Spectral Schur complement techniques for symmetric eigenvalue problems

Vassilis Kalantzis

Computer Science and Engineering Department
University of Minnesota - Twin Cities, USA

Oral Preliminary Exam, 05-27-2016
Acknowledgments

- To my advisor Prof. Yousef Saad
- My deepest gratitudes to the exam committee members
- Special thanks to Prof. Bernardo Cockburn for our discussions regarding the presented work
- This work is partly based on computations performed at the Minnesota Supercomputing Institute
- Work supported by NSF and DOE (DE-SC0008877)
Contents

1 Introduction

2 The domain decomposition framework

3 Solving the spectral Schur complement eigenvalue problem
   - Computing more than one eigenpairs

4 Experiments

5 Conclusion
The symmetric eigenvalue problem

\[ Ax^{(i)} = \lambda_i x^{(i)}, \quad i = 1, \ldots, n, \]

where \( A \in \mathbb{R}^{n \times n} \) is symmetric, \( x^{(i)} \in \mathbb{R}^n \) and \( \lambda_i \in \mathbb{R} \). A pair \((\lambda_i, x^{(i)})\) is an eigenpair of \( A \).

Focus

1. Compute, up to a tolerance \( \text{tol} \), all eigenpairs \((\lambda_i, x^{(i)})\) located inside the interval \([\alpha, \beta]\) where \( \alpha, \beta \in \mathbb{R} \) and \( \lambda_1 \leq \alpha, \beta \leq \lambda_n \).
2. Given a shift \( \zeta \in \mathbb{R} \), find the \( k \) \((\lambda, x)\) pairs closest to \( \zeta \).

How to compute a few eigenpairs of \( A \)?

1. Typically: Lanczos on \( A \) or \( \rho(A - \sigma I) \), \( \sigma \in \mathbb{C} \)
2. In this talk we consider Schur complement eigenvalue solvers
Block matrix representation

Consider matrix $A$ written as

$$A = \begin{pmatrix} B & E \\ E^T & C \end{pmatrix}$$

where $B \in \mathbb{R}^{d \times d}$, $E \in \mathbb{R}^{d \times s}$ and $C \in \mathbb{R}^{s \times s}$, $n = d + s$.

Let $x = [u^T, y^T]^T$, $u \in \mathbb{R}^d$, $y \in \mathbb{R}^s$, assume $\lambda \notin \Lambda(B)$

Solution of $(A - \lambda I)x = 0$ is transformed to:

1. Solve $(B - \lambda I)u = -Ey$
2. Solve $S(\lambda)y = 0$

where

$$S(\lambda) = C - \lambda I - E^T(B - \lambda I)^{-1}E,$$

is the spectral Schur complement matrix.
Some references on spectral Schur complements

- Component Mode Synthesis and Automated Multi-Level Substructuring (AMLS) [BeLe] for the analysis of frequency response
  - Substructuring techniques (domain decomposition)
  - Approximate $y$ by linearizing $S(\lambda) \rightarrow$ Generalized eig. problem with pencil $(S(0), S'(0))$
  - Approximate $u$ by approximating $-(B - \lambda I)^{-1}Ey$
  - Perform a Rayleigh-Ritz projection
  - AMLS is a multilevel extension of CMS (combined with mode truncation)
  - Typically: fast computation but modest accuracy

- Other work: Abramov and Chishov (spectral Schur complements), Lui (Kron’s method),...
### Hybridized Raviart-Thomas approximation of 2nd order elliptic eigenvalue problems [Coc]

- The original eigenvalue problem is condensed by hybridization.
- Result: transformation to a non-linear, but smaller eigenvalue problem.
- Lower energy modes can be recovered accurately.
- Solving the non-linear eigenvalue problem:
  - First solve a perturbed (linearized) version to obtain accurate approximations.
  - Exploit Newton’s method (or RQI) to refine each approximate eigenvalue.

### HDG approximation of 2nd order elliptic eigenvalue problems [Gop]

- Similar framework with the Raviart-Thomas approximation.
- Covers a narrower effective spectrum but can be benefited by postprocessing.
In this talk we concentrate on Schur complement eigenvalue solvers from a **domain decomposition** viewpoint and contribute:

1. A numerical scheme based on Newton root-finding procedures
2. An extension of current understanding regarding Schur complement eigenvalue solvers
3. An implementation of the proposed scheme in distributed computing environments
Partitioning of the domain (Metis, Scotch,...)

