ABSTRACT

Rodriguez, Jose S. Ph.D., Purdue University, May 2019. Solution of Large-scale Structured Optimization Problems with Schur-complement and Augmented Lagrangian Decomposition Methods. Major Professor: Carl D. Laird.

In this dissertation we develop numerical algorithms and software tools to facilitate parallel solutions of nonlinear programming (NLP) problems. In particular, we address large-scale, block-structured problems with an intrinsic decomposable configuration. These problems arise in a great number of engineering applications, including parameter estimation, optimal control, network optimization, and stochastic programming. The structure of these problems can be leveraged by optimization solvers to accelerate solutions and overcome memory limitations, and we propose variants to two classes of optimization algorithms: augmented Lagrangian (AL) schemes and Schur-complement interior-point methods.

The convergence properties of augmented Lagrangian decomposition schemes like the alternating direction method of multipliers (ADMM) and progressive hedging (PH) are well established for convex optimization but convergence guarantees in non-convex settings are still poorly understood. In practice, however, ADMM and PH often perform satisfactorily in complex non-convex NLPs. In this work, we study connections between the method of multipliers (MM), ADMM, and PH to derive benchmarking metrics that explain why PH and ADMM work in practice. We illustrate the concepts using challenging dynamic optimization problems. Our exposition seeks to establish more formalism in benchmarking ADMM, PH, and AL schemes and to motivate algorithmic improvements.

The effectiveness of nonlinear interior-point solvers for solving large-scale problems relies quite heavily on the solution of the underlying linear algebra systems.

The schur-complement decomposition is very effective for parallelizing the solution of linear systems with modest coupling. However, for systems with large number of coupling variables the schur-complement method does not scale favorably. We implement an approach that uses a Krylov solver (GMRES) preconditioned with ADMM to solve block-structured linear systems that arise in the interior-point method. We show that this ADMM-GMRES approach overcomes the well-known scalability issues of Schur decomposition.

One important drawback of using decomposition approaches like ADMM and PH is their convergence rate. Unlike Schur-complement interior-point algorithms that have super-linear convergence, augmented Lagrangian approaches typically exhibit linear and sublinear rates. We exploit connections between ADMM and the Schur-complement decomposition to derive an accelerated version of ADMM. Specifically, we study the effectiveness of performing a Newton-Raphson algorithm to compute multiplier estimates for augmented Lagrangian methods. We demonstrate using two-stage stochastic programming problems that our multiplier update achieves convergence in fewer iterations for MM on general nonlinear problems. In the case of ADMM, the newton update significantly reduces the number of subproblem solves for convex quadratic programs (QPs). Moreover, we show that using newton multiplier updates makes the method robust to the selection of the penalty parameter.

Traditionally, state-of-the-art optimization solvers are implemented in low-level programming languages. In our experience, the development of decomposition algorithms in these frameworks is challenging. They present a steep learning curve and can slow the development and testing of new numerical algorithms. To mitigate these challenges, we developed `PyNumero`, a new open source framework implemented in Python and C++. The package seeks to facilitate development of optimization algorithms for large-scale optimization within a high-level programming environment while at the same time minimizing the computational burden of using Python. The efficiency of `PyNumero` is illustrated by implementing algo-

rithms for problems arising in stochastic programming and optimal control. Timing results are presented for both serial and parallel implementations. Our computational studies demonstrate that with the appropriate balance between compiled code and Python, efficient implementations of optimization algorithms are achievable in these high-level languages.