交替方向乘子法(ADMM)

宋晓良

大连理工大学数学科学学院

教材《最优化：建模、算法与理论》配套电子教案

http://bicmr.pku.edu.cn/~wenzw/optbook.html

提纲

典型问题形式

考虑如下凸问题：

$$\begin{aligned} \min_{x_1,x_2} \quad & f_1(x_1)+f_2(x_2), \\ \text{s.t.} \quad & A_1x_1+A_2x_2=b, \end{aligned} \tag{1}$$

- f_1,f_2是适当的闭凸函数，但不要求是光滑的，$x_1\in\mathbb{R}^n, x_2\in\mathbb{R}^m$, $A_1\in\mathbb{R}^{p\times n}, A_2\in\mathbb{R}^{p\times m}$, $b\in\mathbb{R}^p$.
- 问题特点：目标函数可以分成彼此分离的两块，但是变量被线性约束结合在一起. 常见的一些无约束和带约束的优化问题都可以表示成这一形式。

问题形式举例

- 可以分成两块的无约束优化问题

$$\min_{x} \quad f_1(x) + f_2(x).$$

引入一个新的变量z并令$x = z$，将问题转化为

$$\begin{aligned} \min_{x,z} \quad & f_1(x) + f_2(z), \\ \text{s.t.} \quad & x - z = 0. \end{aligned}$$

- 带线性变换的无约束优化问题

$$\min_{x} \quad f_1(x) + f_2(Ax).$$

可以引入一个新的变量z，令$z = Ax$，则问题变为

$$\begin{aligned} \min_{x,z} \quad & f_1(x) + f_2(z), \\ \text{s.t.} \quad & Ax - z = 0. \end{aligned}$$

问题形式举例

- 凸集$C\subset\mathbb{R}^n$上的约束优化问题

$$
\begin{aligned}
\min_{x} \quad & f(x), \\
\text{s.t.} \quad & Ax\in C,
\end{aligned}
$$

$I_C(z)$是集合C的示性函数，引入约束$z=Ax$，那么问题转化为

$$
\begin{aligned}
\min_{x,z} \quad & f(x)+I_C(z), \\
\text{s.t.} \quad & Ax-z=0.
\end{aligned}
$$

- 全局一致性问题

$$
\min_{x} \quad \sum_{i=1}^{N}\phi_i(x).
$$

令$x=z$，并将x复制N份，分别为x_i，那么问题转化为

$$
\begin{aligned}
\min_{x_i,z} \quad & \sum_{i=1}^{N}\phi_i(x_i), \\
\text{s.t.} \quad & x_i-z=0,\ i=1,2,\cdots,N.
\end{aligned}
$$

增广拉格朗日函数法

- 首先写出问题(1)的增广拉格朗日函数

$$
\begin{aligned}
L_\rho(x_1, x_2, y) = & f_1(x_1) + f_2(x_2) + y^{\mathrm{T}}(A_1x_1 + A_2x_2 - b) \\
& + \frac{\rho}{2}\|A_1x_1 + A_2x_2 - b\|_2^2,
\end{aligned} \tag{2}
$$

其中$\rho > 0$是二次罚项的系数.

- 常见的求解带约束问题的增广拉格朗日函数法为如下更新:

$$
(x_1^{k+1}, x_2^{k+1}) = \underset{x_1, x_2}{\operatorname{argmin}} L_\rho(x_1, x_2, y^k), \tag{3}
$$

$$
y^{k+1} = y^k + \tau\rho(A_1x_1^{k+1} + A_2x_2^{k+1} - b), \tag{4}
$$

其中τ为步长.

