Alternating Direction Method of Multipliers

Concept

- It was first introduced in the mid-1970s by Gabay, Mercier, Glowinski, and Marrocco, though similar ideas emerged as early as the mid-1950s. The algorithm was studied throughout the 1980s, and by the mid-1990s
- It takes the form of a *decomposition-coordination* procedure, in which the solutions to small local subproblems are coordinated to find a solution to a large global problem
- It can be viewed as an attempt to blend the benefits of **dual decomposition** and **augmented Lagrangian methods** (also called method of multipliers) for constrained optimization

https://en.wikipedia.org/wiki/Augmented_Lagrangian_method

Dual problem

- convex equality constrained optimization problem

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax = b \end{array}$$

- Lagrangian: $L(x, y) = f(x) + y^T(Ax - b)$

y is dual variable or Lagrange multiplier

- dual function: $g(y) = \inf_x L(x, y)$
- dual problem: maximize $g(y)$
- recover $x^\star = \mathrm{argmin}_x L(x, y^\star)$

Assuming that strong duality holds, the optimal values of the primal and dual problems are the same

Dual ascent

- gradient method for dual problem: $y^{k+1} = y^k + \alpha^k \nabla g(y^k)$
- $\nabla g(y^k) = A\tilde{x} - b$, where $\tilde{x} = \operatorname{argmin}_x L(x, y^k)$
- dual ascent method is

$$
\begin{aligned}
x^{k+1} &:= \operatorname{argmin}_x L(x, y^k) && \text{// } x\text{-minimization} \\
y^{k+1} &:= y^k + \alpha^k (Ax^{k+1} - b) && \text{// dual update}
\end{aligned}
$$

$\alpha^k > 0$ is a step size   residual for the equality constraint

Dual decomposition

- suppose f is separable: $x_i \in \mathbf{R}^{n_i}$ are subvectors of x.

$$f(x) = f_1(x_1) + \cdots + f_N(x_N), \quad x = (x_1, \ldots, x_N)$$

- then L is separable in x: $L(x,y) = L_1(x_1,y) + \cdots + L_N(x_N,y) - y^T b$,

$$L_i(x_i, y) = f_i(x_i) + y^T A_i x_i \qquad A = [A_1 \ \cdots \ A_N],$$

- x-minimization in dual ascent splits into N separate minimizations

$$x_i^{k+1} \ := \ \operatorname*{argmin}_{x_i} L_i(x_i, y^k) \quad y^{k+1} \ := \ y^k + \alpha^k (\textstyle\sum_{i=1}^N A_i x_i^{k+1} - b)$$

which can be carried out in parallel

Each iteration requires a *broadcast and a gather* operation

Augmented Lagrangian methods (method of multipliers)

- Transform the primal problem

$$\begin{array}{ll} \text{minimize} & f(x) + (\rho/2)\|Ax - b\|_2^2 \\ \text{subject to} & Ax = b. \end{array}$$

$\rho > 0$ penalty parameter

- use **augmented Lagrangian** (Hestenes, Powell 1969), $\rho > 0$

$$L_\rho(x, y) = f(x) + y^T(Ax - b) + (\rho/2)\|Ax - b\|_2^2$$

- method of multipliers (Hestenes, Powell; analysis in Bertsekas 1982)

$$\begin{aligned} x^{k+1} &:= \underset{x}{\operatorname{argmin}}\, L_\rho(x, y^k) \\ y^{k+1} &:= y^k + \rho(Ax^{k+1} - b) \end{aligned}$$

(note specific dual update step length ρ)

ADMM

- ADMM problem form (with f, g convex)

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

  - two sets of variables, with separable objective

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

- Augment the objective

$$\begin{array}{ll} \min\limits_{x} & f(x) + g(z) + \dfrac{\rho}{2}\|Ax + Bz - c\|_2^2 \\ \text{subject to} & Ax + Bz = c \end{array}$$

