Conjugate-gradient eigenvalue solvers in computing electronic properties of nanostructure architectures

Stanimire Tomov*

Innovative Computing Laboratory,
The University of Tennessee,
Knoxville, TN 37996-3450, USA
E-mail: tomov@cs.utk.edu
*Corresponding author

Julien Langou

Department of Mathematical Sciences,
University of Colorado at Denver and Health Sciences Center,
Denver, CO 80217-3364, USA
E-mail: julien.langou@cudenvar.edu

Jack Dongarra

Innovative Computing Laboratory,
The University of Tennessee,
Knoxville, TN 37996-3450, USA
E-mail: dongarra@cs.utk.edu

Andrew Canning

Lawrence Berkeley National Laboratory,
Computational Research Division,
Berkeley, CA 94720, USA
E-mail: acanning@lbl.gov

Lin-Wang Wang

Lawrence Berkeley National Laboratory,
Computational Research Division,
Berkeley, CA 94720, USA
E-mail: iwwang@lbl.gov

Abstract: In this paper we report on our efforts to test and expand the current state-of-the-art in eigenvalue solvers applied to the field of nanotechnology. We singled out the non-linear Conjugate Gradients (CG) methods as the backbone of our efforts for their previous success in predicting the electronic properties of large nanostructures and made a library of three different solvers (two recent and one new) that we integrated into the Parallel Energy SCAN (PESCAN) code to perform a comparison. The methods and their implementation are tuned to the specifics of the physics problem. The main requirements are to be able to find (1) a few, approximately 4–10, of the (2) interior eigenstates, including (3) repeated eigenvalues, for (4) large Hermitian matrices.

Keywords: computational nanotechnology; parallel eigenvalue solvers; quantum dots; conjugate gradient methods; block methods.

Reference to this paper should be made as follows: Tomov, S., Langou, J., Dongarra, J., Canning, A. and Wang, L-W. (2006) 'Conjugate-gradient eigenvalue solvers in computing electronic properties of nanostructure architectures', *Int. J. Computational Science and Engineering*, Vol. 2, Nos. 3/4, pp.205–212.

Biographical notes: Stanimire Tomov is Senior Research Associate in the CS Department at the University of Tennessee (UT). His research interests include numerical linear algebra, numerical approximation of PDEs and parallel computing.

Julien Langou received his PhD in 2003 in Applied Mathematics from the Doctoral School of Toulouse. He is now a Research Scientist in the CS Department at UT. His research interest is in numerical linear algebra in particular iterative methods and dense linear algebra.

Jack Dongarra is University Distinguished Professor of CS in the CS Department at UT and Distinguished Research Staff in the CS and Mathematics Division at Oak Ridge National Laboratory. He specialises in numerical algorithms in linear algebra, parallel computing, use of advanced-computer architectures, programming methodology and tools for parallel computers. His research includes the development, testing and documentation of high quality mathematical software. He has contributed to the design and implementation of the following open source software packages and systems: EISPACK, LINPACK, the BLAS, LAPACK, ScaLAPACK, Netlib, PVM, MPI, NetSolve, Top500, ATLAS and PAPI. He has published approximately 200 papers, reports and technical memoranda and he is co-author of several books.

Andrew Canning is a Staff Scientist in the Computational Research Division at LBNL and an Adjunct Professor in the Department of Applied Sciences at UC Davis. His research is in materials science, scientific computation and parallel computing. Along with collaborators he won the 1998 Gordon Bell Prize for the fastest parallel application for a simulation of complex magnetic properties.

Lin-Wang Wang is a Staff Scientist in the Computational Research Division of LBNL. His research areas include nanostructure simulations, ab initio calculations and algorithm/code developments. He is interested in the electronic structures and optical properties of nano-sized systems. He has developed methods to calculate the nanostructures containing thousands of atoms in ab initio accuracy.

1 Introduction

Eigenvalue problems result from the simulation of many physical phenomena. For example, in computational electromagnetics, this may be the problem of finding electric/magnetic field frequencies that propagate through the medium; in structural dynamics, to find the frequencies of free vibrations of an elastic structure, etc. In the field of nanotechnology and more precisely in predicting the electronic properties of semiconductor nanostructure architectures, this may be the problem of finding 'stable' electronic states and their energies (explained below). Because of the eigen-solvers' wide application area they have been investigated by many branches of mathematical physics, computational mathematics and engineering. Our efforts here are to test and expand the current state-of-the-art in eigenvalue solvers in predicting the electronic properties of quantum nanostructures.