Alternating direction method of multipliers, ADMM

- 交替方向乘子法的基本思路: 第一步迭代(3)同时对x_1和x_2进行优化有时候比较困难，而固定一个变量求解关于另一个变量的极小问题可能比较简单，因此我们可以考虑对x_1和x_2交替求极小

- 其迭代格式可以总结如下：

$$x_1^{k+1} = \underset{x_1}{\operatorname{argmin}}\, L_\rho(x_1, x_2^k, y^k), \tag{5}$$

$$x_2^{k+1} = \underset{x_2}{\operatorname{argmin}}\, L_\rho(x_1^{k+1}, x_2, y^k), \tag{6}$$

$$y^{k+1} = y^k + \tau\rho(A_1 x_1^{k+1} + A_2 x_2^{k+1} - b), \tag{7}$$

其中τ 为步长，通常取值于$\left(0, \frac{1+\sqrt{5}}{2}\right]$

原问题最优性条件

- 因为f_1,f_2均为闭凸函数，约束为线性约束，所以当Slater 条件成立时，可以使用凸优化问题的KKT 条件来作为交替方向乘子法的收敛准则．问题(1)的拉格朗日函数为

$$L(x_1,x_2,y)=f_1(x_1)+f_2(x_2)+y^{\mathrm{T}}(A_1x_1+A_2x_2-b).$$

- 根据最优性条件定理，若x_1^*,x_2^*为问题(1)的最优解，y^*为对应的拉格朗日乘子，则以下条件满足：

$$0\in\partial_{x_1}L(x_1^*,x_2^*,y^*)=\partial f_1(x_1^*)+A_1^{\mathrm{T}}y^*, \tag{8a}$$
$$0\in\partial_{x_2}L(x_1^*,x_2^*,y^*)=\partial f_2(x_2^*)+A_2^{\mathrm{T}}y^*, \tag{8b}$$
$$A_1x_1^*+A_2x_2^*=b. \tag{8c}$$

在这里条件(8c)又称为原始可行性条件，条件(8a)和条件(8b)又称为对偶可行性条件．

ADMM单步迭代最优性条件

- 由x_2的更新步骤

$$x_2^k = \underset{x}{\operatorname{argmin}} \left\{ f_2(x) + \frac{\rho}{2} \left\| A_1 x_1^k + A_2 x - b + \frac{y^{k-1}}{\rho} \right\|^2 \right\},$$

根据最优性条件不难推出

$$0 \in \partial f_2(x_2^k) + A_2^{\mathrm{T}}[y^{k-1} + \rho(A_1 x_1^k + A_2 x_2^k - b)]. \qquad (9)$$

当$\tau = 1$时，根据(7)可知上式方括号中的表达式就是y^k，最终有

$$0 \in \partial f_2(x_2^k) + A_2^{\mathrm{T}} y^k,$$

- 由x_1的更新公式

$$x_1^k = \underset{x}{\operatorname{argmin}} \left\{ f_1(x) + \frac{\rho}{2} \|A_1 x + A_2 x_2^{k-1} - b + \frac{y^{k-1}}{\rho}\|^2 \right\},$$

假设子问题能精确求解，根据最优性条件

$$0 \in \partial f_1(x_1^k) + A_1^{\mathrm{T}}[\rho(A_1 x_1^k + A_2 x_2^{k-1} - b) + y^{k-1}].$$

ADMM单步迭代最优性条件

- 根据ADMM 的第三式(7)取$\tau = 1$有

$$0 \in \partial f_1(x_1^k) + A_1^{\mathrm{T}}(y^k + \rho A_2(x_2^{k-1} - x_2^k)). \tag{10}$$

对比条件(8a)可知多出来的项为$A_1^{\mathrm{T}} A_2(x_2^{k-1} - x_2^k)$。因此要检测对偶可行性只需要检测残差

$$s^k = A_1^{\mathrm{T}} A_2(x_2^{k-1} - x_2^k)$$

- 综上当x_2更新取到精确解且$\tau = 1$时，判断ADMM 是否收敛只需要检测前述两个残差r^k, s^k是否充分小：

$$\begin{aligned} 0 \approx \|r^k\| &= \|A_1 x_1^k + A_2 x_2^k - b\| \quad &\text{（原始可行性）}, \\ 0 \approx \|s^k\| &= \|A_1^{\mathrm{T}} A_2(x_2^{k-1} - x_2^k)\| \quad &\text{（对偶可行性）}. \end{aligned} \tag{11}$$

提纲

线性化

- 线性化技巧使用近似点项对子问题目标函数进行二次近似.
- 不失一般性，我们考虑第一个子问题，即

$$\min_{x_1} \quad f_1(x_1) + \frac{\rho}{2}\|A_1x_1 - v^k\|^2, \tag{12}$$

其中$v^k = b - A_2x_2^k - \frac{1}{\rho}y^k$.

