A refined Lanczos method for computing eigenvalues and eigenvectors of unsymmetric matrices

Jean Christophe Tremblay and Tucker Carrington

Chemistry Department
Queen's University

23 août 2007

We want to solve the eigenvalue problem

$$\mathbf{G}\mathbf{X}_{(R)} = \mathbf{X}_{(\mathbf{R})}\mathbf{\Lambda}$$

- We seek, first and foremost, an eigensolver that minimizes the amount of memory required to solve the problem.
- For symmetric matrices the Lanczos algorithm without re-orthogonalization is a good choice.
  - only two vectors are stored in memory

Eigensolvers for unsymmetric matrices

For unsymmetric matrices the established methods are :

- variants of the Arnoldi algorithm
- the unsymmetric (or two-sided) Lanczos algorithm (ULA)

In this talk I present a method that in some sense is better than both.

What are the deficiencies of Arnoldi ?

- It requires storing many vectors in memory. Every Arnoldi vector is orthogonalized against *all* of the previous computed Arnoldi vectors.
- The CPU cost of the orthogonalization also increases with the number of iterations.
- These problems are to some extent mitigated by using the implicitly restarted Arnoldi (IRA) technique (ARPACK). Nevertheless, for large matrices it is not possible to use ARPACK on a computer that does not have a lot of memory. ARPACK is best if one wants extremal eigenvalues.

What are the deficiencies of the ULA ?

In its primitive form it seldom yields accurate eigenvalues.

Re-biorthogonalization improves the accuracy but negates the method's memory advantage.

The new method (RULE) in a nutshell

- No re-biorthogonalization of the Lanczos vectors ; regardless of the number of required iterations only 4 Lanczos vectors are stored in memory
- The method may be used to compute either extremal or interior eigenvalues
- We use approximate right and left eigenvectors obtained from the ULA to build a projected matrix which we diagonalize

The take-home message

- The RULE makes it possible to extract accurate eigenvalues from large Krylov spaces.
- It is a good alternative to ARPACK
  - if the matrix for which eigenvalues are desired is so large that the memory ARPACK requires exceeds that of the computer
  - if one wishes interior eigenvalues.

The ULA

Two sets of Lanczos vectors are obtained from two three-term recurrence relations,

$$\begin{aligned} \mathbf{G}\mathbf{V}_m &= \mathbf{V}_m\mathbf{T}_m + \rho_{m+1}\mathbf{v}_{m+1}\mathbf{e}_m^* \\ \mathbf{G}^*\mathbf{W}_m &= \mathbf{W}_m T_m^* + \gamma_{m+1}^*\mathbf{w}_{m+1}\mathbf{e}_m^* \quad , \end{aligned}$$

$$\delta_k = \mathbf{w}_k^*\mathbf{v}_k = 1$$

The scalars α_k, ρ_k, and γ_k are elements of the tridiagonal matrix $\mathbf{T}_m = \mathbf{W}_m^* \mathbf{G} \mathbf{V}_m$,

$$\mathbf{T}_m = \begin{pmatrix} \alpha_1 & \gamma_2 & & & \\ \rho_2 & \alpha_2 & \gamma_3 & & \\ & \rho_3 & \alpha_3 & \ddots & \\ & & \ddots & \ddots & \gamma_m \\ & & & \rho_m & \alpha_m \end{pmatrix} .$$

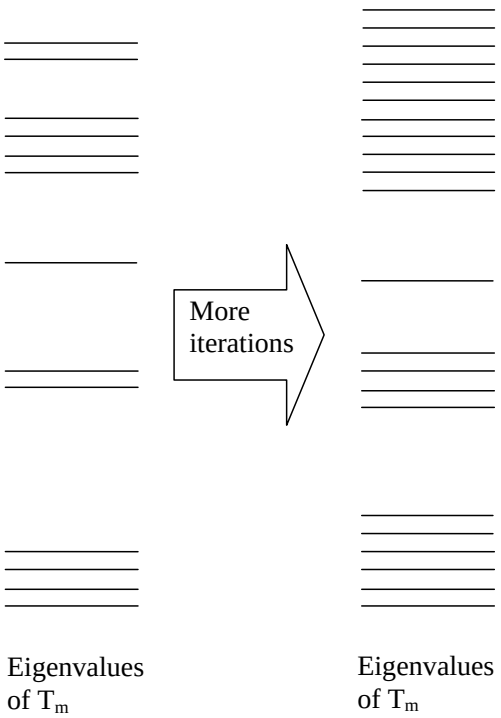
Eigenvalues of the matrix $\mathbf{T}_m$ are computed by solving