First-principles electronic structure calculations are typically carried out by minimising the quantum-mechanical total energy with respect to its electronic and atomic degrees of freedom. Subject to various assumptions and simplifications (Payne et al., 1992), the electronic part of this minimisation problem is equivalent to solving the single particle Schrödinger-type equations (called Kohn–Sham equations)

$$\hat{H}\psi_i(r) = \epsilon_i \psi_i(r) \tag{1}$$

$$\hat{H} = -\frac{1}{2}\nabla^2 + V$$

where $\psi_i(r)$ are the single particle wave functions (of electronic state i) that minimise the total energy and V is the total potential of the system. The wave functions are most commonly expanded in plane-waves (Fourier components) up to some cut-off energy which discretises Equation (1). In this approach the lowest eigen-pairs are calculated for $\hat{H}$ and the Kohn-Sham equations are solved self-consistently. For a review of this approach see reference (Payne et al., 1992) and the references therein. The computational cost of this approach scales as the cube of the number of atoms and the maximum system size that can be studied is of the order of hundreds of atoms. In the approach used in PESCAN developed by Wang and Zunger (1994) a semi-empirical potential or a charge patching method (Wang and Li, 2004) is used to construct V and only the eigenstates of interest around a given energy are calculated, allowing the study of large nanosystems (up to a million atoms). The problem then becomes: find ψ and E close to a given E_{ref} such that

$$H\psi = E\psi \tag{2}$$

where H represents the Hamiltonian matrix, which is Hermitian with dimension equal to the number of Fourier components used to expand ψ. The dimension of H may be of the order of a million for large nanosystems. The eigenvalues E (energy of state ψ) are real and the eigenvectors ψ are orthonormal.

In many cases, like semiconductor quantum dots, the spectrum of H has energy gaps and of particular interest to physicists is to find a few, approximately 4–10, of the interior eigenvalues on either side of the gap which determines many of the electronic properties of the system. Due to its large size H is never explicitly computed. We calculate the kinetic energy part in Fourier space, where it is diagonal and the potential energy part in real space so that the number of calculations used to construct the matrix-vector product scales as $n \log n$ rather than n^2 where n is the dimension of H. Three dimensional FFTs

are used to move between Fourier and real space. H is therefore available as a procedure for computing Hx for a given vector x. Thus one more requirement is that the solver is matrix free. Finally, repeated eigenvalues (degeneracy) of approximately 3 maximum are possible for the problems discussed and we need to be able to resolve such cases to fully understand the electronic properties of our systems.

Currently, Equation (2) is solved by a CG method as coded in the PESCAN package (Wang and Zunger, 1994). While this programme works well for 1000 atom systems with a sizable band gap (e.g. 1 eV), it becomes increasingly difficult to solve for systems with

1 large number of atoms (e.g. more than 1 million)

2 small band gap and where

3 many eigenstates need to be computed (e.g. more than 100) or to solve eigenstates when there is no band gap (e.g. for Auger or transport calculations).

Thus, new algorithm to solve this problem is greatly needed.

The focus of this paper is on non-linear CG methods with folded spectrum. The goal is to solve the interior eigenstates. Alternatives for the folded spectrum transformation are shift-and-invert or fixed-polynomial (Thornquist, 2003). Our choice of method is based on the highly successful current scheme (Canning et al., 2000) which has been proven to be efficient and practical for the physical problems we are solving. It will be the subject of other studies to further investigate the applicability of other alternatives like the Lanczos and Jacobi-Davidson method (for comparison of these and other methods for finding a large number of eigenvectors applied to elasticity problems see Arbenz et al., 2005).

There are many methods for obtaining eigenstates for symmetric (or Hermitian) matrices and several excellent books have been written on the subject (see Bai et al., 2000; Golub and Van Loan, 1989; Parlett, 1980; Saad, 1991). From them methods like QR work extremely well for relatively small matrices. For our case of extremely large matrices parallel iterative methods have to be employed. Some of the most popular methods in this class are the conjugate gradient, Lanczos and inverse iteration. The convergence of these methods depends on the condition number of the matrix at hand, and to accelerate it various preconditioning methods have been developed (see e.g. the survey (Knyazev, 1998)). Compared to preconditioners for linear systems, preconditioners for eigensolvers are less known and developed. A preconditioning method for quantum chemistry problems that has become popular recently is Davidson's method (see e.g. Davidson, 1975; Morgan, 1990; Sleijpen et al., 1996). We are also interested in the method because of its efficiency on diagonally dominant matrices (the case of some of our applications), where even simple diagonal preconditioner will work well. Another class of preconditioners that are a topic of current research and are of significant importance to parallel eigensolvers, are the domain decomposition type preconditioners (see e.g. Knyazev, 2001). For the work described here we have used diagonal preconditioning (Payne et al., 1992).

In Section 2, we describe the three eigensolvers investigated in the paper and the spectral transformation used. We give our numerical results in Section 3 and finally, in Section 4 we give some concluding remarks.

2 Non-linear CG method for eigenvalue problems

The conventional approach for problems of very large matrix size is to use iterative projection methods where at every step one extracts eigenvalue approximations from a given subspace S of small dimension (see e.g. Bai et al., 2000). Non-linear CG methods belong to this class of methods. Let us assume for now that we are looking for the smallest eigenvalue of the Hermitian operator A.