- **Goal:** Partition a discretized domain in $p$ subdomains
- **Two main approaches to partition a graph:**
  - Edge-separator (vertex-based partitioning)
  - Vertex-separator (edge-based partitioning)
- **After partitioning, three types of unknowns:**
  - interior unknowns
  - local interface unknowns
  - external interface unknowns
- **The $i$’th subdomain has** $d_i$ interior unknowns and $s_i$ local interface unknowns

**Figure:** An edge-separator
The local viewpoint in each subdomain

Equation \((A - \lambda I)x = 0\) can be written locally as

\[
\begin{pmatrix}
B_i - \lambda I & \hat{E}_i \\
\hat{E}_i^T & C_i - \lambda I
\end{pmatrix}
\begin{pmatrix}
u_i \\
y_i
\end{pmatrix} + 
\sum_{j \in N_i} E_{ij} y_j = 0.
\]

- \(B_i \in \mathbb{R}^{d_i \times d_i}\): coupling between interior variables
- \(\hat{E}_i \in \mathbb{R}^{d_i \times s_i}\): coupling between interior/loc. interface variables
- \(C_i \in \mathbb{R}^{s_i \times s_i}\): coupling between local interface variables
- \(N_i\): set of indices for subdomains that are neighbors to the \(i'\)th subdomain
- \(E_{ij} \in \mathbb{R}^{s_i \times s_j}\): coupling among subdomains \(i\) and \(j\)

**Figure:** Local viewpoint
Global reordering of the eigenvalue problem

Stack interior variables $u_1, u_2, \ldots, u_p$ into $u$, then interface variables $y_1, y_2, \ldots, y_p$ to $y$, and reorder $A$ so that interior variables are numbered before interface ones:

$$
\begin{pmatrix}
B_1 & E_1 \\
B_2 & E_2 \\
\vdots & \vdots \\
B_p & E_p \\
E_1^T & E_2^T & \cdots & E_p^T \\
\end{pmatrix}
\begin{pmatrix}
u_1 \\
u_2 \\
\vdots \\
u_p \\
y \\
\end{pmatrix}
=
\lambda
\begin{pmatrix}
u_1 \\
u_2 \\
\vdots \\
u_p \\
y \\
\end{pmatrix}
$$

where $E_i = [0_{d_i}, \ell_i, \hat{E}_i, 0_{d_i}, \nu_i]$, with $\ell_i = \sum_{j=1}^{j<i} s_j$, $\nu_i = \sum_{j>i}^{j=p} s_j$. 
Example with $p = 4$ (different color $\rightarrow$ different subdomain):

Notation: write as

$$A = \begin{pmatrix} B & E \\ E^\top & C \end{pmatrix}$$
Eliminating the $u_i$'s we get

$$S(\lambda)y = \begin{pmatrix} S_1(\lambda) & E_{12} & \cdots & E_{1p} \\ E_{21} & S_2(\lambda) & \cdots & E_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ E_{p1} & E_{p2} & \cdots & S_p(\lambda) \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_p \end{pmatrix} = 0$$

with $S_i(\lambda) = C_i - \lambda I - E_i^T (B_i - \lambda I)^{-1} E_i$.

Interface problem (non-linear):

$$S(\lambda)y = 0$$

Top parts can be recovered as $u_i = -(B_i - \lambda I)^{-1} E_i y$
Contents

1 Introduction

2 The domain decomposition framework

3 Solving the spectral Schur complement eigenvalue problem
   - Computing more than one eigenpairs

4 Experiments

5 Conclusion
Spectral Schur complement revisited

Our viewpoint:

Find \( \sigma \in \mathbb{R} \) such that

\[
\mu(\sigma) = 0,
\]

where \( \mu(\sigma) \) denotes the smallest (\(|\cdot|\)) eigenvalue of \( S(\sigma) \).