$$\mathbf{T}_m \mathbf{Z} = \mathbf{Z} \mathbf{\Theta}_m \quad .$$

The ULA does not work !

Roundoff errors lead to loss of biorthogonality of the Lanczos vectors.
$\longrightarrow$ near but poor copies

- it is not possible to find a Krylov subspace size m for which an eigenvalue of the matrix $\mathbf{T}_m$ accurately approximates an eigenvalue of the matrix $\mathbf{G}$
- for each nearly converged eigenvalue of the matrix $\mathbf{G}$ there is a cluster of closely spaced eigenvalues of $\mathbf{T}_m$, but the norm of differences between eigenvalues in the cluster may be so large that it is difficult to identify the cluster and impossible to determine an accurate value for the eigenvalue
- the width of the clusters increases with m

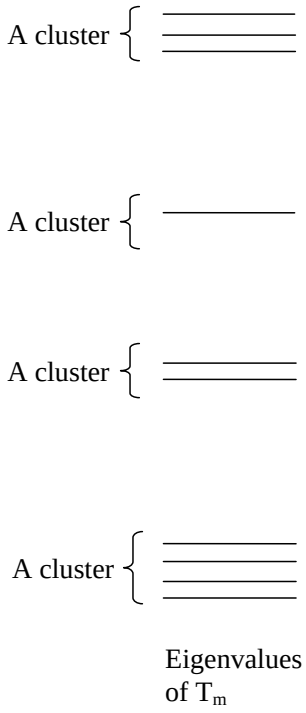

More
iterations
Eigenvalues
of T_m
Eigenvalues
of T_m

What is the RULE ?

- Using the ULA, compute an unsymmetric tridiagonal matrix $\mathbf{T}_m$ for a large value of m.
- Transform $\mathbf{T}_m$ to obtain a complex symmetric tridiagonal matrix with the same eigenvalues
- Compute eigenvalues of $\mathbf{T}_m$
- Form clusters of eigenvalues of $\mathbf{T}_m$. Two eigenvalues are in the same cluster if

$$|\theta_k - \theta_j| \leq \eta \ \max(|\theta_k|, |\theta_j|) \quad ,$$

where η is a user defined tolerance.


A cluster
A cluster
A cluster
A cluster
Eigenvalues
of T_m

- Identify and remove one-eigenvalue clusters that are spurious
- For each non-spurious cluster compute an average eigenvalue
- Use these average eigenvalues as shifts with inverse iteration to determine approximate right and left eigenvectors of the matrix $\mathbf{T}_m$;
  $\mathbf{z}_r^j$ and $\mathbf{z}_l^j$ are the right and left eigenvectors of $\mathbf{T}_m$
- Determine approximate left and right eigenvectors of the matrix $\mathbf{G}$ by reading Lanczos vectors $\mathbf{v}_k$ and $\mathbf{w}_k$ from disk and combining them according to

$$\mathbf{r}_j = \sum_{k=1}^{m} \mathbf{z}_r^j(k)\mathbf{v}_k \quad , \quad \mathbf{l}_j = \sum_{k=1}^{m} \mathbf{z}_l^j(k)\mathbf{w}_k \quad ,$$

where $\mathbf{z}_r^j(k)$ ($\mathbf{z}_l^j(k)$) is the kth component of $\mathbf{z}_r^j$ ($\mathbf{z}_l^j$).

