x_i,y_j)=\frac{1}{h^2}\big(u(x_{i-1},y_j)-2u(x_i,y_j)+u(x_{i+1},y_j)\big)$$
$$+\frac{1}{h^2}\big(u(x_i,y_{j-1})-2u(x_i,y_j)+u(x_i,y_{j+1})\big)+\underbrace{\|\frac{\partial^4}{\partial x^4}u+\frac{\partial^4}{\partial y^4}u\|_\infty N^{-2}}_{\text{truncation error}}$$

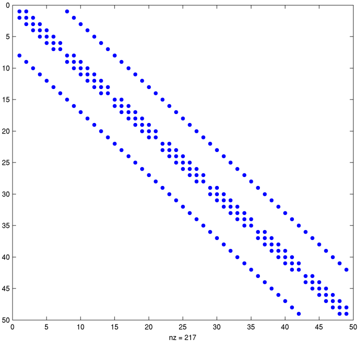
From DEs to linear systems 2D

The linear systems of equations, $Au = h^2 f(x_i, y_j)$, where, say, row $p = i + (j-1)(N-1)$ of A can be expressed succinctly using a (scaled) 5-point stencil:

$$\begin{pmatrix} & A_{p,p+(N-1)} & \\ A_{p,p-1} & A_{p,p} & A_{p,p+1} \\ & A_{p,p-(N-1)} & \end{pmatrix} = \begin{pmatrix} & 1 & \\ 1 & -4 + h^2 b(x_i, y_j) & 1 \\ & 1 & \end{pmatrix}$$

The system matrix is *symmetric positive definite*, and banded, with a band-width of $(N-1)$. Constructing an optimal algorithm to solve this linear system is quite tricky.

The standard *direct* method is *Cholesky* factorisation, which is probably the best approach for “small” problems (no more than about 1,000,000 unknowns, for standard desktop). Complexity is, roughly, $\mathcal{O}(N^3)$.



From DEs to linear systems 2D

The leading *iterative* techniques are based around Conjugate Gradients and/or optimal Multigrid methods, which is $\mathcal{O}(N^2)$. But these are not black-box methods...

While they are well-understood for standard problems, the same cannot be said for linear systems that come from the discretizations of singularly perturbed problems.

Singularly perturbed problems

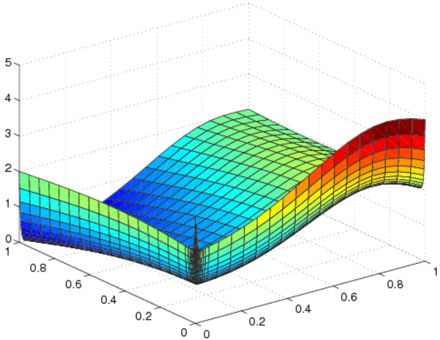
A two-dimensional model problem

$$-\varepsilon^2 \Delta u + bu = f \quad \text{on } (0,1)\times(0,1) + \text{BCs}.$$

It is assumed that $\varepsilon \in (0,1]$, and $b(x,y) > \beta > 0$.

Our test case is a variant on a standard test problem (see, e.g., [Clavero et al., 2005]), but simplified to have only two boundary layers, near the edges $x=0$ and $y=0$, and a corner layer near $(0,0)$. We take $b(x,y)=1$, choose f and g so that

$$u = x^3(1+y^2)+\sin(\pi x^2)+\cos(\pi y/2) + (1+x+y)\big(e^{-2x/\varepsilon}+e^{-2y/\varepsilon}\big).$$



Singularly perturbed problems

We'll use a finite difference scheme on a Cartesian-product grid. Form two one-dimensional meshes on $[0,1]$: $\{x_0, x_1, \dots, x_N\}$ and $\{y_0, y_1, \dots, y_N\}$. Set $\Omega^N = \{(x_i, y_j)\}_{i,j=0}^N$.