This eigenvalue problem can be expressed in terms of function minimisation as: find the variational minimum of $F(x) = < x, Ax >$, under the constraint of $x^T x = I$, on which a non-linear CG method is performed. The orthonormal constraint $x^T x = I$ makes the problem non-linear.

In this section, we first give a description of the algorithms that we have implemented in our library, namely: the Preconditioned Conjugate Gradient (PCG) method, the PCG with $S = \mathrm{span}\{X, R\}$ method (PCG-XR) and the locally optimal PCG method (LOBPCG). Finally, we describe the spectral transformation that we use to get the interior eigenvalues of interest.

2.1 PCG method

In Table 1, we give a pseudo-code of the PCG algorithm for eigen-problems. This is the algorithm originally implemented in the PESCAN code (see also Payne et al., 1992; Wang and Zunger, 1996a).

Table 1 PCG algorithm

```
 1  do i = 1, niter
 2    do m = 1, blockSize
 3      orthonormalize X(m) to X(1 : m − 1)
 4      ax = A X(m)
 5      do j = 1, nline
 6        λ(m) = X(m) · ax
 7        if (||ax − λ(m) X(m)||_2 < tol) exit
 8        r_{j+1} = (I − X(m) X(m)^T) ax
 9        β = (r_{j+1}·Pr_{j+1}) / (r_j·Pr_j)
10        d_{j+1} = −P r_{j+1} + β d_j
11        d_{j+1} = (I − X(m) X(m)^T) d_{j+1}
12        γ = ||d_{j+1}||_2^{−1}
13        θ = 0.5 |atan( 2 γ d_{j+1}·ax / (λ(m) − γ^2 d_{j+1}·A d_{j+1}) )|
14        X(m) = cos(θ) X(m) + sin(θ) γ d_{j+1}
15        ax = cos(θ) ax + sin(θ) γ A d_{j+1}
16      enddo
17    enddo
18    [X, λ] = Rayleigh − Ritz on span{X}
19  enddo
```

Here P is a preconditioner for the operator A, X is the block of `blockSize` column eigenvectors sought and λ is the corresponding block of eigenvalues. In the

above procedure, $X^T X = I$ is satisfied throughout the process. $(I - XX^T)$ is a projection operator, which when applied to y deflates $\mathtt{span}\{X\}$ from y, thus making the resulting vector orthogonal to $\mathtt{span}\{X\}$. The matrix–vector multiplication happens at line 15. Thus there is one matrix–vector multiplication in each j iteration. The above procedure converges each eigen-vector separately in a sequential way. It is also called state-by-state (or band-by-band in the physics community) method, in contrast to the block method to be introduced next.

2.2 LOBPCG method

Briefly, the LOBPCG method can be described with the pseudo-code in Table 2. Note that the difference with the PCG is that the m and j loops are replaced with just the blocked computation of the preconditioned residual, and the Rayleigh-Ritz on $\mathtt{span}\{X_i\}$ with Rayleigh-Ritz on $\mathtt{span}\{X_{i-1}, X_i, R\}$ (in the physics community Rayleigh-Ritz is known as the process of diagonalising A within the spanned subspace and taking the 'blocksize' lowest eigen-vectors). The direct implementation of this algorithm becomes unstable as X_{i-1} and X_i become closer and closer, and therefore special care and modifications have to be taken (see Knyazev, 2001).

Table 2 LOBPCG algorithm

```
1  do i = 1, niter
2     R = P (A X_i − λ X_i)
3     check convergence criteria
4     [X_i, λ] = Rayleigh − Ritz on span{X_i, X_{i−1}, R}
5  enddo
```

2.3 PCG-XR method

PCG-XR is a new algorithm that we derived from the PCG algorithm by replacing line 18 in Table 1 with

$$18 \qquad [X,\ \lambda] = \mathtt{Rayleigh} - \mathtt{Ritz\ on\ span}\{X, R\}$$

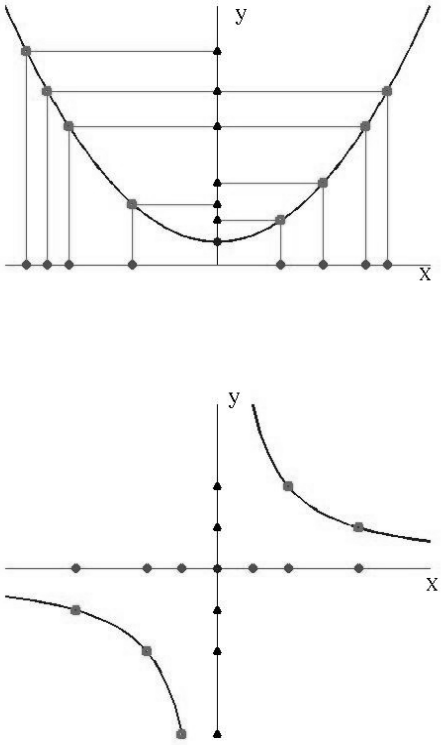
The idea, as in the LOBPCG, is to use the vectors R to perform a more efficient Rayleigh-Ritz step.