- 当子问题目标函数可微时，线性化将问题(12)变为

$$x_1^{k+1} = \mathop{\mathrm{argmin}}_{x_1} \left\{ \left(\nabla f_1(x_1^k) + \rho A_1^{\mathrm{T}}\left(A_1x_1^k - v^k\right)\right)^{\mathrm{T}} x_1 + \frac{1}{2\eta_k}\|x_1 - x^k\|_2^2 \right\},$$

其中η_k是步长参数，这等价于做一步梯度下降.

- 当目标函数不可微时，可以考虑只将二次项线性化，即

$$x_1^{k+1} = \mathop{\mathrm{argmin}}_{x_1} \left\{ f_1(x_1) + \rho\left(A_1^{\mathrm{T}}(A_1x_1^k - v^k)\right)^{\mathrm{T}} x_1 + \frac{1}{2\eta_k}\|x_1 - x^k\|_2^2 \right\},$$

这等价于做一步近似点梯度步.

缓存分解

- 如果目标函数中含二次函数，例如$f_1(x_1) = \frac{1}{2}\|Cx_1 - d\|_2^2$，那么针对$x_1$的更新(5)等价于求解线性方程组

$$(C^{\mathrm{T}}C + \rho A_1^{\mathrm{T}}A_1)x_1 = C^{\mathrm{T}}d + \rho A_1^{\mathrm{T}}v^k.$$

- 虽然子问题有显式解，但是每步求解的复杂度仍然比较高，这时候可以考虑用**缓存分解**的方法．首先对$C^{\mathrm{T}}C + \rho A_1^{\mathrm{T}}A_1$进行Cholesky 分解并缓存分解的结果，在每步迭代中只需要求解简单的三角形方程组
- 当ρ发生更新时，就要重新进行分解．特别地，当$C^{\mathrm{T}}C + \rho A_1^{\mathrm{T}}A_1$一部分容易求逆，另一部分是低秩的情形时，可以用SMW公式来求逆．

- 有时候为了方便求解子问题，可以用一个性质好的矩阵D近似二次项$A_1^\mathrm{T}A_1$，此时子问题(12)替换为

$$\begin{aligned} x_1^{k+1} \quad = \quad & \underset{x_1}{\operatorname{argmin}} \left\{ f_1(x_1) + \frac{\rho}{2}\|A_1x_1 - v^k\|_2^2 \right. \\ & \left. + \frac{\rho}{2}(x_1 - x^k)^\mathrm{T}(D - A_1^\mathrm{T}A_1)(x_1 - x^k) \right\}. \end{aligned}$$

  这种方法也称为**优化转移**.

- 通过选取合适的D，当计算$\underset{x_1}{\operatorname{argmin}} \left\{ f_1(x_1) + \frac{\rho}{2} x_1^\mathrm{T} D x_1 \right\}$明显比计算$\underset{x_1}{\operatorname{argmin}} \{ f_1(x_1) + \frac{\rho}{2} x_1^\mathrm{T} A_1^\mathrm{T} A_1 x_1 \}$要容易时，优化转移可以极大地简化子问题的计算. 特别地，当$D = \frac{\eta_k}{\rho} I$时，优化转移等价于做单步的近似点梯度步.

二次罚项系数的动态调节

- 原始可行性和对偶可行性分别用$\|r^k\|$和$\|s^k\|$度量.
- 求解过程中二次罚项系数ρ太大会导致原始可行性$\|r^k\|$下降很快，但是对偶可行性$\|s^k\|$下降很慢；二次罚项系数太小，则会有相反的效果．这样都会导致收敛比较慢或得到的解的可行性很差．
- 一个自然的想法是在每次迭代时动态调节惩罚系数ρ的大小，从而使得原始可行性和对偶可行性能够以比较一致的速度下降到零．一个简单有效的方式是令

$$\rho^{k+1} = \begin{cases} \gamma_p \rho^k, & \|r^k\| > \mu\|s^k\|, \\ \rho^k/\gamma_d & \|s^k\| > \mu\|r^k\|, \\ \rho^k, & \text{其他}, \end{cases}$$

