Solving Large-Scale Optimization Problems with ADMM

Jaan Tollander de Balsch
Aalto University School of Science, Department of Computer Science, {de.tollander@aalto.fi}

Background

In this report, we examine the *alternating direction method of multipliers* (ADMM), a distributed constrained convex optimization method, suitable for solving large scale optimization problems. ADMM combines the benefits of *dual decomposition* and *augmented Lagrangian* methods for constrained optimization. The dual decomposition method is a distributed variant of the *dual ascent* method possible when the objective function is *separable*. The augmented Lagrangian brings robustness to the *dual ascent* method and yields convergence without assumptions such as strict convexity finiteness of the objective function.

Boyd et al. (2010) covers the ADMM extensively. This report heavily relies on the paper, especially sections 1, 2, 3, and 5.

Algorithm

The ADMM algorithm solves problems in the form

$$\begin{aligned} \text{minimize} \quad & f(x) + g(z) \\ \text{subject to} \quad & Ax + Bz = c \end{aligned}$$

with variables $x \in \mathbf{R}^n$ and $z \in \mathbf{R}^m$, and parameters $A \in \mathbf{R}^{p\times n}$, $B \in \mathbf{R}^{p\times m}$, and $c \in \mathbf{R}^p$.

The augmented Lagrangian takes form

$$L_\rho(x,z,v) = f(x) + g(z) + v^T(Ax + Bz - c) + (\rho/2)\|Ax + Bz - c\|_2^2,$$

where $v \in \mathbf{R}^m$ is the *dual variable* and $\rho > 0$ is referred as the *penalty parameter*.

The ADMM algorithm consists of the following iterations

$$\begin{aligned} x_{k+1} &= \operatorname{argmin}_x L_\rho(x, z_k, v_k) \\ z_{k+1} &= \operatorname{argmin}_z L_\rho(x_{k+1}, z, v_k) \\ v_{k+1} &= v_k + \rho(Ax_{k+1} + Bz_{k+1} - c). \end{aligned}$$

The algorithm updates the variables x and z in alternating fashion, hence the term *alternating direction.*

Convergence and Stopping Criteria

A *residual* at iteration k is defined as

$$r_k = Ax_k + Bz_k - C.$$

We assume that the functions f and g are closed, proper and convex, and the unaugmented Lagrangian has as saddle point, then ADMM convergences as the iterations $k \to \infty$ as follows.

- The *residual converges* towards zero, that is, $r \to 0$.
- The *objective value converges* towards the optimal.
- The *dual variable converges* towards the dual optimal point.

In practical implementation, ADMM will stop when the residual is below a tolerance $\epsilon > 0$, that is, $r < \epsilon$. Because ADMM converges slowly to high accuracy but can reach a modest accuracy with relatively few iterations, the tolerance ϵ is set relatively high compared to methods converge to high accuracy. However, many practical applications do not require high accuracy to produce good results. For example, in machine learning, a slightly more optimal model fit would not produce much better predictions.

Generic Algorithm

The generic constrained optimization problem defined as

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & x \in X. \end{aligned}$$

We can write the problem in ADMM form as

$$\begin{aligned} \text{minimize} \quad & f(x) + g(z) \\ \text{subject to} \quad & x - z = 0 \end{aligned}$$

where g is the indicator function of X.

The augmented Lagrangian function take form

$$L_\rho(x, z, v) = f(x) + g(z) + v^T(x - z) + (\rho/2)\|x - z\|_2^2.$$

The ADMM algorithm consists of the following iterations

$$\begin{aligned} x_{k+1} &= \operatorname{argmin}_x L_\rho(x, z_k, v_k) \\ z_{k+1} &= \operatorname{argmin}_z L_\rho(x_{k+1}, z, v_k) \\ v_{k+1} &= v_k + \rho(x_{k+1} - z_{k+1}). \end{aligned}$$

We will use the generic form when applying ADMM to the problem in the next section.

Applications

In this section, we derive the ADMM formulation for the *stochastic capacity expansion problem.* The objective function is defined as

$$\sum_{i\in I} c_i x_i + \sum_{s\in S} p_s \left(\sum_{i\in I} \sum_{j\in J} f_{i,j} y_{i,j,s} + \sum_{j\in J} q_j u_{j,s} \right)$$

with parameters $c_i, f_{i,j}, q_j$ and probabilities p_s.

