

Combining Numerical Iterative Solvers

Alfredo Goldman, Yanik Ngoko, Denis Trystram

Abstract

Given a linear system there are several solvers which can be used to solve it. However, according to the properties of the linear system different solvers may have different convergence speeds, or may even not converge at all. Nevertheless, it can be difficult to verify these properties in practice, mainly due to rounding errors, and there are also some cases where no direct property can be used. In this special situations there is no easy choice on the best solver, so instead of determining it, we are interest in finding good combinations of the solvers.

We are interested by the resolution of sparse systems with three solvers based on three different iterative methods. The numerical methods used are the Conjugate Gradient (CG) method, the BiConjugate Gradient Stabilized (BiCGSTAB) method and the Transpose Free Quasi Minimal Residual Method (TFQMR).

To combine numerical solvers, we use an approach based on algorithm portfolio. The basic idea is to interleave iterations of numerical solvers in cycles which are executed until one solver finds a solution. We first study the combination of numerical solvers in an offline setting. In this setting we suppose that a representative set of all linear systems is available. The goal is to combine the set of numerical solvers in order to minimize the average completion time.

Then, we study the combination in an on-line setting. In this setting, we do not suppose any previous knowledge. Some heuristics are presented. The first heuristic periodically executes a same cycle of iterations for each numerical solvers. The other heuristics adapts their cycles of iterations from convergence informations gathered from previous cycles. The main difficulty in these latter cases is to define metrics and rules that will be used to evaluate the convergence of previous cycles and to define next cycles.

We experiment our approaches using the SPARSKIT library. We present comparisons among heuristics and solvers, and we also study the impact of the cycle size on the execution times.