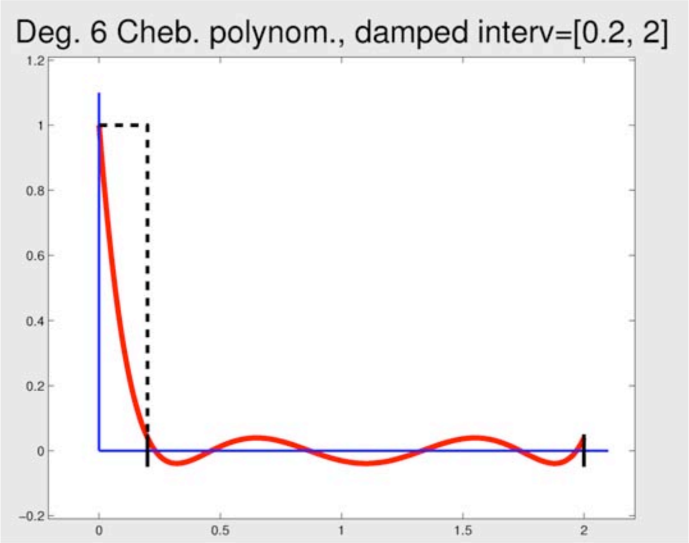
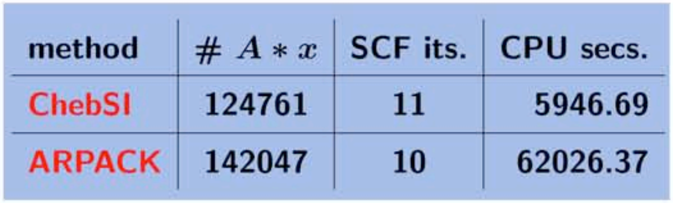
Institute for the Theory of Advanced Materials in Information Technology: James R. Chelikowsky (Texas), Yousef Saad and Renata Wentzcovitch (Minnesota), Steven Louie (UC Berkeley) and Efthimios Kaxiras (Harvard) (DMR- 0551195): Efficient diagonalization methods for the Kohn-Sham equation

Background. The Kohn-Sham equation is typically solved by iterative diagonalization of the associated Hamiltonian matrix. This is in effect a functional iteration for solving a nonlinear eigenvalue problem, combined with a ‘mixing’ technique for improving convergence of the nonlinear self-consistent field (SCF) loop. Many elaborate methods have been used for solving the Kohn-Sham eigenvalue problem. These methods often focus on the diagonalization of matrices. This perspective de-emphasizes the nonlinear nature of the problem and tends to narrow the scope of methods that can be utilized.

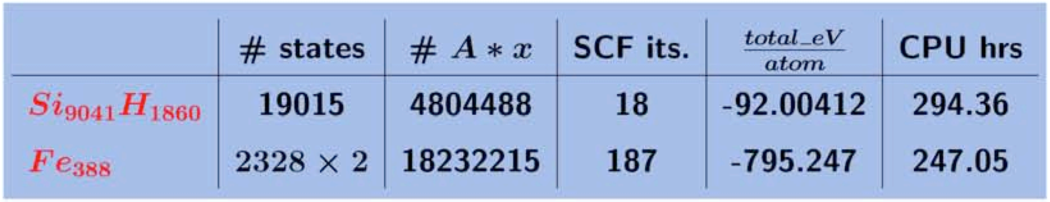
New Approach. We have recently implemented a technique which exploits the nonlinear nature of the self-consistent field iteration. The method solves the original nonlinear Kohn-Sham equation directly by a form of nonlinear subspace iteration. It distinguishes itself from previous methods by not focusing on the intermediate linearized Kohn-Sham eigenvalue problems. Instead, it replaces diagonalization by a polynomial filtering step applied to each basis vector of the subspace being computed. Low degree Chebyshev polynomials are used for filtering as illustrated in the figure. The method reaches convergence within a similar number of self consistent field (SCF) iterations as eigensolver-based approaches but each SCF iteration is much less expensive. The numerical tests show a typical speed-up by an order of magnitude over standard diagonalization-based approaches. This new technique enabled calculations for a number of challenging problems, which heretofore could not be attempted.



Results. The left table below compares the Chebyshev filtering approach with an approach based on ARPACK using one processor for an $Si_{525}H_{276}$ cluster. The right table shows two recent calculations for larger and more challenging systems performed with moderate resources, namely, 48 and 24 SGI Altix (1.6Ghz Itanium) processors respectively.



method	# $A * x$	SCF its.	CPU secs.
ChebSI	124761	11	5946.69
ARPACK	142047	10	62026.37



	# states	# $A * x$	SCF its.	$\frac{total\_eV}{atom}$	CPU hrs
$Si_{9041}H_{1860}$	19015	4804488	18	-92.00412	294.36
Fe_{388}	2328×2	18232215	187	-795.247	247.05