Dual Ascent and ADMM

Dual ascent

- Consider

$$\min_{\mathbf{x}} f(\mathbf{x})$$

$$\text{s.t. } A\mathbf{x} = b$$

- We have
  - $L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda^\top (A\mathbf{x} - b)$
  - $L^*(\lambda) = \inf_{\mathbf{x}} L(\mathbf{x}, \lambda)$
    (under strong duality, infimum is attained)
  - Recapping definition of the convex conjugate function $f^*(\mathbf{h}) = \sup_{\mathbf{x}} \mathbf{h}^T\mathbf{x} - f(\mathbf{x})$

    $L^*(\lambda) = \inf_{\mathbf{x}} L(\mathbf{x}, \lambda) =$

- Consider

$$\min_{\mathbf{x}} f(\mathbf{x})$$

$$\text{s.t. } A\mathbf{x} = b$$

- We have
  - $L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda^\top (A\mathbf{x} - b)$
  - $L^*(\lambda) = \inf_{\mathbf{x}} L(\mathbf{x}, \lambda)$
    (under strong duality, infimum is attained)
  - Recapping definition of the convex conjugate function $f^*(\mathbf{h}) = \sup_{\mathbf{x}} \mathbf{h}^T\mathbf{x} - f(\mathbf{x})$

    $L^*(\lambda) = \inf_{\mathbf{x}} L(\mathbf{x}, \lambda) = -f^*(-A^T\lambda) - \mathbf{b}^T\lambda$ For information only

Idea of dual ascent

1. Initialize $\lambda^{(0)}$
2. Iteratively
   1. $\mathbf{x}^{k+1} = \operatorname{argmin}_{\mathbf{x}} L(\mathbf{x}, \lambda^k)$
   2. Gradient ascent for dual maximization problem: $d^* = \max_{\lambda \geq 0} L^*(\lambda)$...approximated as

Approximate d* using L(\lambda^k,x^k+1)

[11] There are other algorithms such as cutting plane algorithm that also work for non-differentiable dual.

Idea of dual ascent

1. Initialize $\lambda^{(0)}$
2. Iteratively
   1. $\mathbf{x}^{k+1} = \operatorname{argmin}_{\mathbf{x}} L(\mathbf{x}, \lambda^k)$
   2. Gradient ascent for dual maximization problem: $d^* = \max_{\lambda \geq 0} L^*(\lambda)$...approximated as
      - ★ $d^* = \max_{\lambda \geq 0} L(\mathbf{x}^{k+1}, \lambda)$
      - ★ $\lambda^{k+1} = \lambda^k + t^k \partial_\lambda \left(f(\mathbf{x}^{k+1}) + \lambda^\top (A\mathbf{x}^{k+1} - b) \right)$ subgradient ascent

Update \lambda based on residual of the linear constraint

[11] There are other algorithms such as cutting plane algorithm that also work for non-differentiable dual.

Idea of dual ascent

1. Initialize $\lambda^{(0)}$
2. Iteratively
   1. $\mathbf{x}^{k+1} = \operatorname{argmin}_{\mathbf{x}} L(\mathbf{x}, \lambda^k)$
   2. Gradient ascent for dual maximization problem: $d^* = \max_{\lambda \geq 0} L^*(\lambda)$...approximated as
      - ⋆ $d^* = \max_{\lambda \geq 0} L(\mathbf{x}^{k+1}, \lambda)$
      - ⋆ $\lambda^{k+1} = \lambda^k + t^k \partial_\lambda \left(f(\mathbf{x}^{k+1}) + \lambda^\top (A\mathbf{x}^{k+1} - b) \right)$
        $= \lambda^k + t^k (A\mathbf{x}^{k+1} - b)$
      - ⋆ Leads to convergence (under assumptions of strong convexity etc) even if the Lagrange dual $L^*(\lambda)$ is non-differentiable[11].

on the f(x)

[11] There are other algorithms such as cutting plane algorithm that also work for non-differentiable dual.

- If λ converges to $\lambda^* = \underset{\lambda}{\operatorname{argmax}}\, L^*(\lambda)$ and strong duality holds, *i.e.*

$$\min_{\mathbf{x}} f(\mathbf{x}) = \max_{\lambda \geq 0} L^*(\lambda)$$

$$\text{s.t. } A\mathbf{x} = b$$

then,

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}}\, L(\mathbf{x}, \lambda^*)$$

- If f is **strongly convex with constant** m, and you ensure $t^k \leq m$, then convergence rate is $O\left(\frac{1}{k}\right)$.