- To solve

$$\mathbf{G}\mathbf{X}_{(R)} = \mathbf{X}_{(R)}\mathbf{\Lambda} \quad ,$$

replace

$$\mathbf{X}_{(R)} \simeq \hat{\mathbf{X}}_{(R)} = \mathbf{R}_k\mathbf{Y}_{(R)} \quad .$$

This yields,

$$\mathbf{G}\mathbf{R}_k\mathbf{Y}_{(R)} = \mathbf{R}_k\mathbf{Y}_{(R)}\tilde{\mathbf{\Lambda}} \quad ,$$

Solve the generalized eigenvalue problem,

$$\mathbf{G}_k\mathbf{Y}_{(R)} = \mathbf{S}_k\mathbf{Y}_{(R)}\tilde{\mathbf{\Lambda}} \quad ,$$

where $\mathbf{G}_k = \mathbf{L}_k^*\mathbf{G}\mathbf{R}_k$ and $\mathbf{S}_k = \mathbf{L}_k^*\mathbf{R}_k$.

- Eigenvalues are computed in groups

How do we choose m?

Increasing m does degrade the approximate eigenvectors so there is no need to carefully search for the best m. We choose a large value of m, generate Lanczos vectors, and compute $\mathbf{G}_k$. We compare the eigenvalues determined with those computed with a larger value of m and increase m if necessary.

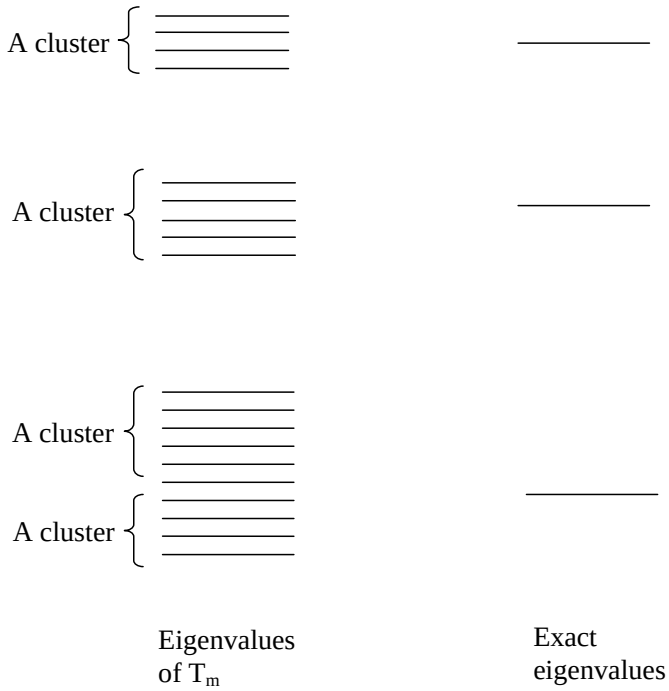
How do we choose η ?

If η is too small :

Several $\mathbf{r}_j$ and $\mathbf{l}_j$ vectors are nearly parallel.

This occurs if several of the clusters contain eigenvalues of the matrix $\mathbf{T}_m$ associated with one exact eigenvalue of the matrix $\mathbf{G}$.

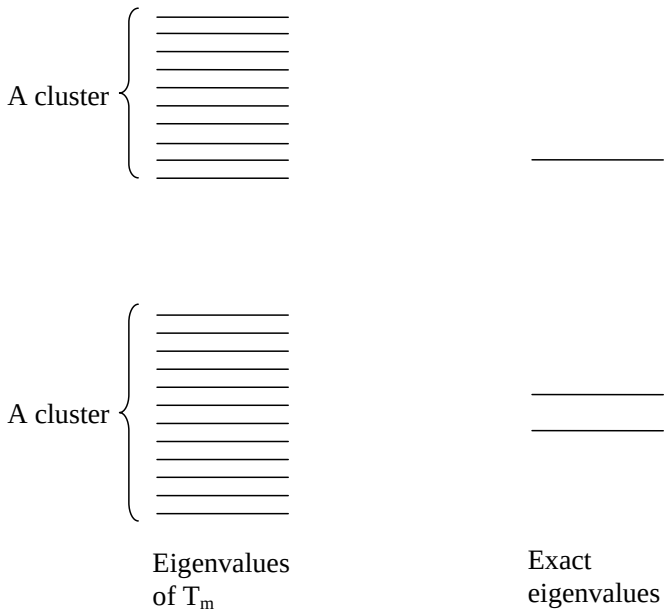
To minimize the near linear dependance of the vectors $\mathbf{r}_j$ and $\mathbf{l}_j$ and to reduce the number of the $\{\mathbf{z}_r^j, \mathbf{z}_l^j\}$ pairs, we retain only pairs for which Ritz values obtained from inverse iteration differ by more than $\eta \max(|\theta_j^{m_j}|, |\theta_k^{m_k}|)$.


