Travis Austin

Automatic Construction of Sparse Preconditioners for High-Order Finite Element Problems

Tech-X Corporation
5621 Arapahoe Ave
Suite A
Boulder
CO 80303
austin@txcorp.com
Marian Brezina
University of Colorado at Boulder
Thomas Manteuffel
University of Colorado at Boulder
John Ruge
University of Colorado at Boulder

In recent years, the advantage to using high-order finite element methods and spectral element methods for the discretization of partial differential equations has become fully realized. The accuracy that can be achieved with high-order methods relative to the work required is more attractive than with first-order finite element methods. Furthermore, in climate modeling, researchers want a high-degree of precision at each time step in order to minimize the accumulation of errors due to long time integrations, and in MHD modeling, strong magnetic field anisotropies are not accurately resolved without high-order finite element methods. As a result of this increased interest in high-order finite element methods, it is worthwhile to reconsider the development of optimized multigrid solution methods for systems derived from high-order finite elements and spectral elements.

High-order finite elements yield much denser systems of equations requiring greater memory consumption for both the matrix and for the corresponding solver infrastructure, like AMG. For large scale MHD calculations, it has been observed that memory is a limit due to the memory consumption of the high-order matrices. Thus, there is a need for a sparser approximation of the high-order finite element systems that still yields reasonable convergence. A well-known method for generated a sparser preconditioner is to use low-order finite elements to generate a sparse approximation. In this talk, we show that these sparse preconditioners require less memory and can produce better convergence behavior for 3D problems when inverted with an AMG method. We also introduce our concept for automatically constructing these preconditioners and address the computational cost and memory consumption of such an approach.