We can treat \( \mu(\sigma) \) as a function \( \rightarrow \) root-finding problem

Bisection: slow. Newton: fast, but needs the derivative

Other functions are possible, e.g. \( \mu(\sigma) = \det[S(\sigma)] \) (Lui) \( \rightarrow \) not very practical

Eigenbranches

For each \( \sigma \): \( S(\sigma)y^{(i)}(\sigma) = \mu_i(\sigma)y^{(i)}(\sigma), \ i = 1, \ldots, s \) (in sorted ascending algebraic order)

Each \( \mu_i(\sigma) \) is a function of \( \sigma \) (we call it an eigenbranch)

The eigenvalues of \( B \) are the poles of the eigenbranches
Eigenbranches – illustration

An example for a discretized 2D Laplacian

Eigs 1 through 9 in pole-interval [0.0000 0.1100]
Basic algorithm - Newton’s scheme

Derivative of $\mu_i(\sigma)$

- Eigenbranch $\mu_i(\sigma)$ is analytic for any $\sigma \notin \Lambda(B)$ with

$$
\frac{d\mu_i(\sigma)}{d\sigma} = \frac{(S'(\sigma)y^{(i)}(\sigma), y^{(i)}(\sigma))}{(y^{(i)}(\sigma), y^{(i)}(\sigma))} = -1 - \frac{\|(B - \sigma I)^{-1}Ey^{(i)}(\sigma)\|_2^2}{\|y^{(i)}(\sigma)\|_2^2}.
$$

where

$$
S'(\sigma) = -I - E^\top (B - \sigma I)^{-2}E.
$$

Algorithm 3.1

1: Select $\sigma$, tol
2: repeat
3: Compute $\mu(\sigma) =$ Smallest eigenvalue in modulus of $S(\sigma)$
4: along with associated unit eigenvector $y(\sigma)$
5: Set $\eta := \|(B - \sigma I)^{-1}Ey(\sigma)\|_2$, and $\sigma := \sigma + \mu(\sigma)/(1 + \eta^2)$
6: until $|\mu(\sigma)| \leq \text{tol}$
“Chasing” an eigenbranch using Newton’s iteration

Figure: Computing the algebraically smallest eigenvalue of a 2D Laplacian
Practical details

Computing \((\mu(\sigma), y(\sigma))\) – the main computational kernel!

- Perfect setting for a form of (inexact) inverse iteration (MINRES, GMRES)
- Alternatively, the Lanczos algorithm can be used
- Note that \(\mu'(\sigma)\) is upper-bounded by \(-1\) and its computation is entirely local across the subdomains

Residual of the approximation + connection with RQ

It can be shown that

\[
\|(A - \sigma I)\hat{x}(\sigma)\| = |\mu(\sigma)|
\]

where \(\hat{x}(\sigma) = [-(B - \sigma I)^{-1}Ey(\sigma); y(\sigma)]\) is the approximate eigenvector, and the Newton update also is the Rayleigh quotient,

\[
\sigma = \rho(A, \hat{x}(\sigma)).
\]
Eigenbranches across the poles (I)

Let \((\theta_j, v_j), \ j = 1, \ldots, d\) be the eigenpairs of \(B\). We re-write:

\[
S(\sigma) = C - \sigma I - E^T (B - \sigma I) E = C - \sigma I - \sum_{j=1}^{d} \frac{w_j w_j^T}{\theta_j - \sigma}, \quad \text{with} \quad w_j \equiv E^T v_j.
\]

Analysis (assume \(\theta_k, \ 1 \leq k \leq d, \) is a simple pole)

We have

\[
\mu_j(\sigma) = \min_{\mathcal{U}_j, \dim(\mathcal{U}_j)=j} \max_{r \in \mathcal{U}_j} \rho(\sigma, r), \quad \rho(\sigma, r) = \frac{(S(\sigma)r, r)}{(r, r)}.
\]

- As \(\sigma \to \theta_k^-\), one eigenbranch \((\mu_1(\sigma))\) will descend to \(-\infty\)
- However, if \(r \perp w_k \to \) eigenbranches \(\mu_2(\sigma), \ldots, \mu_s(\sigma)\) are well defined and converge to the eigenvalues of an operator which is orthogonal to \(w_k\)
Figure: Behavior of a few eigenbranches as they cross a pole. Red circles: eigenvalues of $A$ (roots). Dashed lines: eigenvalues of $B$ (poles) Left: Eigenbranches $\mu_{26}, \ldots, \mu_{32}$ in an interval not containing their poles. Right: Eigenbranches $\mu_1(\sigma)$ and $\mu_2(\sigma)$ as $\sigma$ approaches the pole of $\mu_1(\sigma)$. 
A branch-hopping scheme