Recap that for gradient descent with strong convexity m our convergence rate was (1-m/L)^k

This is certainly not good

Dual decomposition Special case of Dual Ascent

- $f(\mathbf{x})$ is decomposable into v blocks of variables (such as in Machine Learning, with decomposition over examples)

Dual decomposition

- $f(\mathbf{x})$ is decomposable into v blocks of variables (such as in Machine Learning, with decomposition over examples)

$$\min_{\mathbf{x}} f(\mathbf{x}) = \min_{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_v} \sum_{i=1}^{v} f_i(\mathbf{x}_i)$$

$$\text{s.t. } A\mathbf{x} = b$$

- Let $A = [A_1, A_2 \ldots A_i .. A_v]$ be a matrix of v blocks of columns of A corresponding to the blocks $\mathbf{x}_i$.

$$\underbrace{\begin{bmatrix} A_{11} & A_{i1} & A_{v1} \\ A_{12} & A_{i2} & A_{v2} \\ A_{1p} & A_{ip} & A_{vp} \end{bmatrix}}_{p \text{ Linear constraints}} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_i \\ \mathbf{x}_v \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{v} A_{i1}\mathbf{x}_i \\ \sum_{i=1}^{v} A_{i2}\mathbf{x}_i \\ \sum_{i=1}^{v} A_{ip}\mathbf{x}_i \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_p \end{bmatrix}$$

pth constraint

Dual decomposition (contd.)

- Thus: $f(\mathbf{x}) = \sum_{i=1}^{v} f_i(\mathbf{x}_i)$ and $\sum_{i=1}^{v} A_i\mathbf{x}_i = \mathbf{b}$
- Using this, simplify the first iterative step of dual ascent as $\mathbf{x}^{k+1} = \underset{\mathbf{x}}{\text{argmin}}\, f(\mathbf{x}) + \lambda^{k^\top}(A\mathbf{x} - b)$

solve v instances of this problem independently using
the shared \lambda^k from previous iteration

Dual decomposition (contd.)

- Thus: $f(\mathbf{x}) = \sum_{i=1}^{v} f_i(\mathbf{x}_i)$ and $\sum_{i=1}^{v} A_i\mathbf{x}_i = \mathbf{b}$
- Using this, simplify the first iterative step of dual ascent as

  $$\mathbf{x}^{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}}\, f(\mathbf{x}) + {\lambda^k}^\top (A\mathbf{x} - b)$$

  $$= \arg \min_{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_v} \sum_{i=1}^{v} f_i(\mathbf{x}_i) + {\lambda^k}^T \left(\left(\sum_{i=1}^{v} A_i\mathbf{x}_i \right) - \mathbf{b} \right)$$

- Thus, the following **SCATTER** step can be executed parallely for each block indexed by *i* after broadcasting λ^k from the previous iteration

Dual decomposition (contd.)

- Thus: $f(\mathbf{x}) = \sum_{i=1}^{v} f_i(\mathbf{x}_i)$ and $\sum_{i=1}^{v} A_i\mathbf{x}_i = \mathbf{b}$
- Using this, simplify the first iterative step of dual ascent as

$$\mathbf{x}^{k+1} = \operatorname*{argmin}_{\mathbf{x}} f(\mathbf{x}) + {\lambda^{k}}^{\top}(A\mathbf{x} - b)$$

$$= \arg \min_{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_v} \sum_{i=1}^{v} f_i(\mathbf{x}_i) + {\lambda^{k}}^{T}\left(\left(\sum_{i=1}^{v} A_i\mathbf{x}_i\right) - \mathbf{b}\right)$$

- Thus, the following **SCATTER** step can be executed parallely for each block indexed by i after broadcasting λ^k from the previous iteration

$$\mathbf{x}_i^{k+1} = \operatorname*{argmin}_{\mathbf{x}_i} f_i(\mathbf{x}_i) + {\lambda^{k}}^{\top}(A_i\mathbf{x}_i)$$

- Subsequently, GATHER all the xi's to compute next \lambda^k+1

Dual decomposition (contd.)

MAIN ADVANTAGE: Parallel execution of xi updates

- Thus: $f(\mathbf{x}) = \sum_{i=1}^{V} f_i(\mathbf{x}_i)$ and $\sum_{i=1}^{V} A_i\mathbf{x}_i = \mathbf{b}$
- Using this, simplify the first iterative step of dual ascent as

$$\mathbf{x}^{k+1} = \operatorname*{argmin}_{\mathbf{x}} f(\mathbf{x}) + {\lambda^{k}}^{\top}(A\mathbf{x} - b)$$

$$= \arg\min_{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_V} \sum_{i=1}^{V} f_i(\mathbf{x}_i) + {\lambda^{k}}^{T}\left(\left(\sum_{i=1}^{V} A_i\mathbf{x}_i\right) - \mathbf{b}\right)$$

- Thus, the following **SCATTER** step can be executed parallely for each block indexed by i after broadcasting λ^k from the previous iteration

Only a computational trick

$$\mathbf{x}_i^{k+1} = \operatorname*{argmin}_{\mathbf{x}_i} f_i(\mathbf{x}_i) + {\lambda^{k}}^{\top}(A_i\mathbf{x}_i)$$

- Subsequently, **GATHER** $\mathbf{x}_i^{k+1}$ from all nodes and update λ^{k+1} for again broadcasting

You first gather xis
and then update lambda

$$\lambda^{k+1} = \lambda^k + t^k(A\mathbf{x}^{k+1} - b)$$

Dual decomposition (contd.)

