Consensus Optimization with Automatic Variable Splitting

Tri Dao  
Yushi Wang  
EE364b: Convex Optimization II Class Project

Introduction

In many distributed optimization algorithms, for example the Alternating Directions Method of Multipliers (ADMM), the objective function of an optimization problem is assumed to split into separate additive components, or subproblems. These subproblems are only connected to each other via equality linear constraints. The general form of such a problem is called the consensus form via a process called variable splitting. We seek to convert a general convex optimization problem into the consensus form.

Variable Splitting Objectives

- Transform a general convex problem into the consensus form, with a specified number of subproblems.
- Subproblems have roughly equal size.
- Minimize the number of linear equality constraints between subproblems.

Spectral Graph Partitioning

We form a graph of components of the objective function:

- Vertices: components of the objective function.
- Edges: complicating variables.

We implemented a graph partitioning algorithm based on spectral graph theory: approximating the sparsest cut of a graph (into two pieces) through the second eigenvector of its Laplacian matrix, which can be computed efficiently. We apply this step recursively to get to the desired number of pieces/subproblems. Given a simple graph \( G = (V, E) \) with \( n \) vertices, its Laplacian matrix \( L_G \) is

\[
L = D - A,
\]

where \( D \) is the degree matrix \( \text{diag}(\text{deg}(1), \ldots, \text{deg}(n)) \) and \( A \) is the adjacency matrix of the graph.

The Laplacian has the quadratic form

\[
x^T L x = \sum_{(i,j) \in E} (x_i - x_j)^2.
\]

The eigenvector corresponding to the second smallest eigenvalue \( \lambda_2 \) partitions the graph into two pieces that approximately minimizes the number of edges cut.

Example: Sum of Squares

We plot the objective function value versus iteration for the minimum cut and 5 different random cuts.

Example: SVM

We plot the objective function value versus iteration for the minimum cut and 4 different random cuts.

Consensus Solver Implementation

After splitting the problem into subproblems, we run the ADMM algorithm:

\[
x_i^{(k+1)} := \text{argmin} \left( f_i(x_i) + y_i^T (x_i - z^{(k)}) + (\rho/2) ||x_i - z^{(k)}||_2^2 \right)
\]

\[
z^{(k+1)} := \frac{1}{N} \sum_{i=1}^N (x_i^{(k+1)} + (1/\rho) y_i^T)
\]

\[
y_i^{(k+1)} := y_i^{(k)} + \rho (x_i^{(k+1)} - z^{(k+1)}).
\]

The objective function is assumed to split into separate additive components, or subproblems. Moreover, we aim to do this in a manner that minimizes the running time of the distributed optimization algorithm.

Discussion

In terms of performance, our program is highly unoptimized, since we focused on simplicity of code design within the scope of this class. Nonetheless, we are able to solve most of our unconstrained minimization problems within an order of magnitude of CVXPY.

Our automatic splitting favors problems whose graph representations have "bottlenecks". Two benefits of splitting problems along bottlenecks:

- Reduce number of iteration.
- Reduce running time per iteration.

As more problems requires more calls to CVXPY’s solve, our implementation of automatic variable splitting does not help with problems with a single large variable (e.g. SVM), as shown.

Future Work

- Adapting value of parameter \( \rho \) dynamically.
- Accounting for variable sizes.
- Splitting of large variables.
- Optimizing performance of variable splitting step and consensus solver.