Set $\quad h_i = x_i - x_{i-1}, \qquad k_j = y_j - y_{j-1},$
and $\quad \bar{h}_i = (x_{i+1} - x_{i-1})/2, \quad \bar{k}_j = (y_{j+1} - y_{j-1})/2.$

The scaled 5-point second-order central difference (discrete Laplacian) operator is:

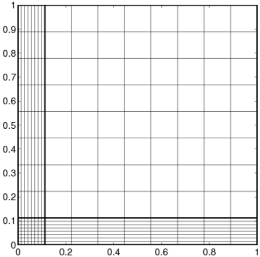
$$\Delta_{ij}^N := \begin{pmatrix} & \dfrac{\bar{h}_i}{k_{j+1}} & \\ \dfrac{\bar{k}_j}{h_i} & -\bar{k}_j\left(\dfrac{1}{h_i} + \dfrac{1}{h_{i+1}}\right) - \bar{h}_i\left(\dfrac{1}{k_j} + \dfrac{1}{k_{j+1}}\right) & \dfrac{\bar{k}_j}{h_{i+1}} \\ & \dfrac{\bar{h}_i}{k_j} & \end{pmatrix}.$$

Then (neglecting boundary conditions), the numerical scheme is:

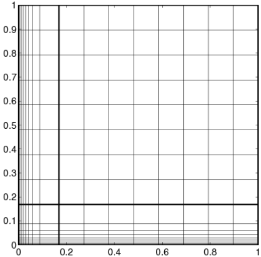
$$\left(-\varepsilon^2 \Delta_{ij}^N + \bar{h}_i \bar{k}_j b_{ij}\right) U_{ij} = \bar{h}_i \bar{k}_j f_{ij}. \qquad \text{We write this as: } A_\varepsilon U = F.$$

Singularly perturbed problems

Shishkin mesh
$(\tau = \min\left\{\frac{1}{2}, 2\frac{\varepsilon}{\beta}\ln N\right\})$



Bakhvalov mesh



FDMs on such meshes yield uniformly convergent approximations, U, i.e., there exists a constant C, independent of ε and N such that

$$\|\mathbf{u} - \mathbf{U}\|_{\infty,\bar{\Omega}^N} \leqslant C \begin{cases} N^{-2}\ln^2 N & \text{Shishkin: [Clavero et al., 2005]} \\ N^{-2} & \text{Bakhvalov: [Kellogg et al., 2008].} \end{cases}$$

As expected, for both meshes, the computed solution is robust with respect to ε and gives (almost) second-order accuracy.

Max (pointwise) errors on a Shishkin mesh.

ε^2	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
1	2.441×10^{-5}	6.103×10^{-6}	1.526×10^{-6}	3.814×10^{-7}	9.447×10^{-8}
10^{-4}	1.648×10^{-3}	5.227×10^{-4}	1.614×10^{-4}	4.883×10^{-5}	1.453×10^{-5}
10^{-8}	1.692×10^{-3}	5.370×10^{-4}	1.658×10^{-4}	5.023×10^{-5}	1.495×10^{-5}
10^{-12}	1.692×10^{-3}	5.372×10^{-4}	1.660×10^{-4}	5.025×10^{-5}	1.496×10^{-5}

Max (pointwise) errors on a Bakhvalov mesh.

ε^2	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
1	2.441×10^{-5}	6.103×10^{-6}	1.526×10^{-6}	3.814×10^{-7}	9.447×10^{-8}
10^{-4}	7.170×10^{-6}	1.794×10^{-6}	4.486×10^{-7}	1.122×10^{-7}	2.802×10^{-8}
10^{-8}	7.241×10^{-6}	1.814×10^{-6}	4.537×10^{-7}	1.135×10^{-7}	2.840×10^{-8}
10^{-12}	7.245×10^{-6}	1.815×10^{-6}	4.543×10^{-7}	1.137×10^{-7}	2.850×10^{-8}

Are these methods really "robust"?

So, it is now possible to compute a numerical solution so that, for the Shishkin-mesh (for example)

$$\|\mathbf{u} - \mathbf{U}\|_{\infty, \bar{\Omega}^N} \leqslant C(N^{-1} \ln N)^2.$$

But recall that the entries in the matrix depend strongly on the parameter ε. For the "robust" error bound to be meaningful, it must be possible to solve the linear system

$$A_\varepsilon U = F.$$

with efficiency that is independent of ε.