A cluster
A cluster
A cluster
A cluster
Eigenvalues
of T_m
Exact
eigenvalues

How do we choose η ?

If η is too large :

Eigenvalues of the matrix $\mathbf{T}_m$ associated with two or more exact eigenvalues will be lumped into a single cluster and one will miss eigenvalues.


A cluster
A cluster
Eigenvalues
of T_m
Exact
eigenvalues

One could actually use the RULE with $\eta = 0$ (i.e. cluster width of zero).

This would increase the number of $\{\boldsymbol{z}_r^j, \boldsymbol{z}_l^j\}$ pairs to compute and make the calculation more costly.

We put η equal to be the square root of the machine precision.

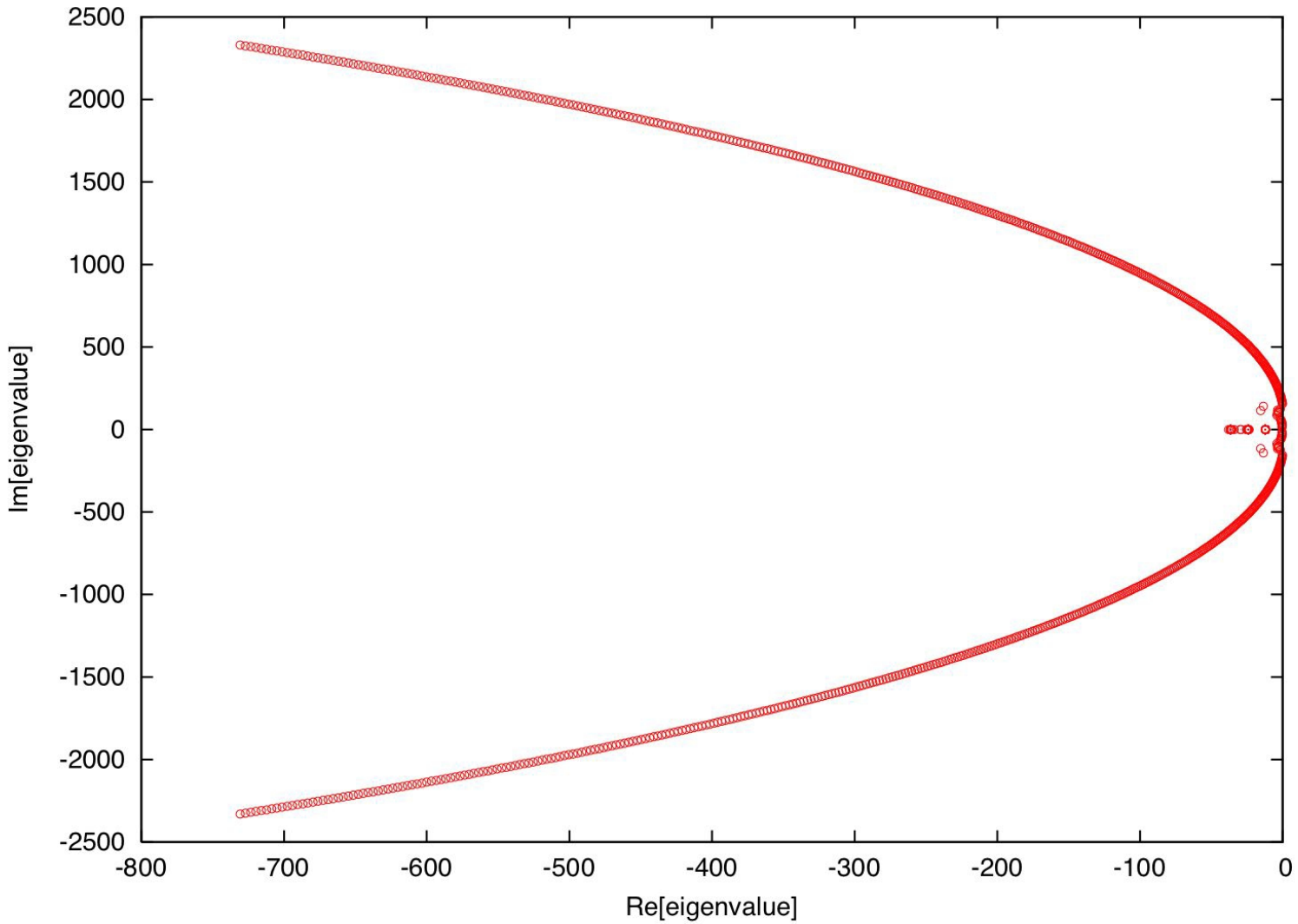
Numerical experiments

TOLOSA matrices from the matrix market.

The largest matrix is 2000×2000.

The eigenvalues of interest are the three with the largest imaginary parts.

Distribution of eigenvalues of the TOLOSA2000 matrix



Error in extremal eigenvalues of the TOLOSA2000 matrix

Eigenvalue	ULA*	RULE**	ARPACK†	$\lvert \mathbf{x}^*_{(L)} \mathbf{x}_{(R)} \rvert$
$-723.2940 - 2319.859i$	5.4E-05	6.3E-07	2.5E-09	7.8574E-04
$-723.2940 + 2319.859i$	5.4E-05	6.3E-07	2.5E-09	7.8574E-04
$-726.9866 - 2324.992i$	4.8E-06	8.8E-11	1.4E-07	7.8360E-04
$-726.9866 + 2324.992i$	4.8E-06	8.5E-11	1.4E-07	7.8360E-04
$-730.6886 - 2330.120i$	6.7E-07	1.2E-11	4.6E-08	7.8148E-04
$-730.6886 + 2330.120i$	6.7E-07	1.2E-11	4.6E-08	7.8148E-04