The problem is separable in terms of scenarios S. We need to introduce the scenario dependent variables x_s for the variable x such that

$$\sum_{s\in S} p_s f(x_s) = \sum_{s\in S} p_s \left(\sum_{i\in I} c_i x_{i,s} + \sum_{i\in I} \sum_{j\in J} f_{i,j} y_{i,j,s} + \sum_{j\in J} q_j u_{j,s} \right).$$

The augmented Lagrangian is form as

$$L_\rho(x_s, z, v_s) = f(x_s) + g(z) + v_s^T (x_s - z) + (\rho/2)\|x_s - z\|_2^2.$$

We have $g(z) = 0$ and $v^T z = 0$

$$L_\rho(x_s, z, v_s) = f(x_s) + v_s^T x_s + (\rho/2)\|x_s - z\|_2^2.$$

The ADMM algorithm consists of the following iterations

$$\begin{aligned} x_s^{k+1} &= \operatorname{argmin}_{x_s} L_\rho(x_s, z^k, v_s^k) \\ z^{k+1} &= \operatorname{argmin}_z \sum_{s\in S} p_s L_\rho(x_s^{k+1}, z, v_s^k) \\ v_s^{k+1} &= v_s^k + \rho(x_s^{k+1} - z^{k+1}). \end{aligned}$$

with a stopping criterion

$$\sum_{s\in S} p_s \rho \|x_s^{k+1} - z^k\|_2 < \epsilon,$$

where $\epsilon > 0$.

Since z is unconstrained, we can obtain the z update by taking the gradient and setting it to zero.

$$\begin{aligned} \nabla_z L_\rho(x_{k+1}, z, v_k) &= 0 \\ \sum_{s\in S} p_s \rho (x_s^{k+1} - z) &= 0 \end{aligned}$$