At first glance, this should be possible with a direct method since, for all $\varepsilon \in (0, 1]$, the matrix A_ε has the same structure (sparsity pattern), and is SPD.

A direct solver

Conventional wisdom is that the performance of a direct solver depends only on N and the matrix structure, and so should be independent of ε

Below are the solve times in seconds for a standard solver, CHOLMOD (supernodal sparse Cholesky factorization and update/downdate,[Chen et al., 2009]) on an AMD Opteron 2427, 2200 MHz processor with 32Gb of RAM.

The solve times are erratic, and often quite poor.

ε^2	$N = 2^7$	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
1	0.07	0.39	2.65	18.29	195.87	1680.57
10^{-2}	0.06	0.38	2.66	18.27	196.18	1678.79
10^{-4}	0.06	0.38	2.66	18.39	196.23	1689.43
10^{-6}	0.07	0.97	11.83	89.03	860.62	7515.59
10^{-8}	0.15	1.25	10.62	71.40	478.32	2676.85
10^{-10}	0.19	1.16	8.34	46.22	343.66	1521.52
10^{-12}	0.18	1.10	6.72	36.11	257.12	1166.78

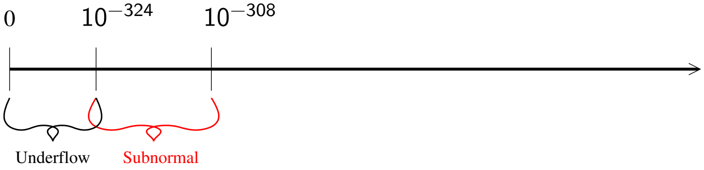
It transpired that the source of difficulty is the presence of subnormal numbers in the calculation of the Cholesky factors...

A direct solver

When the system matrix has small off-diagonal entries, relative to the diagonal in each row, the entries in the factorization scale like ε^k, where k is the distance from the diagonal.

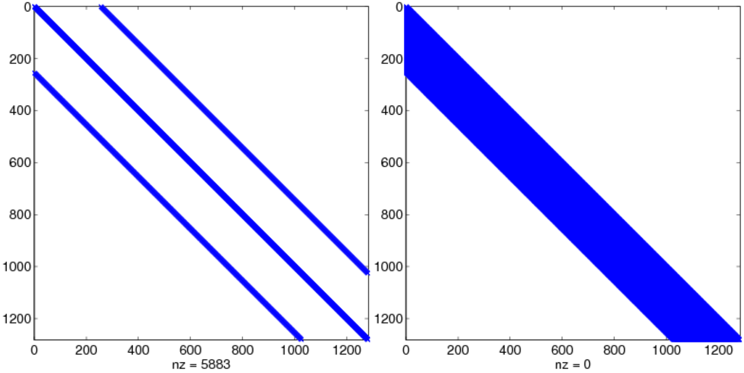
In IEEE double precision numbers are represented as $\pm X \times 2^{Y-1023}$ where 52 bits are used to store the significand, X, 11 bits are used to store the exponent, Y, and the remaining bit stores the sign of X. For "normal" numbers, with $0 < Y < 2047$, X is a binary decimal with leading digit 1. So the smallest **normal** number is $2^{-1022} \approx 10^{-308}$. Smaller numbers are represented by allowing X to have leading zeros (at the cost of precision), giving **subnormal numbers** as small as $2^{-52} \times 2^{-1022} \approx 5 \times 10^{-324}$.

However, most processors do not support subnormal arithmetic in the hardware, and so a much slower software implementation is used.