* 170 Lanczos iterations ; 340 matrix-vector products

** 170 Lanczos iterations ; 346 matrix-vector products

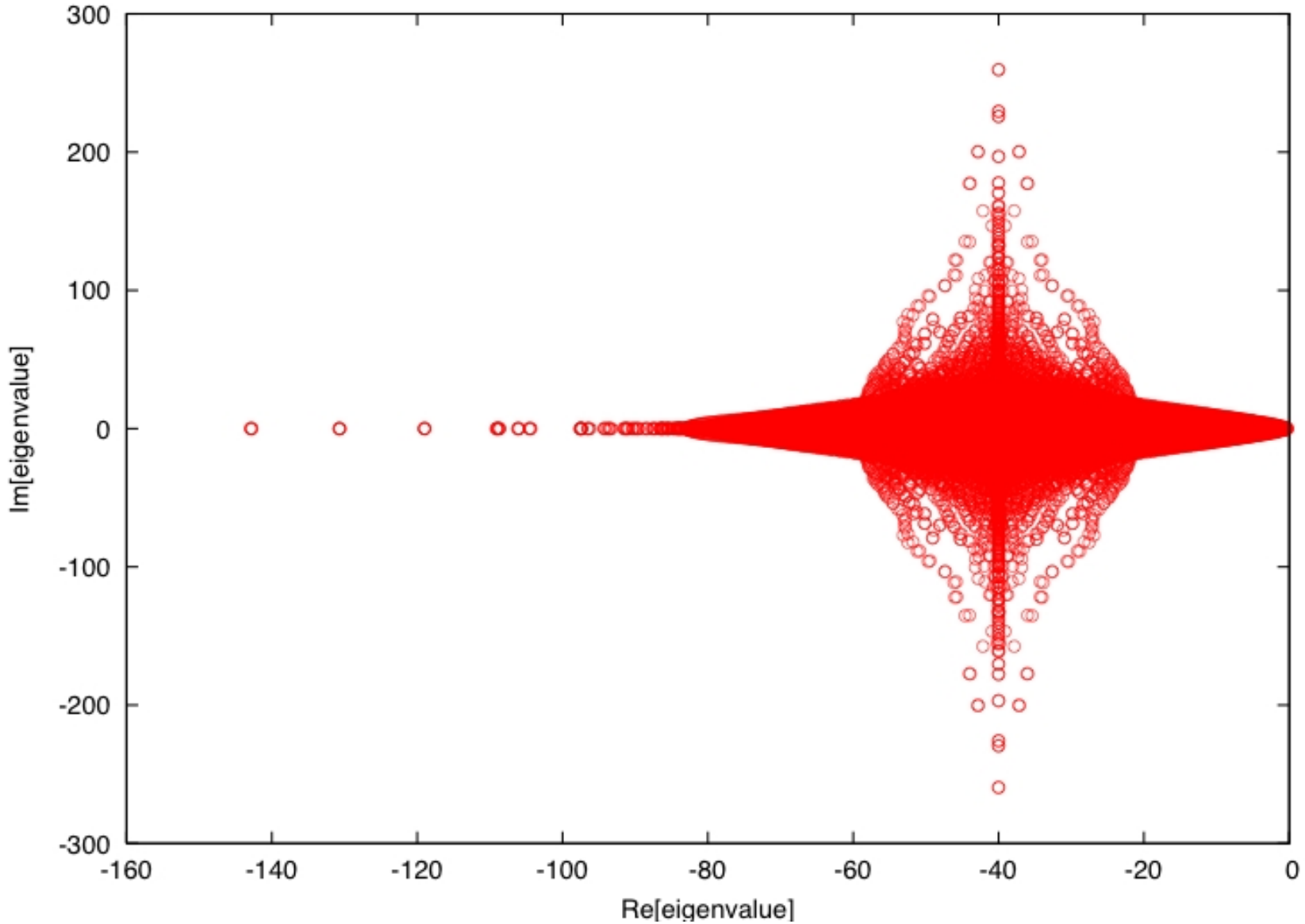
† ARPACK parameters : k = 6, p=300, 2 restarts, 888 matrix-vector products

A larger matrix - The AIRFOIL matrix

It is 23'560 $\times$ 23'560.

We focus on the eigenvalues of largest magnitude.

Distribution of eigenvalues of the Airfoil-23'560 matrix



Error in extremal eigenvalues of the Airfoil-23'560 matrix

Eigenvalue	ULA*	RULE**	ARPACK†	RULE Residual
$-37.18288 + 200.1360i$	1.1E-08	7.1E-13	2.3E-07	3.3E-08
$-37.18288 - 200.1360i$	1.1E-08	4.3E-13	2.3E-07	3.5E-08
$-37.14444 + 200.2190i$	3.8E-09	7.4E-13	4.8E-07	1.1E-07
$-37.14444 - 200.2190i$	3.8E-09	5.8E-13	4.8E-07	9.1E-08