with $\sum_{s\in S} p_s = 1$ yields

$$z^{k+1} = \sum_{s\in S} p_s x_s^{k+1}.$$

Discussion and Conclusions

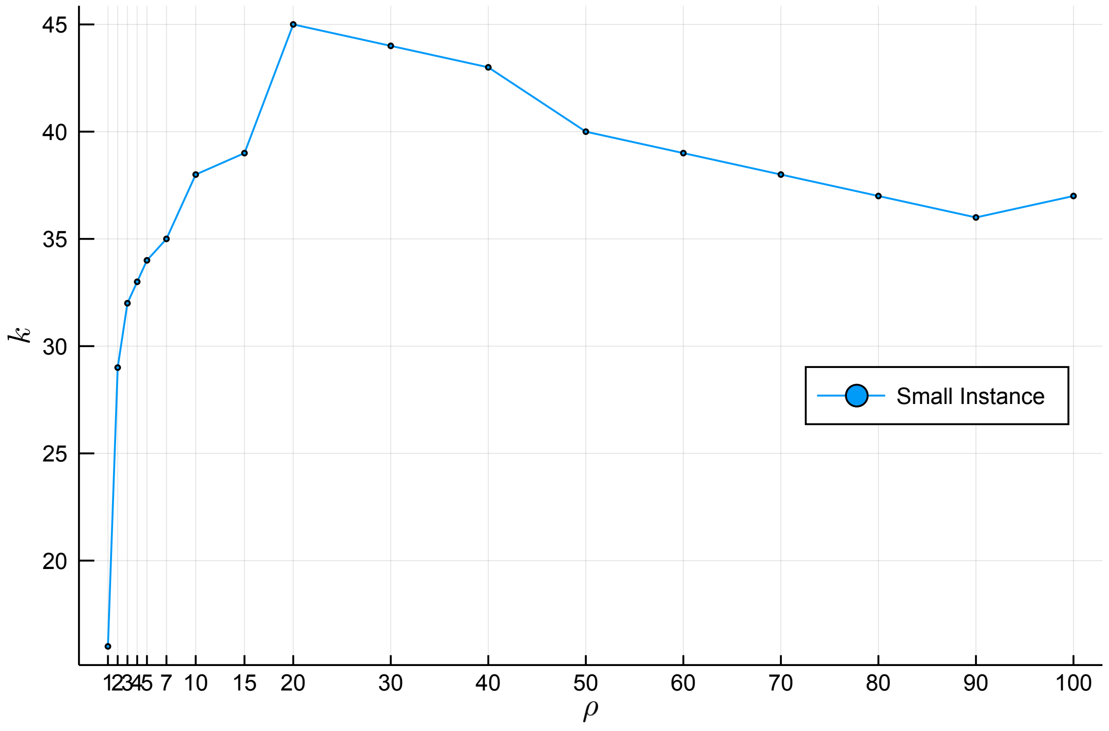

Figure 1: Small instance

We experimented with two instances of the stochastic capacity expansion problem. The small instance has $|S| = 100$ and a large instance has $|S| = 500$, and both have $|I| = 20$ and $|J| = 30$.

For both instances, we analyzed the number of iterations k as a function of the penalty parameter ρ for convergence with solution tolerance $\epsilon = 10^{-1}$. Figures 1 and 2 visualize how the number of iterations grows as the penalty parameters grows for both instances. In other words, the performance of ADMM is best with a lower penalty parameter.

Table 1: Comparison of the performance of the deterministic method and ADMM.

Instance	Deterministic	ADMM ($\rho = 1$)
Small	20 s	16 s
Large	205 s	106 s

We also compared the performance of the deterministic method, that is, solving the problem using the interior point method, and ADMM. As can be seen in table 1, ADMM is faster than the deterministic method for the large instance. ADMM is faster because it parallelizes the problem and uses multiple processors for computing the solution.

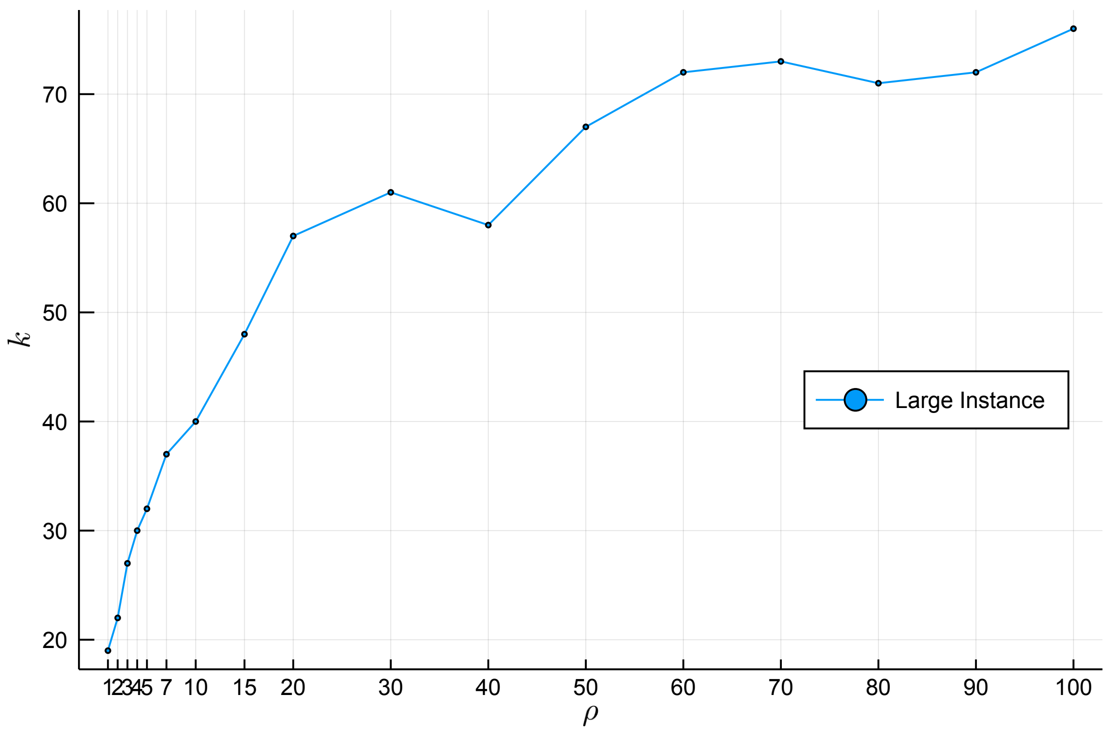

Figure 2: Large instance

References

Boyd, S., Parikh, N., Chu, E., Peleato, B. and Eckstein, J., 2010. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1), pp.1–122.