A direct solver

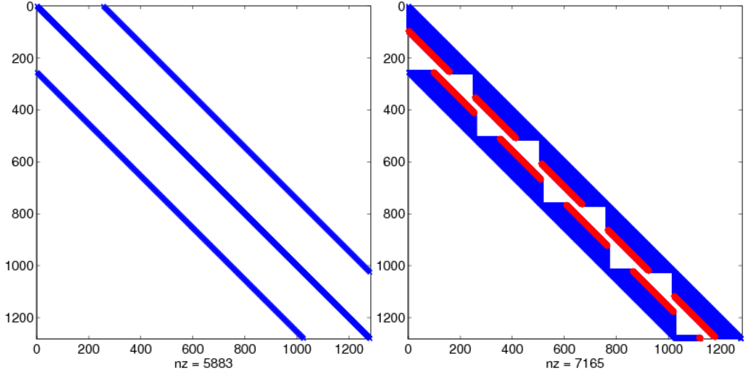
To demonstrate this, the figure below shows the sparcity pattern of a section of the system matrix A_ε (left) and the Cholesky factor (right) for the case $N = 256$ and $\varepsilon = 1$. The subnormal numbers are highlighted in red (there are none).



The system matrix (left) and Cholesky factor (right) for $N = 256$ and $\varepsilon = 1$

A direct solver

Next we show the corresponding figures with $\varepsilon = 10^{-4}$. Again the subnormal numbers are highlighted in red. Moreover, there are many zero entries. Computing these zeros also involves subnormal arithmetic, so they are also expensive.



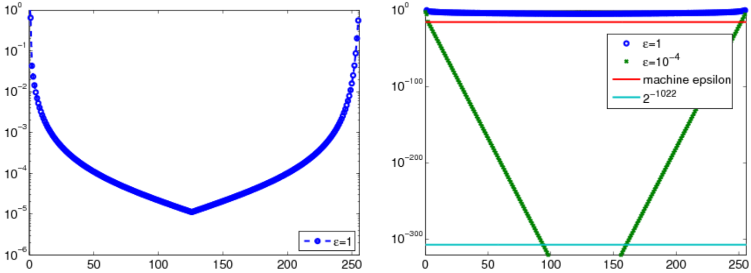
The system matrix (left) and Cholesky factor (right) for $N = 256$ and $\varepsilon = 10^{-4}$.

A direct solver

To show that the entries in the factor do decay exponentially, we show below the maximum entry in the k^{th} diagonal entry.

One the left we show the case $\varepsilon = 1$: entries are smaller in the center of the band, but not extremely so.

On the right are the results for $\varepsilon = 10^{-4}$.



A semi-log plot of the maximum entry in the k^{th} diagonal.

A more detailed analysis is in [Nhan and Madden, 2015a].

Preconditioners

For the remainder of this presentation, we'll motivate a preconditioners suited to boundary layer problems, to be used in conjunction with conjugate gradients. Many details are omitted, including the important topic of appropriately chosen stopping criteria.

For simplicity, we'll consider a *Shishkin mesh*.

. . . .

The underlying scheme is the method of **Conjugate Gradients** (CG). The motivation for using this is given in [MacLachlan and Madden, 2013]. Here we will focus on designing a *preconditioner* for the algorithm. This means, roughly, instead of solving

$$A_\varepsilon u = b,$$

we solve

$$M^{-1} A_\varepsilon u = M^{-1} b.$$

Here, M is a matrix (or proceedure) that approximates A, but is easy to "invert". More generally, a preconditioner can be considered as a function, M of A_ε, where $M(A_\varepsilon) \approx I$.

Preconditioners

Our linear systems are symmetric positive definite, and so **BG** is the method of choice for solving them. If you are unfamilar with BG, then look it up. For now we just need to know

- it is an iterative proceedure;
- it is easy to implement,
- is very efficient for many well-conditioned problems.

However, if the linear system is ill-conditioned, convergence may be slow unless a suitable preconditioning method is employed.