$-42.81924 + 200.1356i$	7.5E-08	1.1E-12	1.9E-07	1.2E-07
$-42.81924 - 200.1356i$	7.5E-08	7.0E-13	1.9E-07	7.0E-08
$-42.85767 + 200.2186i$	9.0E-08	9.5E-13	3.5E-07	8.3E-08
$-42.85767 - 200.2186i$	9.0E-08	5.5E-13	3.5E-07	8.9E-08
$-40.00075 + 225.7328i$	1.2E-08	6.5E-13	8.6E-07	4.7E-08
$-40.00075 - 225.7328i$	1.2E-08	3.7E-13	8.6E-07	2.2E-08
$-40.00074 + 225.7714i$	3.4E-09	7.4E-13	8.4E-07	2.9E-08
$-40.00074 - 225.7714i$	3.3E-09	1.1E-12	8.4E-07	2.8E-08
$-40.00010 + 229.5187i$	1.1E-08	3.1E-13	6.2E-07	3.9E-08
$-40.00010 - 229.5187i$	1.1E-08	1.1E-13	6.2E-07	3.3E-08
$-40.00095 + 229.5291i$	1.2E-08	5.7E-13	8.0E-07	7.7E-08
$-40.00095 - 229.5291i$	1.2E-08	1.5E-13	8.0E-07	2.9E-08
$-40.00045 + 259.5435i$	1.2E-10	1.0E-12	4.8E-07	2.2E-07