- Augmented Lagrangian

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

ADMM

- ADMM repeats the steps, for k=1, 2, 3

$$
\begin{aligned}
x^{k+1} &:= \operatorname{argmin}_x L_\rho(x, z^k, y^k) && \text{// } x\text{-minimization} \\
z^{k+1} &:= \operatorname{argmin}_z L_\rho(x^{k+1}, z, y^k) && \text{// } z\text{-minimization} \\
y^{k+1} &:= y^k + \rho(Ax^{k+1} + Bz^{k+1} - c) && \text{// dual update}
\end{aligned}
$$

- Note that if we minimized over x and z jointly, reduces to method of multipliers
- Else in an alternating fashion, which accounts for the term *alternating direction*

Example: Lasso regression

- Problem

$$\text{minimize} \quad (1/2)\|Ax - b\|_2^2 + \lambda\|x\|_1$$

- ADMM form

$$\begin{array}{ll} \text{minimize} & (1/2)\|Ax - b\|_2^2 + \lambda\|z\|_1 \\ \text{subject to} & x - z = 0 \end{array}$$

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

$$f(x) = (1/2)\|Ax - b\|_2^2$$

$$g(z) = \lambda\|z\|_1.$$

- Update

$$\begin{aligned} x^{k+1} &:= (A^TA + \rho I)^{-1}(A^Tb + \rho(z^k - u^k)) \\ z^{k+1} &:= S_{\lambda/\rho}(x^{k+1} + u^k) \\ u^{k+1} &:= u^k + x^{k+1} - z^{k+1}. \end{aligned}$$

Example: Lasso regression

➢ Soft thresholding operator S

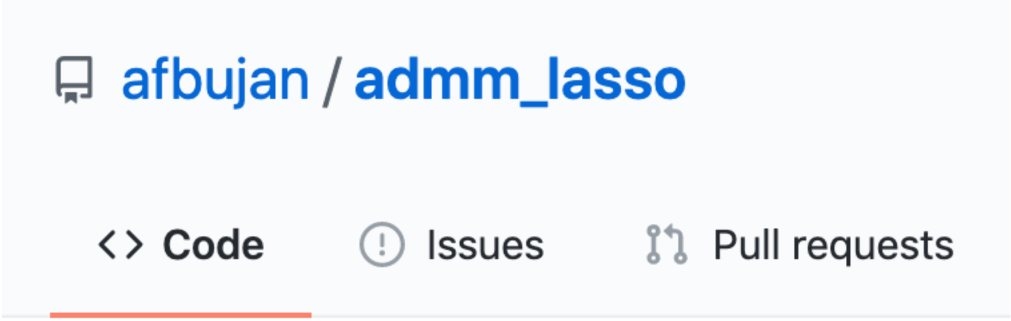
$$S_\kappa(a) = \begin{cases} a-\kappa & a>\kappa \\ 0 & |a|\le\kappa \\ a+\kappa & a<-\kappa, \end{cases}$$

$$S_\kappa(a) = (a-\kappa)_+ - (-a-\kappa)_+.$$

Code

ADMM

```
lambda = 1;
rho = 1/lambda;

tic;

x = zeros(n,1);
z = zeros(n,1);
u = zeros(n,1);

[L U] = factor(A, rho);

for k = 1:MAX_ITER

    % x-update
    q = Atb + rho*(z - u);
    if m >= n
       x = U \ (L \ q);
    else
       x = lambda*(q - lambda*(A'*(U \ ( L \ (A*q) ))));
    end
```



https://web.stanford.edu/~boyd/papers/prox_algs/lasso.html#8
https://github.com/afbujan/admm_lasso/blob/master/lasso_admm.py

Thanks

算法 数学 机器学习 最优化 运筹学

交替方向乘子法（ADMM）算法的流程和原理是怎样的?

关注问题 写回答 邀请回答 好问题 21 1 条评论 分享

https://www.zhihu.com/question/36566112

Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein

Foundations and Trends in Machine Learning, 3(1):1–122, 2011. (Original draft posted November 2010.)

- Paper
- Matlab examples
- MPI example
- ADMM links and resources

https://stanford.edu/~boyd/papers/admm_distr_stats.html