Let $\kappa_2(A_\varepsilon) := \|A_\varepsilon\|_2\|A_\varepsilon^{-1}\|_2$ be the condition number of A_ε associated with the 2-norm, and $u^{(k)}$ be the approximation of u^N after k iterations of the CG algorithm. Then the error at iteration k is bounded as follows [Greenbaum, 1997, Thm. 3.1.1]

$$\|u^N - u^{(k)}\|_{A_\varepsilon} \leqslant 2\left(\frac{\sqrt{\kappa_2(A_\varepsilon)} - 1}{\sqrt{\kappa_2(A_\varepsilon)} + 1}\right)^k \|u^N - u^{(0)}\|_{A_\varepsilon}, \qquad (1)$$

where $\|x\|_{A_\varepsilon} = (x^T A_\varepsilon x)^{1/2}$.

However, for us...

Theorem

The coefficient matrix A_ε of the symmetrized finite-difference discretization on the Shishkin mesh, Ω_S^N, satisfies

$$\kappa_2(A_\varepsilon) \leqslant C(\varepsilon \ln N)^{-2}. \qquad (2)$$

Preconditioners

The following table shows that the above method is not ε-robust. This is verified in the table below.

Solve times in seconds (and iteration counts) for unpreconditioned CG.

ε^2	$N = 128$	$N = 256$	$N = 512$	$N = 1024$
1	0.89 (428)	8.53 (911)	70.81 (1899)	674.11 (4030)
10^{-2}	0.52 (247)	5.16 (566)	46.46 (1251)	462.30 (2701)
10^{-4}	0.47 (222)	3.85 (421)	32.48 (810)	264.81 (1569)
10^{-6}	1.36 (651)	11.91 (1316)	101.74 (2617)	858.04 (5245)
10^{-8}	4.97 (2418)	38.32 (4252)	297.74 (7824)	2532.53 (16112)
10^{-10}	9.99 (4810)	118.15 (12940)	961.80 (26320)	6287.82 (41945)
10^{-12}	13.09 (6306)	152.28 (16593)	1659.64 (43818)	15207.06 (100933)

Preconditioners

Recall that the idea of a preconditioner is that we choose a matrix, M, and use CG to solve $M^{-1}A_\varepsilon u^N = M^{-1}f_N$.

This is equivalent to applying CG directly to solving the symmetric positive definite system

$$(M^{-1/2}A_\varepsilon M^{-1/2})(M^{1/2}u^N) = M^{-1/2}f^N,$$

where $M^{1/2}$ is the principal square root of the symmetric positive definite matrix M. The matrices $M^{-1}A_\varepsilon$ and $M^{-1/2}A_\varepsilon M^{-1/2}$ are similar, so suffices to analyse the condition number of the latter.

We first student the simple diagonal preconditioner

$$D := \mathrm{diag}(a_{11}, a_{22}, \dots, a_{nn}).$$

That is, we take $M = D$ where D is the diagonal matrix whose entries are taken from the main diagonal of A_ε. As we shall show, the condition number of the resulting system is independent of ε.

Theorem

Let $A_D = D^{-1/2} A_\varepsilon D^{-1/2}$. *Then*

$$\kappa_2(A_D) \leqslant C \frac{N^2}{\ln^2 N}. \tag{3}$$

Solve times (in seconds) and iteration counts for diagonal-preconditioned CG.

ε^2	$N = 64$	$N = 128$	$N = 256$	$N = 512$	$N = 1024$
1	0.11 (167)	0.75 (355)	6.91 (756)	59.01 (1603)	516.74 (3394)
10^{-2}	0.06 (92)	0.44 (209)	4.15 (456)	37.05 (1001)	343.10 (2210)
10^{-4}	0.03 (33)	0.15 (67)	1.30 (141)	10.54 (283)	87.74 (568)
10^{-6}	0.03 (33)	0.15 (68)	1.27 (137)	10.23 (275)	90.14 (582)
10^{-8}	0.03 (36)	0.15 (69)	1.27 (138)	10.91 (293)	91.11 (583)
10^{-10}	0.03 (36)	0.15 (69)	1.38 (149)	10.99 (295)	91.23 (584)
10^{-12}	0.03 (36)	0.15 (69)	1.38 (149)	10.98 (295)	91.33 (584)

So this shows that a preconditioner can make the method parameter robust. One can do even better, using an *incomplete Cholesky factorisation*: see [Nhan and Madden, 2015a] for details.