2.4 Folded spectrum method

Projection methods are good at finding well separated (non-clustered) extremal eigenvalues. In our case, we are seeking for interior eigenvalues and thus we have to use a spectral transformation, the goal being to map the sought eigenvalue of our operator to extremal eigenvalues of another one.

To do so we use the folded spectrum method. The interior eigenvalue problem $Hx = \lambda x$ is transformed to find the smallest eigenvalues of $(H - E_{\text{ref}} I)^2 x = \mu x$. The eigenvalues of the original problem are given back by $\mu = (\lambda - s)^2$ (see Figure 1, top). A drawback is that $(H - E_{\text{ref}} I)^2$ has significantly higher than H condition number. Alternative is shift and invert where the transformed problem is $(H - E_{\text{ref}} I)^{-1} x = \mu x$ and the original eigenvalues are given by $\lambda = E_{\text{ref}} + 1/\mu$ (see Figure 1, bottom). A drawback is that one has to invert/solve $H - E_{\text{ref}} I$.

Figure 1 Spectral transformations: folded spectrum (top) and shift and invert (bottom). Shown are discrete spectral values of a matrix on the x-axis and the discrete spectrum of the transformed matrix on the y-axis. The y-axis intersects x at the point of interest E_{ref}



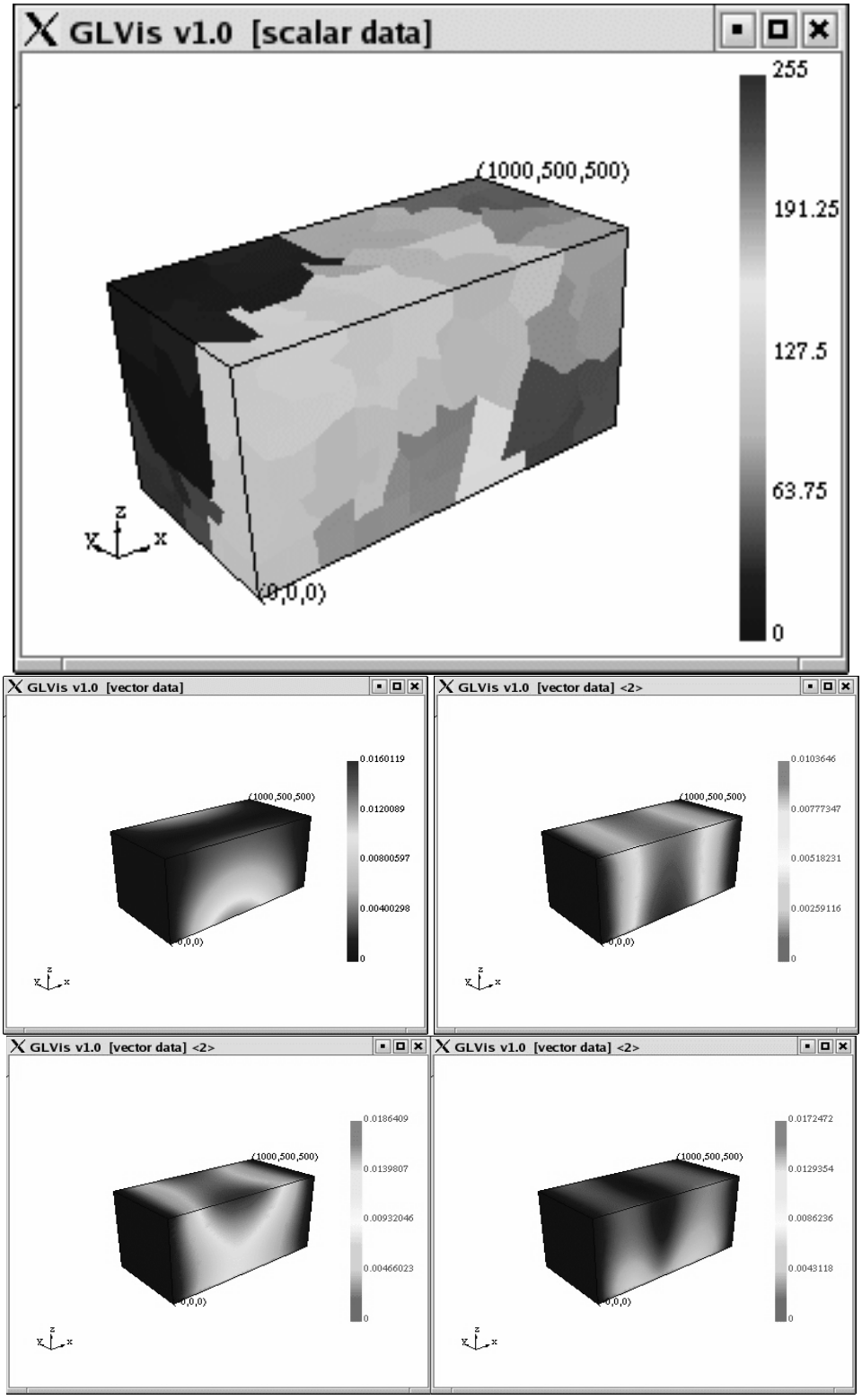
The PCG algorithm in its folded form (FS-PCG) is described in Wang and Zunger (1996a). To adapt the folded spectrum to LOBPCG (FS-LOBPCG), we have added three more block vectors that store the matrix–vector products of the blocks X, R and P with the matrix H. This enables us to control the magnitude of the residuals for an overhead of a few more axpy operations (otherwise we just have access to the magnitude of the residuals of the squared operator). Also the deflation strategy of LOBPCG is adapted in FS-LOBPCG, as the vectors are deflated when the residual relative to H has converged (not H^2).