* 125 Lanczos iterations ; 250 matrix-vector products

** 125 Lanczos iterations ; 270 matrix-vector products

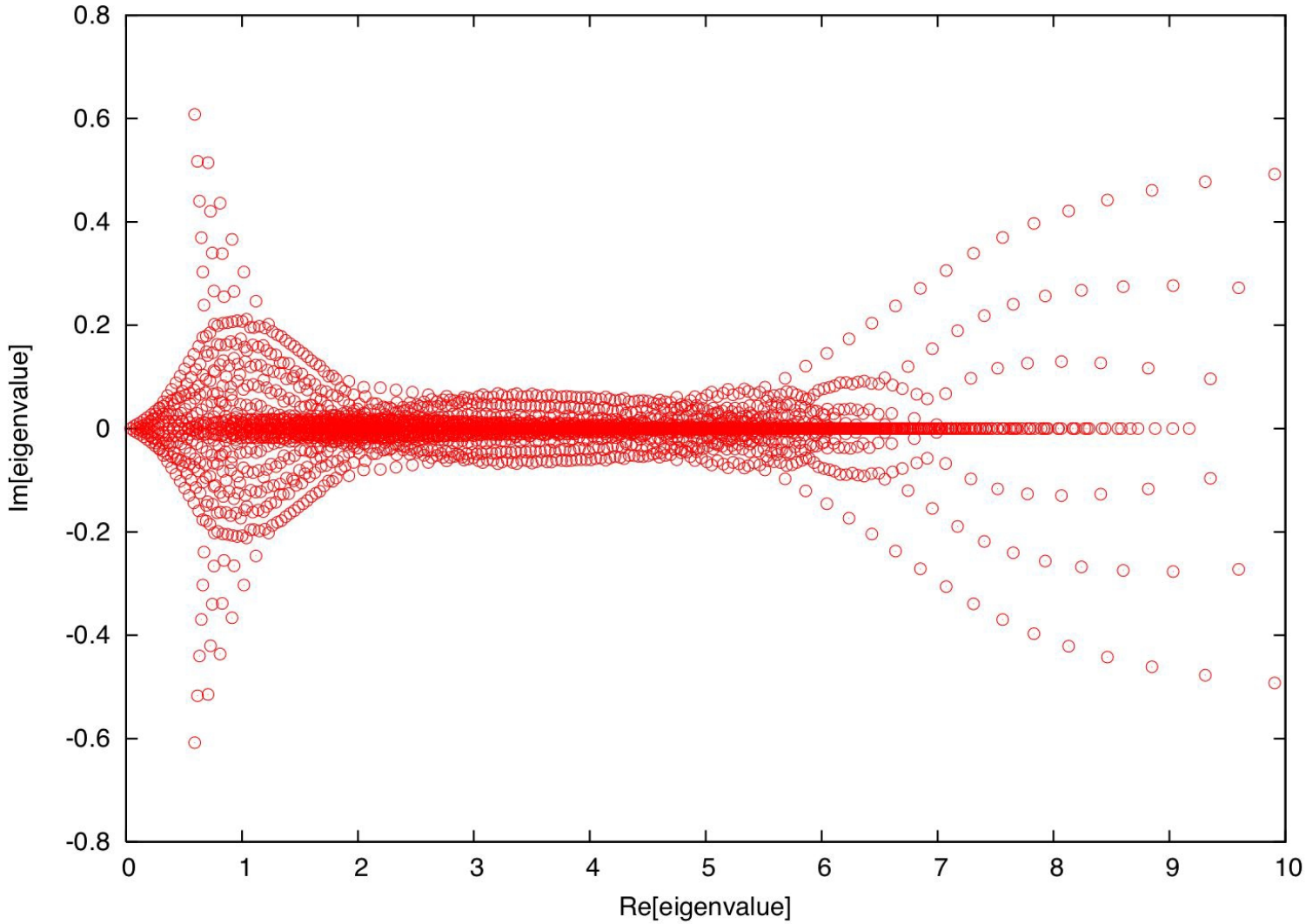
† ARPACK parameters : k = 20, p=50, 6 restarts, 172 matrix-vector products