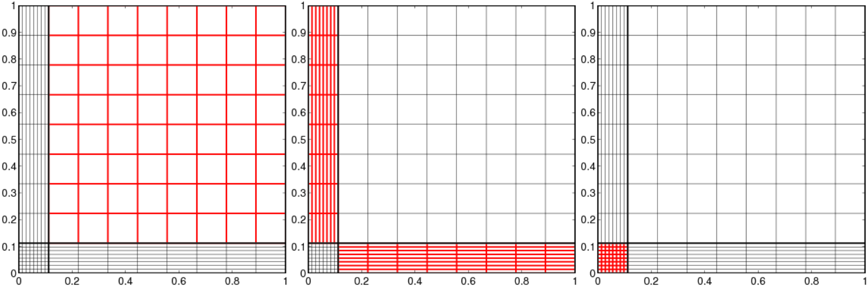
But we will try something more ambitious... based on a preconditioner tailored to our problem.

A boundary layer preconditioner

When designing a suitable preconditioner, it is natural to consider 3 distinct regions:

(i) Interior
(ii) Edges
(iii) Corner

We'll consider each of these in turn.



A boundary layer preconditioner

In the interior,

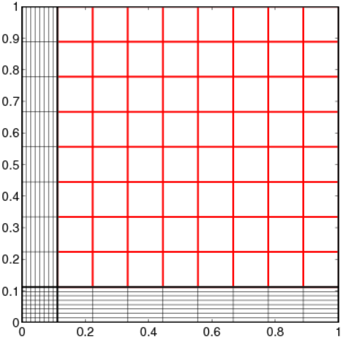
$$h_i = k_j = 2(1-\tau)N^{-1} \approx 2N^{-1}.$$

and

$$\frac{\hbar_i}{k_j} = \frac{\bar{k}_j}{h_i} = 1.$$

The finite difference operator is (roughly)

$$L^N_{ij} \approx \begin{pmatrix} & -\varepsilon^2 & \\ -\varepsilon^2 & 4\varepsilon^2 + 4N^{-2}b_{ij} & -\varepsilon^2 \\ & -\varepsilon^2 & \end{pmatrix}.$$



As expected the reaction term will dominate for small ε:

- **No surprise**: the solution away from the layer closely resembles the solution to the reduced problem;
- If we neglect the $\mathcal{O}(\varepsilon^2)$ terms, the resulting diagonal system is very easy to solve.

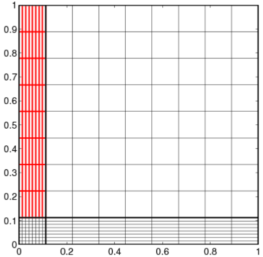
A boundary layer preconditioner

Focusing on just the edge at $x = 0$,

$$h_i = 2\tau N^{-1} = \frac{4 \ln N}{\beta} N^{-1} \varepsilon,$$

$$k_j = 2(1-\tau)N^{-1} \approx 2N^{-1};$$

$$\frac{\hbar_i}{k_j} = \tau = \frac{2 \ln N}{\beta}\varepsilon, \quad \frac{\bar{k}_j}{h_i} \approx \frac{\beta}{2 \ln N}\varepsilon^{-1}.$$



Now the finite difference operator is approximately

$$L_{ij}^N \approx \begin{pmatrix} & -\dfrac{2 \ln N}{\beta}\varepsilon^3 & \\ -\dfrac{\beta}{2 \ln N}\varepsilon & \left[\dfrac{\beta}{\ln N}\varepsilon + \dfrac{4 \ln N}{\beta}\varepsilon^3 + 8b_{ij}\dfrac{\ln N}{\beta}N^2\varepsilon\right] & -\dfrac{\beta}{2 \ln N}\varepsilon \\ & -\dfrac{2 \ln N}{\beta}\varepsilon^3 & \end{pmatrix}.$$

If the $\mathcal{O}(\varepsilon^3)$ terms are neglected, we have a set of tridiagonal problem (associated with a set of ODEs) which are easily solved.