3 Numerical results

3.1 Software package

We implemented the LOBPCG,[1] PCG and PCG-XR methods in a software library. Currently, it has single/double precision for real/complex arithmetic and both parallel (MPI) and sequential versions. The library is written in Fortran 90. The folded spectrum spectral transformation is optional. The implementation is stand alone and meant to be easily integrated in various physics codes. For example, we integrated our solvers with `ParaGrid`, a finite element/volume method based testing software for massively parallel computations. Figure 2 illustrates a numerical result on applying our solvers on an elasticity problem generated with `ParaGrid` on 256 processors.

Figure 2 Our eigen-solver's library applied within `ParaGrid`, top: partitioning of the computational domain into 256; bottom: the 4 lowest eigenmodes for an elasticity problem (solved in parallel on 256 processors)



Tables 3 and 4 represent a test on the scalability of the CG eigensolver. Table 3 is for a problem of size 230,000 and Table 4 is for size $\approx 1{,}900{,}000$.

A test case is provided with the software. It represents a 5-point operator where the coefficients a (diagonal) and b (for the connections with the 4 closest neighbours on a regular 2D mesh) can be changed. In Table 5 the output of the test is presented. It is performed on a Linux Intel Pentium IV with Intel Fortran compiler and parameters ($a = 8$, $b = -1 - i$). We are looking for the 10 smallest eigenstates, the matrix size is 20,000, and the iterations are stopped when all the eigencouples (x, λ) satisfy $\|Hx - x\lambda\| \leq \texttt{tol}\|x\|$, with $\texttt{tol} = 10^{-8}$. In general LOBPCG always performs fewer iterations (i.e. fewer matrix–vector products) than PCG. This advantage comes to the cost of more vector operations (axpys and dot products) and more memory requirements. In

this case, LOBPCG performs approximately 2 times more dot products for 2 times fewer matrix vector products than the PCG method, since the 5-point matrix–vector product takes approximately the time of 7 dot products, PCG gives a better timing.

Table 3 Scalability of the CG eigensolver for 100 iterations on a problem of size 230,000

	Number of processors					
	1	*2*	*4*	*8*	*16*	*32*
Time (s)	29.26	11.67	5.58	3.17	1.85	0.95
Speedup		2.5	5.2	9.23	15.8	30.8

Table 4 Scalability of the CG eigensolver for 100 iterations on a problem of size $\approx$1,900,000

	Number of processors				
	16	*32*	*64*	*128*	*256*
Time (s)	20.65	10.5	5.93	2.75	1.95
Speedup		31.52	55.68	120.16	169.44

Table 5 Comparison of the PCG, PCG-XR and LOBPCG methods in finding 10 eigenstates on a problem of size 20,000 $\times$ 20,000

	PCG	*LOBPCG*	*PCG-XR*
Time (s)	37.1	61.7	20.2
Matvecs	3, 555	1, 679	1, 760
Dotprds	68, 245	137, 400	37, 248
Axpys	66, 340	158, 261	36, 608
Copys	6, 190	9, 976	3, 560

The CG-XR method represents for this test case an interesting alternative for those two methods: it inherits the low number of matrix vector products from the LOBPCG and the low number of dot products from the PCG method.

3.2 Numerical results on some quantum dots

In this section, we present numerical results on quantum dots up to thousand of atoms. The experiments are performed on the IBM-SP `seaborg` at NERSC.