Algorithm 3.2 – “Chasing” more than one eigenvalues

1. Given $a, b$. Select $\sigma = a$
2. while $\sigma < b$ do
3. Compute $S(\sigma) \mu(\sigma) = \mu(\sigma) y(\sigma)$
4. if $|\mu(\sigma)| \leq \text{tol}$ then
5. Obtain $\mu(\sigma) =$ smallest positive eigenvalue of $S(\sigma)$
6. end if
7. Compute derivative and update $\sigma$ as in Algorithm 3.1
8. end while

Algorithm 3.2 assumes that we move rightwards; can be trivially modified for the case where we move leftwards.
Figure: Computing the few smallest eigenvalues for the same 2D Laplacian
Figure: The shape of eigenbranches $\mu_1(\sigma), \ldots, \mu_9(\sigma)$. Left: $p = 4$. Right: $p = 16$. 

Effects of $p$ in convergence – illustration
Effects of $p$ in convergence – eigenvalues of $B$ as $p$ varies

Figure: Monitoring the few smallest eigenvalues of $B$ in a fixed interval as $p$ varies.
A few more considerations

**Multiple eigenvalues**

Multiple eigenvalue $\lambda \rightarrow$ multiple (with the same degree of multiplicity) zero eigenvalue of $S(\lambda)$.

**Misconvergence**

It is possible to misconverge when the root of the eigenbranch is close to its pole. To deal with this (potential) issue:

- Standard tool: inertia
- Have “better linear” eigenbranches
- Post-process new $\sigma$ by inverse iteration

**Figure**: An example of misconvergence.
Experiments

Computational system

- Tests performed on Itasca Linux cluster @ MSI
- Each node is a two-socket, quad-core 2.8 GHz Intel Xeon X5560 “Nehalem EP” with 24 GB of system memory
- Interconnection: 40-gigabit QDR InfiniBand (IB)

Test matrices

- Tests on 3D dicretized Laplacians (7pt. st. – FD) lying on the unit cube
- Dirichlet boundary conditions
- We use $n_x$, $n_y$, $n_z$ to denote the number of nodes along each dimension
- $\text{tol}$ for each eigenpair set to $1e-8$
- *Software*: PETSc C++ (METIS+CHOLMOD)
Qualitative behavior of Alg 3.2 on a few 3D Laplacians

<table>
<thead>
<tr>
<th>( \alpha, \beta ) := [0, 0.5]</th>
<th>( \alpha, \beta ) := [2, 2.2]</th>
<th>( \alpha, \beta ) := [4.1, 4.2]</th>
</tr>
</thead>
<tbody>
<tr>
<td># Eigvls</td>
<td>It</td>
<td># Eigvls</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----</td>
<td>-------------------</td>
</tr>
<tr>
<td>( n = 21 \times 20 \times 19 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td># of subdomains ( (p) )</td>
<td>2</td>
<td>60</td>
</tr>
<tr>
<td>4</td>
<td>35</td>
<td>43</td>
</tr>
<tr>
<td>8</td>
<td>35</td>
<td>116</td>
</tr>
<tr>
<td>16</td>
<td>39</td>
<td>96</td>
</tr>
<tr>
<td>( n = 41 \times 20 \times 19 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td># of subdomains ( (p) )</td>
<td>2</td>
<td>210</td>
</tr>
<tr>
<td>4</td>
<td>72</td>
<td>170</td>
</tr>
<tr>
<td>8</td>
<td>154</td>
<td>273</td>
</tr>
<tr>
<td>16</td>
<td>138</td>
<td>241</td>
</tr>
<tr>
<td>( n = 41 \times 40 \times 20 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td># of subdomains ( (p) )</td>
<td>2</td>
<td>385</td>
</tr>
<tr>
<td>4</td>
<td>160</td>
<td>354</td>
</tr>
<tr>
<td>8</td>
<td>296</td>
<td>502</td>
</tr>
<tr>
<td>16</td>
<td>270</td>
<td>451</td>
</tr>
</tbody>
</table>
Figure: Rel. res. for a few consecutive eigenvalues. Left: $n_x = 21$, $n_y = 20$ and $n_z = 19$. Right: $n_x = 41$, $n_y = 20$ and $n_z = 19$. 
Combining Alg. 3.1 with inverse iteration