Interior eigenalues - The PDE matrix

Obtained from matrix market.

It is 2961×2961.

Distribution of eigenvalues of the PDE-2961 matrix



Interior eigenvalues of the PDE-2961 matrix

The chosen target is $8.3 + 0.35i$.

Refined eigenvalue	ULA*	RULE**	$\lvert \mathbf{x}^*_{(L)} \mathbf{x}_{(R)} \rvert$
$7.831661 + 0.3970848i$	7.4E-09	2.4E-12	3.0064E-02
$7.928410 + 0.2564949i$	3.3E-07	3.0E-11	2.0611E-03
$8.130354 + 0.4211668i$	1.0E-08	6.3E-14	5.3381E-02
$8.240396 + 0.2678090i$	3.8E-08	3.0E-13	3.8846E-03
$8.465128 + 0.4423992i$	2.7E-09	1.4E-14	8.1620E-02
$8.600357 + 0.2746980i$	9.4E-08	1.1E-13	6.3615E-03

* 450 Lanczos iterations ; 900 matrix-vector products

** 450 Lanczos iterations ; 906 matrix-vector products

Interior eigenvalues of a 184'000 × 184'000 matrix used to compute lifetimes of metastable states of HCO

Refined eigenvalue	RULE Residual	$\lvert\mathbf{x}^*_{(L)}\mathbf{x}_{(R)}\rvert$
$0.000000000001608 + 99.63317i$	4.3E-09	2.007E-02
$0.000000000423172 + 94.06122i$	4.3E-09	2.126E-02
$-0.000000000030880 + 89.78747i$	2.0E-07	2.227E-02
$0.000000000017839 + 88.24727i$	7.6E-08	2.266E-02
$0.000000000000180 + 86.54286i$	1.3E-08	2.311E-02
$0.000000000000180 + 86.54286i$	1.9E-09	2.392E-02
$0.000000000013164 + 82.15547i$	1.9E-07	2.434E-02
$-0.000000000000079 + 80.30376i$	1.6E-08	2.490E-02
$-0.000000000003208 + 78.87198i$	2.0E-08	2.535E-02
$0.000000000002248 + 77.02845i$	8.2E-09	2.596E-02
$-0.000000000000226 + 75.73480i$	2.5E-09	2.640E-02
$-0.000000000000173 + 75.02610i$	9.3E-09	2.665E-02
$-0.000000000046711 + 73.83727i$	3.4E-08	2.708E-02
$-0.000000000044467 + 73.27523i$	1.8E-08	2.729E-02
$0.000000000000506 + 71.77733i$	3.2E-09	2.786E-02
$-0.000000000001369 + 70.00861i$	2.3E-09	2.856E-02
$-0.001503206670272 + 69.62269i$	1.2E-08	2.872E-02

Conclusion

- A simple refinement (the RULE) makes it possible to extract accurate eigenvalues from the Krylov subspaces obtained from the ULA.
- The RULE makes it possible to use large Krylov subspaces without storing a large number of vectors in memory and re-biorthogonalizing Lanczos vectors.
- It can therefore be used to compute many eigenvalues of very large matrices.
- The refinement is inexpensive. Eigenvalues are computed in groups of about 20. For each group of 20 the cost of the refinement is only 20 additional matrix-vector products.

This work has been supported by the
Natural Sciences and Engineering
Research Council of Canada