For all the experiments we are looking for `mx` $= 10$ interior eigenvalues around $E_{\text{ref}} = -4.8$ eV, where the band gap is about 1.5–3 eV. We have calculated 4 typical quantum dots: (20Cd, 19Se), (83Cd, 81Se), (232Cd, 235Se) and (534Cd, 527Se). These are real physical systems which can be experimentally synthesised and have been studied previously using the PCG method (Wang and Zunger, 1996b). Non-local pseudopotential is used for the potential term in Equation (1) and spin-orbit interaction is also included. The cutoff energy for the plane-wave basis set is 6.8 Ryd. The stopping criterion for the eigenvector is

$$\|Hx - x\lambda\| \leq 10^{-6}\|x\|$$

All the runs are performed on one node with 16 processors except for the smallest case (20Cd, 19Se) which is run on 8 processors. All the solvers are started with the same initial guess.

The spherical quantum dots Cd20Se19, Cd83Se81, Cd232Se235 and Cd534Se527 that we use in the calculations below have corresponding diameters of: 12.79, 20.64, 29.25 and 38.46 Angstrom. Their Hamiltonians are modelled using pseudopotentials. More details for these module Hamiltonians can be found at Wang and Zunger (1996b).

The preconditioner is the one given in Wang and Zunger (1996a): diagonal with diagonal elements

$$p_i = \frac{E_k^2}{(\frac{1}{2}q_i^2 + V_0 - E_{\text{ref}})^2 + E_k^2)}$$

where q_i is the diagonal term of the Laplacian, V_0 is the average potential and E_k is the average kinetic energy of the wave function ψ. It is meant to be an approximation of the inverse of $(H - E_{\text{ref}})^2$.

A notable fact is that all the solvers find the same 10 eigenvalues with the correct accuracy for all the runs. Therefore they are all robust.

The timing results are given in Table 6. For each test case the number of atoms of the quantum dot and the order n of the corresponding matrix is given. The parameter for the number of iterations in the inner loop ($nline$) for FS-PCG and FS-PCG-XR is chosen to be the optimal one among the values 20, 50, 100, 200 and 500 and is given in brackets after the solver.

Table 6 Comparison of FS-PCG, FS-PCG-XR and FS-LOBPCG methods in finding 10 eigenstates around the gap of quantum dots of increasing size

	# matvec	*# outer it*	*Time*
(20Cd, 19Se) n = 11,331			
FS-PCG(50)	4898	(8)	50.4 sec
FS-PCG-XR(50)	4740	(6)	49.1 sec
FS-LOBPCG	4576		52.0 sec
(83Cd, 81Se) n = 34,143			
FS-PCG(200)	15,096	(11)	264 sec
FS-PCG-XR(200)	12,174	(5)	209 sec
FS-LOBPCG	10,688		210 sec
(232Cd, 235Se) n = 75,645			
FS-PCG(200)	15,754	(8)	513 sec
FS-PCG-XR(200)	15,716	(6)	508 sec
FS-LOBPCG	11,864		458 sec
(534Cd, 527Se) n = 141,625			
FS-PCG(500)	22,400	(6)	1406 sec
FS-PCG-RX(500)	21,928	(4)	1374 sec
FS-LOBPCG	17,554		1399 sec

From Table 6, we observe that the three methods behave almost the same. The best method (in term of time) being either FS-PCG-XR or FS-LOBPCG.

FS-LOBPCG should also benefit in speed over FS-PCG and FS-PCG-XR from the fact that the matrix–vector products are performed by block. This is not the case in the version of the

code used for this paper where the experiments are performed on a single node. The blocked implementation of FS-LOBPCG in PESCAN should run faster and also scale to larger processor counts as latency is less of an issue in the communications part of the code.

Another feature of FS-LOBPCG that is not stressed in Table 6 is its overwhelming superiority over FS-PCG when no preconditioner is available. In Table 7, we illustrate this latter feature. For the quantum dot (83Cd, 81Se), FS-LOBPCG runs 4 times faster than FS-PCG without preconditioner whereas it runs only 1.4 times faster with.

Table 7 Comparison of FS-PCG and FS-LOBPCG with and without preconditioner to find $\mathtt{mx} = 10$ eigenvalues of the quantum dots (83Cd, 81Se)

		# matvec	*Time*
(83Cd, 81Se) n = 34,143			
FS-PCG(200)	precond	15,096	264 sec
FS-LOBPCG	precond	10,688	210 sec
FS-PCG(200)	no precond	71,768	1274 sec
FS-LOBPCG	no precond	17,810	341 sec

For the four experiments presented in Table 6, the number of inner iteration that gives the minimum total time is always attained for a small number of outer iteration, this is illustrated in Table 8 for (232Cd, 235Se) where the minimum time is obtained for 6 outer iterations. Another and more practical way of stopping the inner iteration is in fixing the requested tolerance reached at the end of the inner loop. We call FS-PCG(k) FS-PCG where the inner loop is stopped when the accuracy is less than $k^{n_{\text{outer}}}$, where n_{outer} is number of the corresponding outer iteration.