![Graphs showing convergence behavior](image)

**Figure:** Convergence behavior for computing the eigenpair closest to $\zeta$ when combining inverse iteration and Newton’s method. Left: $\zeta = 0.0$. Right: $\zeta = 3.0$. 
Experiments in distributed computing environments

Performing the Matrix-Vector product with $S(\zeta)$

1. MV multiplication with $C - \sigma I$ (non-local),
2. MV multiplication with $E$ and $E^T$ (local),
3. system solution with $B - \sigma I$ (local),

Comparisons against residual inverse iteration (RI)

- RI: Similar to inverse iteration but with fixed inner tolerance
- Computation of $(\mu(\sigma), y(\sigma))$ is performed inexactly
- We use MINRES as the iterative solver
- Each subdomain is handled by a separate MPI process
Computing \( k = 1 \) and \( k = 5 \) eigenvalues next to \( \zeta \) for a \( 41 \times 40 \times 39 \) 3D Laplacian. For the case where \( \zeta \neq 0 \), the starting shift for each particular eigenpair computation in Newton’s scheme was provided by first performing three steps of inexact Inverse Iteration. Times are listed in seconds.

<table>
<thead>
<tr>
<th>((p, k))</th>
<th>(s)</th>
<th>(\zeta = 0.0)</th>
<th>(\zeta = 0.1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>((16,1))</td>
<td>15423</td>
<td>0.21 3 0.10</td>
<td>1.07 4 1.32</td>
</tr>
<tr>
<td>((16,5))</td>
<td>- -</td>
<td>1.39 15 0.62</td>
<td>5.85 19 7.77</td>
</tr>
<tr>
<td>((32,1))</td>
<td>20037</td>
<td>0.06 3 0.03</td>
<td>0.27 2 0.90</td>
</tr>
<tr>
<td>((32,5))</td>
<td>- -</td>
<td>0.32 14 0.19</td>
<td>1.52 14 4.86</td>
</tr>
<tr>
<td>((64,1))</td>
<td>24789</td>
<td>0.09 3 0.04</td>
<td>0.14 3 0.66</td>
</tr>
<tr>
<td>((64,5))</td>
<td>- -</td>
<td>0.44 14 0.21</td>
<td>1.01 15 3.51</td>
</tr>
</tbody>
</table>
Table: Computing \( k = 1 \) and \( k = 5 \) eigenvalues next to \( \zeta \) for a \( 71 \times 70 \times 69 \) 3D Laplacian. For the case where \( \zeta \neq 0 \), the starting shift for each particular eigenpair computation in Newton’s scheme was provided by first performing three steps of inexact Inverse Iteration. Times are listed in seconds.

\[ n = 71 \times 70 \times 69 \]

<table>
<thead>
<tr>
<th>((p, k))</th>
<th>(s)</th>
<th>(\zeta = 0.0)</th>
<th>(\zeta = 0.1) (137)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(64,1)</td>
<td>83358</td>
<td>0.80 3 0.61</td>
<td>15.4 2 15.9</td>
</tr>
<tr>
<td>(64,5)</td>
<td>- -</td>
<td>4.20 14 3.22</td>
<td>80.4 10 79.9</td>
</tr>
<tr>
<td>(128,1)</td>
<td>108508</td>
<td>0.19 3 0.32</td>
<td>3.12 2 8.41</td>
</tr>
<tr>
<td>(128,5)</td>
<td>- -</td>
<td>1.25 14 1.71</td>
<td>15.1 10 38.5</td>
</tr>
<tr>
<td>(256,1)</td>
<td>136159</td>
<td>0.10 3 0.27</td>
<td>5.99 2 12.7</td>
</tr>
<tr>
<td>(256,5)</td>
<td>- -</td>
<td>0.68 13 1.45</td>
<td>25.3 10 51.7</td>
</tr>
</tbody>
</table>
Table: Computing $k = 1$ and $k = 5$ eigenvalues next to $\zeta$ for a $101 \times 100 \times 99$ 3D Laplacian. For the case where $\zeta \neq 0$, the starting shift for each particular eigenpair computation in Newton’s scheme was provided by first performing three steps of inexact Inverse Iteration. Times are listed in seconds.