其中$\mu > 1, \gamma_p > 1, \gamma_d > 1$是参数, 常见的选择为$\mu = 10, \gamma_p = \gamma_d = 2$. 在迭代过程中将原始可行性$\|r^k\|$和对偶可行性$\|s^k\|$保持在彼此的$\mu$倍内．如果发现$\|r^k\|$或$\|s^k\|$下降过慢就应该相应增大或减小二次罚项系数$\rho^k$．

超松弛

- 在(6)式与(7)式中，$A_1x_1^{k+1}$可以被替换为
$$\alpha_k A_1 x_1^{k+1} + (1-\alpha_k)(A_2x_2^k - b),$$
其中$\alpha_k \in (0,2)$是一个松弛参数．
- 当$\alpha_k > 1$时，这种技巧称为超松弛；当$\alpha_k < 1$时，这种技巧称为欠松弛．实验表明$\alpha_k \in [1.5, 1.8]$的超松弛可以提高收敛速度．

多块问题的ADMM

- 考虑有多块变量的情形

$$\begin{aligned} \min_{x_1,x_2,\cdots,x_N} \quad & f_1(x_1)+f_2(x_2)+\cdots+f_N(x_N), \\ \text{s.t.} \quad & A_1x_1+A_2x_2+\cdots+A_Nx_N=b. \end{aligned} \tag{13}$$

  这里$f_i(x_i)$是闭凸函数，$x_i \in \mathbb{R}^{n_i}, A_i \in \mathbb{R}^{m\times n_i}$.

- 同样写出增广拉格朗日函数$L_\rho(x_1,x_2,\cdots,x_N,y)$，相应的多块ADMM 迭代格式为

$$\begin{aligned} x_1^{k+1} &= \operatorname*{argmin}_x L_\rho(x,x_2^k,\cdots,x_N^k,y^k), \\ x_2^{k+1} &= \operatorname*{argmin}_x L_\rho(x_1^{k+1},x,\cdots,x_N^k,y^k), \\ &\cdots\cdots\cdots\cdots\cdot \\ x_N^{k+1} &= \operatorname*{argmin}_x L_\rho(x_1^{k+1},x_2^{k+1},\cdots,x,y^k), \\ y^{k+1} &= y^k+\tau\rho(A_1x_1^{k+1}+A_2x_2^{k+1}+\cdots+A_Nx_N^{k+1}-b), \end{aligned}$$

  其中$\tau \in (0,(\sqrt{5}+1)/2)$为步长参数.

提纲

LASSO 问题的Primal 形式

- LASSO 问题

$$\min \quad \mu\|x\|_1 + \frac{1}{2}\|Ax - b\|^2.$$

转换为标准问题形式：

$$\begin{aligned} \min_{x,z} \quad & \frac{1}{2}\|Ax - b\|^2 + \mu\|z\|_1, \\ \text{s.t.} \quad & x = z. \end{aligned}$$

- 交替方向乘子法迭代格式为

$$\begin{aligned} x^{k+1} &= \operatorname*{argmin}_x \left\{ \frac{1}{2}\|Ax - b\|^2 + \frac{\rho}{2}\|x - z^k + y^k/\rho\|_2^2 \right\}, \\ &= (A^{\mathrm{T}}A + \rho I)^{-1}(A^{\mathrm{T}}b + \rho z^k - y^k), \\ z^{k+1} &= \operatorname*{argmin}_z \left\{ \mu\|z\|_1 + \frac{\rho}{2}\|x^{k+1} - z + y^k/\rho\|^2 \right\}, \\ &= \operatorname{prox}_{(\mu/\rho)\|\cdot\|_1}\left(x^{k+1} + y^k/\rho\right), \\ y^{k+1} &= y^k + \tau\rho(x^{k+1} - z^{k+1}). \end{aligned}$$