Table 8 The problem of finding the best inner length for FS-PCG can be avoided by fixing a tolerance as stopping criterion in the inner loop

	# matvec	*# outer it*	*Time*
(232Cd, 235Se) n = 75,645			
FS-PCG(100)	17062	(15)	577 sec
FS-PCG(200)	15716	(6)	508 sec
FS-PCG(300)	15990	(4)	517 sec
FS-PCG(10^{-1})	15076	(6)	497 sec

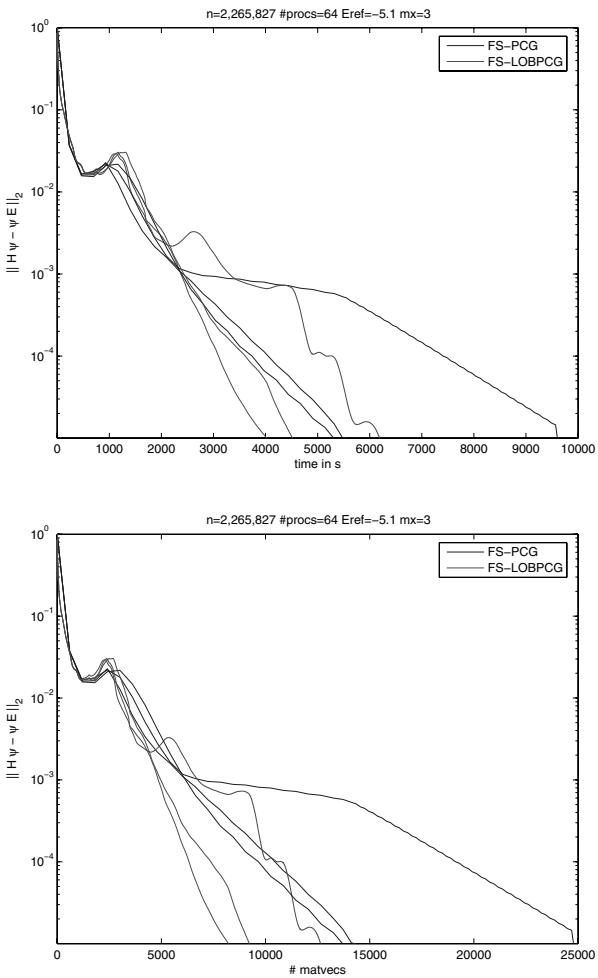
In Table 8, we give the results for FS-PCG(10^{-1}) and (223Cd, 235Se). It comes without a surprise that this solver converges in 6 outer steps as the first inner loop guarantees an accuracy of 10^{-1}, the second inner loop guarantees an accuracy of 10^{-2}, and so on until 10^{-6}. This scheme looks promising. It also allows a synchronized convergence of the block vectors.

3.3 Numerical results on a nanowire system

Here we present numerical results on a larger nanosystem. Namely, we present the results for an InAs nanowire system of about 70,000 atoms. The resulting from the discretisation eigensystem is of size 2,265,827. The run is for 3 eigenstates around $E_{\text{ref}} = -5.1$ on 64 processors on the IBM-SP `seaborg` at NERSC. The stopping criterion, as in the quantum dots cases above, is $\|Hx - x\lambda\| \leq 10^{-6}\|x\|$. We have used a diagonal preconditioner as described for the quantum dots computations.

The numerical results are summarised on Figure 3 where we give a time and number of matrix–vector products comparison for the FS-PCG and FS-LOBPCG eigensolvers. For this problem FS-LOBPCG shows 36% improvement in speed and 49% in matrix–vector products compared to FS-PCG. We used blocked matrix–vector product for the time comparison results. Currently, our block version is slower: a block matrix–vector product is 0.42 sec compared to non-block for 0.36 sec. This also means that FS-LOBPCG has 43% improvement in speed for the non-block matrix vector product.

Figure 3 Time (top) and number of matrix-vector products (bottom) comparison for an InAs nanowire system of about 70,000 atoms using FS-PCG and FS-LOBPCG



4 Conclusions

In this paper, we described and compared three non-linear CG methods with folded spectrum to find a small amount of interior eigenvalues around a given point.

The application is to make a computational prediction of the electronic properties of quantum nanostructures.

The methods were specifically selected and tuned for computing the electronic properties of large nanostructures. There is a need for such methods in the community and the success of doing large numerical simulations in the field depends on them.

All three methods are similar and thus often the results are close; a general ranking being: FS-LOBPCG is the fastest, next FS-PCG-XR and finally FS-PCG. Our numerical experiments demonstrated that the improvement of FS-LOBPCG versus FS-PCG in terms of speed ranges from 10–43% for the nano-problems discussed. In terms of memory requirement the three methods are ranked in the same way: FS-LOBPCG /FS-PCG-XR requires four/two times as much memory as FS-PCG. As our problem scales up the memory has not shown up as a bottleneck yet, that is, using FS-LOBPCG is affordable.

