On the need for efficient and scalable solvers for ill-conditioned sparse matrix equations

J. Chen (jchen@pppl.gov), N. Ferraro (nferraro@pppl.gov), S. Jardin (jardin@pppl.gov)
Princeton Plasma Physics Laboratory

Challenge: Many problems of interest to research groups in the DOE Office of Science, and to a wider group of scientists and engineers as well, involve large sparse matrix equations, $\mathbf{A}\bullet\mathbf{X}=\mathbf{B}$, where the matrices $\mathbf{A}$ are poorly conditioned. Our experience is in the implicit MHD equations for fusion applications, but these types of equations also occur in other disciplines such as power grid simulations, subsurface flow, circuit simulation, and computational cardiology. In each instance, the condition number is very high, but for one or more different underlying reasons. This leads to difficulties in iterative solutions as the number of iterations required for solution to a given tolorance normally increases with the condition number.

Opportunity: Domain scientists or engineers, applied mathematicians, and computer scientists all look at these problems differently. The domain scientist might understand the reason for the large condition number, for example in terms of the wide range of time and spatial scales present in the underlying physics equations. The domain scientist and applied mathematician may discuss different preconditioning techniques based on addressing these underlying reasons. The computer scientist may be the best one to implement the preconditioner and iterative solver on modern CPU/GPU hardware in a scalable way. Perhaps the best way to approach this problem is to form an inter-disciplinary team around each domain -specific sparse matrix $\mathbf{A}$ possessing its own unique properties, and to work towards a solution for that specialized problem.

As a case in point, we refer to the sparse matrix used in the implicit MHD code M3D-C^1 [1] to advance the velocity one time step using the semi-implicit algorithm. This matrix has a wide range of eigenvalues as it was formed by discritizing equations describing all MHD waves with oscillations ranging from zone-to-zone to the size of the entire device. The code can also be run in 2 "reduced MHD" regimes where only the shear-Alfven wave is present (NV=1), where that and the slow wave are present (NV=2), or where all three waves are present, (NV=3 or full MHD). The geometry is a torus, which is represented by high order C^1 continuous finite elements in 3 dimensions. The elements are structured in the direction going the long way around the torus, but unstructured in the toroidal planes at a given toroidal angle. It was recognized that the underlying wave equations are stiffest within a given toroidal plane where the zone dimensions are also smallest. This led to the development of a block-Jacobi preconditioner that uses the sparse direct solver SuperLU_dist to concurrently factor each diagonal block describing wave propagation within a given plane and to use those to form a preconditioner. The efficacy of this approach can be seen in Figure 1 where we plot the magnitude of each of the eigenvalues of the velocity matrix $\mathbf{A}$ before and after being preconditioned for NV=1, 2, and 3. It is seen that the condition number for the full MHD case has been reduced from about 10^{15} to 30 by this preconditioning technique.

While this approach with the block-Jacobi preconditioner led to a workable solution method for the M3D-C^1 velocity matrix, it is not an optimal solution for the other matrices in M3D-C^1 describing the magnetic field and pressure evolution. The pressure equation is often dominated by a large anisotropic thermal conductivity that is many orders of magnitude larger in the direction parallel to the magnetic field than in the other directions. The ill-conditioning that results from this is not well addressed by the block-Jacobi preconditioner. Also, we have yet to see much speedup when the block-Jacobi preconditioned GMRES iterative solver is implemented on GPUs as compared to CPUs as it is communication dominated.

Timeliness: There is a need for this now because matrices are getting larger and more poorly conditioned, for example those in the implicit MHD applications describing ITER. Because of the large physical size of ITER and because of its large magnetic field strength and projected temperatures and densities, the condition number of the MHD matrices will be orders of magnitude larger than those now used in modeling existing tokamaks. Also, the HPC hardware is getting more difficult to program efficiently due to the CPU/GPU processors, communication bottlenecks, and memory hierarchies.

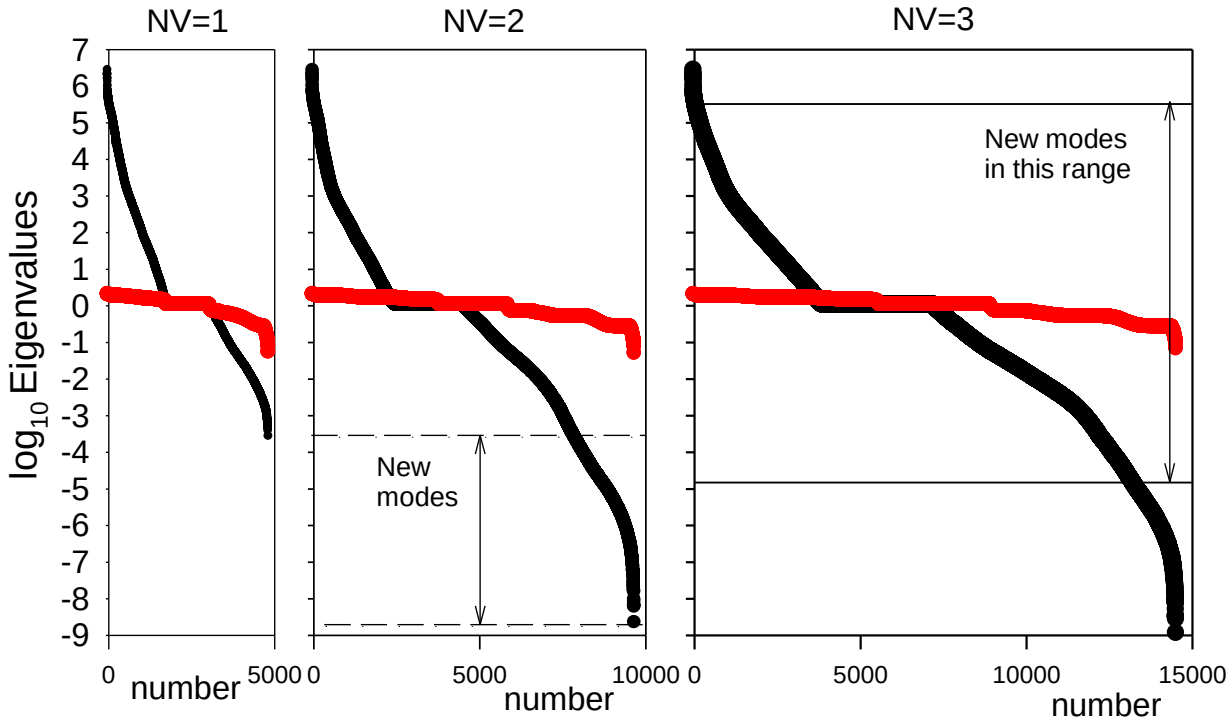

Figure 1 Plotted are all the eigenvalues of a full 3D M3D-C1 velocity matrix before (in black) and after (in red) the block-Jacobi preconditioner has been applied for 1,2 and 3 velocity variables. The ratio of the largest to the smallest eigenvalue has been reduced from about 10^{15} to approximately 30 for the 3 velocity variable case.

References:

[1] S. C. Jardin, N. Ferraro, J. Breslau, and J. Chen, "Multiple timescale calculations of sawteeth and other global macroscopic dynamics of tokamak plasmas", Comput. Science and Discovery **5** 014002 (2012)