<table>
<thead>
<tr>
<th>$(p, k)$</th>
<th>$s$</th>
<th>$\zeta = 0.0$</th>
<th></th>
<th>$\zeta = 0.1$ (439)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$T_{NT}$</td>
<td>It</td>
<td>$T_{RI}$</td>
<td>$T_{NT}$</td>
</tr>
<tr>
<td>(128,1)</td>
<td>230849</td>
<td>2.73</td>
<td>3</td>
<td>2.02</td>
<td>48.1</td>
</tr>
<tr>
<td>(128,5)</td>
<td>- -</td>
<td>13.2</td>
<td>15</td>
<td>10.3</td>
<td>233.2</td>
</tr>
<tr>
<td>(256,1)</td>
<td>293626</td>
<td>1.10</td>
<td>3</td>
<td>1.61</td>
<td>23.4</td>
</tr>
<tr>
<td>(256,5)</td>
<td>- -</td>
<td>5.80</td>
<td>14</td>
<td>8.32</td>
<td>129.2</td>
</tr>
<tr>
<td>(512,1)</td>
<td>369663</td>
<td>0.62</td>
<td>2</td>
<td>0.99</td>
<td>32.4</td>
</tr>
<tr>
<td>(512,5)</td>
<td>- -</td>
<td>3.01</td>
<td>12</td>
<td>5.71</td>
<td>168.9</td>
</tr>
</tbody>
</table>
For the case $\zeta = 0.0$, $k = 5$:

1. **Time to factorize $B - \sigma I$**
2. **Time spent on solving linear systems**
3. **Avg. of MINRES iters per eigenpair**
A comparison with ARPACK

Table: Computing $k = 1$ and $k = 5$ eigenvalues next to $\zeta$ with the proposed Newton scheme and ARPACK. The discretization selected as $n_x = 71$, $n_y = 70$, and $n_z = 69$. Times are listed in seconds. Newton’s scheme is using 8 cores.

<table>
<thead>
<tr>
<th>$(p, k)$</th>
<th>$\zeta = 0.0$</th>
<th>$\zeta = 0.1$ (137)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$T_{NT}$</td>
<td>$T_{ARP}$</td>
</tr>
<tr>
<td>(64,1)</td>
<td>5.5</td>
<td>35.4</td>
</tr>
<tr>
<td>(128,1)</td>
<td>3.4</td>
<td>–</td>
</tr>
<tr>
<td>(256,1)</td>
<td>5.3</td>
<td>–</td>
</tr>
<tr>
<td>(64,5)</td>
<td>28.3</td>
<td>94.1</td>
</tr>
<tr>
<td>(128,5)</td>
<td>15.3</td>
<td>–</td>
</tr>
<tr>
<td>(256,5)</td>
<td>25.9</td>
<td>–</td>
</tr>
</tbody>
</table>
Contents

1. Introduction

2. The domain decomposition framework

3. Solving the spectral Schur complement eigenvalue problem
   - Computing more than one eigenpairs

4. Experiments

5. Conclusion
Conclusion

**In this talk**

- We presented a Schur complement eigenvalue solver that focuses solely on the interface problem region.
- Eigenvalue branches of the SSC $\rightarrow$ Newton’s method.
- Potential for 2-D level parallelism.
- Ultimately, $k$ eigenpairs are computed at the cost of one.