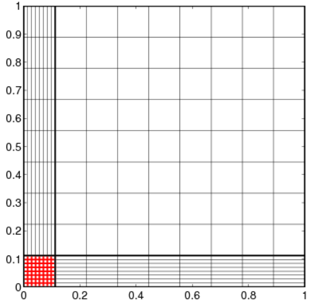
A boundary layer preconditioner

Finally, considering the corner at $(0,0)$,

$$h_i = k_j = 2\tau N^{-1} = \frac{4\ln N}{\beta} N^{-1}\varepsilon,$$

and so

$$\frac{\hbar_i}{k_j} = \frac{\bar{k}_j}{h_i} = 1.$$



And the finite difference operator is

$$L_{ij}^N \approx \left(\begin{array}{ccc} & -\varepsilon^2 & \\ -\varepsilon^2 & \left(4 + 16 b_{ij} \dfrac{\ln^2 N}{\beta^2} N^{-2}\right)\varepsilon^2 & -\varepsilon^2 \\ & -\varepsilon^2 & \end{array} \right).$$

Since the diffusion and reaction terms are of the same order, we can apply a solver which is successful for a non-singularly perturbed problem.

Specifying the preconditioner

Partition system into corners, edge layers, and interior:

$$A = \begin{bmatrix} A_{\mathrm{CC}} & A_{\mathrm{CE}} & 0 \\ A_{\mathrm{EC}} & A_{\mathrm{EE}} & A_{\mathrm{EI}} \\ 0 & A_{\mathrm{IE}} & A_{\mathrm{II}} \end{bmatrix}.$$

We then take the preconditioner to be

$$\text{Take } A_{\mathrm{D}} = \begin{bmatrix} A_{\mathrm{CC}} & 0 & 0 \\ 0 & T_{\mathrm{EE}} & 0 \\ 0 & 0 & D_{\mathrm{II}} \end{bmatrix}.$$

- D_{II} is a diagonal matrix whose entries come from the reaction term in the interior;
- T_{EE} is a tridiagonal matrix (with suitable ordering) associated with the edges.

Specifying the preconditioner

Take h_I to be uniform mesh spacing in interior for 2D problem, $-\varepsilon^2\Delta u + u = f$, with two boundary layers intersecting at corner. Let $\delta_h = (\varepsilon/h_I)^2$.

Expect $\delta_h \ll 1$ for interesting problems

Take $\left(D_{II}\right)_{ii} = h_I^2 b_{ii}$, T_{EE} to discard all lateral connection.

Then,

Theorem ([MacLachlan and Madden, 2013])

$$(1-3\delta_h)V^T A_D V \leqslant V^T A V \leqslant (1+9\delta_h)V^T A_D V$$

for all vectors, V.

Specifying the preconditioner

As mentioned earlier, A_{CC} has the characteristics of a non-singularly perturbed problem. Take M_{CC} to be a preconditioner for A_{CC}, with

$$c_- V_C^T M_{CC} V_C \leqslant V_C^T A_{CC} V_C \leqslant c_+ V_C^T M_{CC} V_C \text{ for all } V_C$$

$$\text{Then } A_M = \begin{bmatrix} M_{CC} & 0 & 0 \\ 0 & T_{EE} & 0 \\ 0 & 0 & D_{II} \end{bmatrix}$$

satisfies

$$\begin{aligned} \min(1-3\delta_h, c_-(1-2\delta_h))V^T A_M V \\ &\leqslant V^T A V \\ &\leqslant \max(1+9\delta_h, c_+(1+2\delta_h))V^T A_M V \end{aligned}$$

for all V. Effective bound is (usually)

$$c_-(1-2\delta_h)V^T A_M V \leqslant V^T A V \leqslant c_+(1+2\delta_h)V^T A_M V$$

Preconditioner results

SP-PCG solve times, N × N Bakhvalov mesh

ε^2	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
10^{-6}	0.05	0.26			
10^{-8}	0.04	0.23	1.24	6.63	28.01
10^{-10}	0.04	0.23	1.24	6.63	28.20
10^{-12}	0.04	0.22	1.24	6.62	28.15