LASSO 问题的Primal 形式

- 注意，因为$\rho > 0$，所以$A^{\mathrm{T}}A + \rho I$总是可逆的．x迭代本质上是计算一个岭回归问题（ℓ_2范数平方正则化的最小二乘问题）；而对z的更新为ℓ_1范数的邻近算子，同样有显式解．在求解x迭代时，若使用固定的罚因子ρ，我们可以缓存矩阵$A^{\mathrm{T}}A + \rho I$的初始分解，从而减小后续迭代中的计算量．
- 需要注意的是，在LASSO 问题中，矩阵$A \in \mathbb{R}^{m \times n}$通常有较多的列（即$m \ll n$），因此$A^{\mathrm{T}}A \in \mathbb{R}^{n \times n}$是一个低秩矩阵，二次罚项的作用就是将$A^{\mathrm{T}}A$增加了一个正定项．该ADMM 主要运算量来自更新$x$变量时求解线性方程组，复杂度为$O(n^3)$（若使用缓存分解技术或SMW 公式则可进一步降低每次迭代的运算量）

LASSO 问题的对偶形式

- 考虑LASSO 问题的对偶问题

$$\begin{aligned} \min \quad & b^{\mathrm{T}}y + \tfrac{1}{2}\|y\|^2, \\ \text{s.t.} \quad & \|A^{\mathrm{T}}y\|_\infty \le \mu. \end{aligned} \tag{14}$$

- 引入约束$A^{\mathrm{T}}y+z=0$，可以得到如下等价问题：

$$\begin{aligned} \min \quad & \underbrace{b^{\mathrm{T}}y + \frac{1}{2}\|y\|^2}_{f(y)} + \underbrace{I_{\|z\|_\infty \le \mu}(z)}_{h(z)}, \\ \text{s.t.} \quad & A^{\mathrm{T}}y + z = 0. \end{aligned} \tag{15}$$

- 对约束$A^{\mathrm{T}}y+z=0$引入乘子x，对偶问题的增广拉格朗日函数为

$$L_\rho(y,z,x) = b^{\mathrm{T}}y + \frac{1}{2}\|y\|^2 + I_{\|z\|_\infty \le \mu}(z) - x^{\mathrm{T}}(A^{\mathrm{T}}y+z) + \frac{\rho}{2}\|A^{\mathrm{T}}y+z\|^2.$$

- 当固定y, x时，对z的更新即向无穷范数球$\{z \| \|z\|_\infty \leq \mu\}$做欧几里得投影，即将每个分量截断在区间$[-\mu, \mu]$中；当固定$z, x$时，对$y$的更新即求解线性方程组

$$(I + \rho AA^{\mathrm{T}})y = A(x^k - \rho z^{k+1}) - b.$$

- 因此得到ADMM 迭代格式为

$$\begin{aligned}
z^{k+1} &= \mathcal{P}_{\|z\|_\infty \leq \mu}\left(x^k/\rho - A^{\mathrm{T}} y^k\right), \\
y^{k+1} &= (I + \rho AA^{\mathrm{T}})^{-1}\Big(A(x^k - \rho z^{k+1}) - b\Big), \\
x^{k+1} &= x^k - \tau\rho(A^{\mathrm{T}} y^{k+1} + z^{k+1}).
\end{aligned}$$

- 虽然ADMM 应用于对偶问题也需要求解一个线性方程组，但由于LASSO 问题的特殊性（$m \ll n$），求解y更新的线性方程组需要的计算量是$O(m^3)$，使用缓存分解技巧后可进一步降低至$O(m^2)$，这大大小于针对原始问题的ADMM．

References

Douglas-Rachford method, ADMM, Spingarn's method

- J. E. Spingarn, *Applications of the method of partial inverses to convex programming: decomposition*, Mathematical Programming (1985)
- J. Eckstein and D. Bertsekas, *On the Douglas-Rachford splitting method and the proximal algorithm for maximal monotone operators*, Mathematical Programming (1992)
- P.L. Combettes and J.-C. Pesquet, *A Douglas-Rachford splitting approach to nonsmooth convex variational signal recovery*, IEEE Journal of Selected Topics in Signal Procesing (2007)
- S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, *Distributed optimization and statistical learning via the alternating direction method of multipliers* (2010)
- N. Parikh, S. Boyd, *Block splitting for distributed optimization* (2013)

image deblurring: the example is taken from
D. O'Connor and L. Vandenberghe, *Primal-dual decomposition by operator splitting and applications to image deblurring* (2014)