**Considerations**

- Use subspace acceleration.
- Solution of generalized eigenvalue problems.
- Combine with an efficient mechanism to obtain initial guesses (AMLS?).
- Use Newton’s method on a higher-order approximation of $S(\lambda)$. 
Figure: An example of misconvergence.
Table: Computing \( k = 1 \) and \( k = 5 \) eigenvalues next to \( \zeta \) for a \( 801 \times 800 \) 2D Laplacian. For the case where \( \zeta \neq 0 \), the starting shift for each particular eigenpair computation in Newton’s scheme was provided by first performing three steps of inexact Inverse Iteration. Times are listed in seconds.

\[
\begin{array}{ccccccc}
\text{n} & = & 801 \times 800 \\

\begin{array}{|c|c|c|c|c|c|}
\hline
(p, k) & s & \zeta = 0.0 & \zeta = 0.01 (488) \\
\hline
(64,1) & 24945 & 1.09 & 3 & 1.25 & 37.4 & 2 & 156.4 \\
(64,5) & - & 5.95 & 15 & 6.98 & 198.7 & 12 & 775.1 \\
(128,1) & 36611 & 0.27 & 2 & 0.67 & 24.0 & 2 & 75.4 \\
(128,5) & - & 1.31 & 9 & 3.82 & 125.0 & 11 & 382.1 \\
(256,1) & 52319 & 0.22 & 2 & 0.48 & 11.2 & 2 & 44.9 \\
(256,5) & - & 1.59 & 9 & 2.73 & 61.3 & 10 & 231.6 \\
\hline
\end{array}
\]
Table: Computing $k = 1$ and $k = 5$ eigenvalues next to $\zeta$ for a $1001 \times 1000$ 2D Laplacian. For the case where $\zeta \neq 0$, the starting shift for each particular eigenpair computation in Newton’s scheme was provided by first performing three steps of inexact Inverse Iteration. Times are listed in seconds.

\begin{align*}
\text{n} &= 1001 \times 1000 \\

\begin{array}{|c|c|c|c|c|c|c|}
\hline
(p, k) & s & T_{NT} & \text{lt} & T_{RI} & T_{NT} & \text{lt} & T_{RI} \\
\hline
(128, 1) & 46073 & 0.42 & 3 & 1.03 & 95.3 & 2 & 102.1 \\
(128, 5) & - & 2.84 & 15 & 5.33 & 482.7 & 10 & 532.1 \\
(256, 1) & 65780 & 0.27 & 2 & 0.64 & 54.2 & 2 & 73.4 \\
(256, 5) & - & 1.35 & 10 & 3.32 & 281.3 & 9 & 381.4 \\
(512, 1) & 93440 & 0.25 & 2 & 0.58 & 49.4 & 2 & 58.1 \\
(512, 5) & - & 1.42 & 10 & 3.21 & 256.7 & 10 & 312.8 \\
\hline
\end{array}
\end{align*}
Let $\theta_k^-$ be a simple eigenvalue of $B$ and define

$$S_k(\sigma) = C - \sigma I - \sum_{j=1, j\neq k}^d \frac{w_j w_j^T}{\theta_j - \sigma}, \quad \rho_k(\sigma, r) = \frac{(S_k(\sigma)r, r)}{(r, r)}$$

In the following we assume that $\sigma \in [\sigma_0, \theta_k]$ where $[\sigma_0, \theta_k)$ contains no eigenvalues of $B$.

As $\sigma \to \theta_k^-$, for any $j > 1$, we have

$$\mu_j(\sigma) = \rho(\sigma, y^{(j)}(\sigma)) = \rho_k(\sigma, y^{(j)}(\sigma)) - \frac{(y^{(j)}(\sigma)^T w_k)^2}{\theta_k - \sigma},$$

and since $\rho_k(\sigma, y^{(j)}(\sigma))$ is bounded, we get $\lim_{\sigma \to \theta_k^-} w_k^T y^{(j)}(\sigma) = 0$. 
Let

$$P_k = I - w_k w_k^T / (w_k^T w_k) = I - \hat{w}_k \hat{w}_k^T.$$ 

Multiplying $S(\sigma)y^{(j)}(\sigma) = \mu_j(\sigma)y^{(j)}(\sigma)$ by $P_k$ from the left yields

$$P_k S_k(\sigma)P_k y^{(j)}(\sigma) - \mu_j(\sigma)P_k y^{(j)}(\sigma) = -P_k S_k(\sigma)(\hat{w}_k^T y^{(j)}(\sigma))\hat{w}_k.$$

Therefore, the eigenpair $(\mu_j(\sigma), P_k y^{(j)}(\sigma))$ converges to an eigenpair of $P_k S_k(\theta_k)P_k$, which is a trivial extension of $S_k,|_1(\sigma) = [P_k S_k(\sigma)P_k]_{w_k \perp}$. 