SP-PCG iteration counts

ε^2	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
10^{-6}	6	8			
10^{-8}	6	7	8	8	8
10^{-10}	6	7	8	8	8
10^{-12}	6	7	8	8	8

Preconditioner results

CPU times across algorithms, $N = 2^{12}$

ε^2	CHOLMOD	AMG-PCG	BoxMG-PCG	SP-PCG
1	1680.57	166.10	110.39	—
⋮	⋮	⋮	⋮	⋮
10^{-6}	7515.59	143.10	101.48	—
10^{-8}	2676.85	113.79	101.48	28.01
10^{-10}	1521.52	142.97	101.60	28.20
10^{-12}	1166.78	142.93	101.82	28.15

- As mentioned, the direct solver, `CHOLMOD`, scales poorly.
- A "standard" AMG preconditioning approach for CG appears robust but can be improved upon;
- BoxMG which focuses on maintaining the regular, tensor-product grid structure of the fine mesh, is better.
- The boundary layer preconditioner out-performs all.

Summary

- It can't be assumed that **direct solvers** are "robust" when solving linear systems arising from discretizations of singular perturbed reaction-diffusion problems.
- We've presented a suitable preconditioner that exploits the singularly perturbed nature of the problem.

  There are other preconditioners, which have ready-made implementations that could be used and analysed.

Thank you!

Thank you all for your find attention, and let me express my gratitude for allowing me to take part in this GIAN Workshop. ...

Ireland and *India* are somewhat different in populaton (by a ratio of about 1:3000), in geography, and weather.

However, in one respect India is the **transpose** of Ireland... mathematically,

$$(\text{India}) = (\text{Ireland})^{\mathsf{T}}.$$

Thank you!

Thank you all for your find attention, and let me express my gratitude for allowing me to take part in this GIAN Workshop. ...

Ireland and *India* are somewhat different in populaton (by a ratio of about 1:3000), in geography, and weather.

However, in one respect India is the **transpose** of Ireland... mathematically,

$$(\mathrm{India}) = (\mathrm{Ireland})^{\mathsf{T}}.$$

Thank you!

Thank you all for your find attention, and let me express my gratitude for allowing me to take part in this GIAN Workshop. ...

Ireland and *India* are somewhat different in populaton (by a ratio of about 1:3000), in geography, and weather.

However, in one respect India is the **transpose** of Ireland... mathematically,

$$(\mathrm{India}) = (\mathrm{Ireland})^{\mathsf{T}}.$$

Thanks you!
Míle Buiochas!

धन्यवाद

References

Chen, Y., Davis, T., and Hager, W. (2009).
Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate.
ACM Trans. Math. Software, 35(3).

Clavero, C., Gracia, J. L., and O'Riordan, E. (2005).
A parameter robust numerical method for a two dimensional reaction-diffusion problem.
Math. Comp., 74(252):1743–1758.

Greenbaum, A. (1997).
Iterative methods for solving linear systems, volume 17 of *Frontiers in Applied Mathematics*.
Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA.

References

Kellogg, R. B., Linss, T., and Stynes, M. (2008).
A finite difference method on layer-adapted meshes for an elliptic reaction-diffusion system in two dimensions.
Math. Comp., 77(264):2085–2096.

MacLachlan, S. and Madden, N. (2013).
Robust solution of singularly perturbed problems using multigrid methods.
SIAM J. Sci. Comput., 35(5):A2225–A2254.

Nhan, T. A. and Madden, N. (2015a).
An analysis of simple preconditioners for a singularly perturbed problem on a layer-adapted mesh.
(submitted for publication).

References

Nhan, T. A. and Madden, N. (2015b).

Cholesky factorisation of linear systems coming from finite difference approximations of singularly perturbed problems.

In *BAIL 2014–Boundary and Interior Layers, Computational and Asymptotic Methods*, Lect. Notes Comput. Sci. Eng., pages 209–220. Springer International Publishing.