The main drawback of FS-PCG and FS-PCG-XR is their sensitivity to the parameter `nline` (the number of iterations in the inner loop). In order to get rid of this parameter one can instead have a fixed residual reduction to be achieved on each step of the outer loop.

On other applications, the performance of FS-LOBPCG would be still better than FS-PCG if a fast block matrix–vector product and an accommodating preconditioner are available.

Finally, based on our results, if memory is not a problem and block version of the matrix–vector multiplication can be efficiently implemented, the FS-LOBPCG will be the method of choice for the type of problems discussed.

Acknowledgements

This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the US Department of Energy. We thank Gabriel Bester from the National Renewable Energy Laboratory, Golden, CO, for providing the nanowire computations data in Section 3.3. This work was supported by the US Department of Energy, Office of Science, Office of Advanced Scientific Computing (MICS) and Basic Energy Science under LAB03-17 initiative, contract Nos. DE-FG02-03ER25584 and DE-AC03-76SF00098.

References

Arbenz, P., Hetmaniuk, U.L., Lehoucq, R.B. and Tuminaro, R.S. (2005) 'A comparison of eigensolvers for large-scale 3D modal analysis using AMG-preconditioned iterative methods', *International Journal for Numerical and Methods in Engineering*, Vol. 64, pp.204–236.

Bai, Z., Demmel, J., Dongarra, J., Ruhe, A. and van der Vorst, H. (Eds) (2000) *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, SIAM, Philadelphia.

Canning, A., Wang, L.W., Williamson, A. and Zunger, A. (2000) 'Parallel empirical pseudopotential electronic structure calculations for million atom systems', *Journal of Computational Physics*, Vol. 160, pp.29–41.

Davidson, E.R. (1975) 'The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices', *Journal of Computational Physics*, Vol. 17, pp.817–953.

Golub, G.H. and Van Loan, C. (1989) *Matrix Computations*, Baltimore, The John Hopkins University Press.

Knyazev, A. (2001) 'Toward the optimal preconditioned eigensolver: locally optimal block preconditioned conjugate gradient method', *SIAM Journal on Scientific Computing*, Vol. 23, No. 2, pp.517–541.

Knyazev, A. (1998) 'Preconditioned eigensolvers - an oxymoron?' *Electronic Transactions on NA*, Vol. 7, pp.104–123.

Kuznetsov, Y.A. (1986) 'Iterative methods in subspaces for eigenvalue problems', in A.V. Balakrishinan, A.A. Dorodnitsyn and J.L. Lions, (Eds). *Vistas in Applied Mathematics in Numerical Analysis, Atmospheric Sciences, Immunology*, New York: Optimization Software, Inc., pp.96–113.

Morgan, R.B. (1990) 'Davidson's method and preconditioning for generalized eigenvalue problems', *Journal of Computational Physics*, Vol. 89, pp.241–245.

Parlett, B.N. (1980) *The Symmetric Eigenvalue Problem*, Englewood Cliffs, Prentice Hall.

Payne, M.C., Teter, M.P., Allan, D.C., Arias, T.A. and Joannopoulos, J.D. (1992) 'Iterative minimization techniques for ab initio total-energy calculations: molecular dynamics and conjugate gradients', *Reviews of Modern Physics*, Vol. 64, pp.1045–1097.

Saad, Y. (1991) *Numerical Methods for Large Eigenvalue Problems*, Manchester University Press Series in Algorithms and Architectures for Advanced Scientific Computing.

Sleijpen, G.L.G. and Van der Vorst, H. (1996) 'A jacobi-davidson iteration method for linear eigenvalue problems', *SIAM Journal on Matrix Analysis and Applications*, Vol. 17, No. 2, pp.401–425.

Thornquist, H. (2003) 'Fixed-polynomial approximate spectral transformations for preconditioning the eigenvalue problem', Masters Thesis, Department of Computational and Applied Mathematics, Rice University.

Wang, L.W. and Li, J. (2004) 'First principle thousand atom quantum dot calculations', *Physics Review B*, Vol. 69, pp.153–302.

Wang, L.W. and Zunger, A. (1996a) 'Pseudopotential theory of nanometer silicon quantum dots application to silicon quantum dots', In P.V., Kamat and D. Meisel (Eds). *Semiconductor Nanoclusters*, pp.161–207.

Wang, L.W. and Zunger, A. (1996b) 'Pseudopotential calculations of nanoscale CdSe quantum dots', *Physics Review B*, Vol. 53, p.9579.

Wang, L.W. and Zunger, A. (1994) 'Solving schrodinger's equation around a desired energy: application to silicon quantum dots', *Journal of Chemical Physics*, Vol. 100, No. 3, pp.2394–2397.

Note

[1]Available at: http:// www-math.cudenver.edu/˜aknyazev/software /CG/la- test/lobpcg.m (revision 4.10 written in Matlab, with some slight modifications).