- If we have an inequality constraint instead of an equality, *e.g.* $A\mathbf{x} \leq b$

Dual decomposition (contd.)

- If we have an inequality constraint instead of an equality, *e.g.* $A\mathbf{x} \leq b$
  - Just project the computed λ^{k+1} to $\mathbf{R}_+^m$

$$\lambda^{k+1} \leftarrow \left(\lambda^{k+1}\right)_+$$

$$\textit{i.e.} \quad \lambda^{k+1} \leftarrow \mathit{max}\left(0, \lambda^{k+1}\right)$$

Making dual methods more robust: Augmented Lagrangian

- Dual ascent methods are too sensitive to $t^k \leq m$
- The idea is to bring in some **strong convexity** by transforming

$$\begin{aligned} &\min_{\mathbf{x}\in\mathbf{R}^n} f(\mathbf{x}) \\ &\text{s.t. } A\mathbf{x} = \mathbf{b} \end{aligned}$$

into

Making dual methods more robust: Augmented Lagrangian

- Dual ascent methods are too sensitive to $t^k \leq m$
- The idea is to bring in some **strong convexity** by transforming

$$\min_{\mathbf{x}\in\mathbf{R}^n} f(\mathbf{x})$$
$$\text{s.t. } A\mathbf{x} = \mathbf{b}$$

into

$$\min_{\mathbf{x}\in\mathbf{R}^n} f(\mathbf{x}) + \frac{\rho}{2}\|A\mathbf{x} - \mathbf{b}\|^2$$
$$\text{s.t. } A\mathbf{x} = \mathbf{b}$$

If A has full column rank, primal objective is strongly convex with constant $\rho\sigma^2_{min}(A)$

- In the initial iteration, $\lambda^{(0)}$ can be arbitrary and $\mathbf{x}^{(1)}$ need not satisfy $A\mathbf{x} = \mathbf{b}$
  Danger: $\mathbf{x}^{k+1}$ may very slowly start satisfying $A\mathbf{x} = \mathbf{b}$
- The transformed objective does not change the final solution, but improves the convergence of dual ascent methods

Augmented Lagrangian: Making dual methods more robust

- One of our main concerns with dual ascent is the sensitivity to $t^k \leq m$
  - If we take the augmented Lagrangian approach, we can use a default value of t^k **using the strong convexity factor that is proportional to ρ** (more motivation on next slide)
- Iterate
  1. $\mathbf{x}^{k+1} = \operatorname*{argmin}_{\mathbf{x}} f(\mathbf{x}) + {\lambda^k}^\top A\mathbf{x} + \frac{\rho}{2}\|A\mathbf{x} - \mathbf{b}\|^2$
     - ★ The last term here is kind of a barrier function. As we will see, in interior point or barrier methods applied to general inequality constraints, ρ will have to be reduced/changed at each step
  2. $\lambda^{k+1} = \lambda^k + \rho(A\mathbf{x}^{k+1} - \mathbf{b})$
     - ★ Due to ρ (related to strong convexity) instead of t^k , we get better convergence

Augmented Lagrangian: Making dual methods more robust (contd.)

More motivation for replacing t^k with ρ:

- Using ρ instead of t^k, we must have $0 \in \partial\left(f(\mathbf{x}^{k+1})\right) + A^T\left(\lambda^k + \rho(A\mathbf{x}^{k+1} - b)\right)$
- Considering $\widehat{\lambda}^{k+1} = \left(\lambda^k + \rho(A\mathbf{x}^{k+1} - b)\right)$, we get $0 \in \partial\left(f(\mathbf{x}^{k+1})\right) + A^T\widehat{\lambda}^{k+1}$ which is a necessary condition for our original problem
  - $\widehat{\lambda}^{k+1}$ in place of λ^*

Augmented Lagrangian: Making dual methods more robust (contd.)

More motivation for replacing t^k with ρ:

- Using ρ instead of t^k, we must have
  $0 \in \partial\left(f(\mathbf{x}^{k+1})\right) + A^T\left(\lambda^k + \rho(A\mathbf{x}^{k+1} - b)\right)$
- Considering $\widehat{\lambda}^{k+1} = \left(\lambda^k + \rho(A\mathbf{x}^{k+1} - b)\right)$, we get
  $0 \in \partial\left(f(\mathbf{x}^{k+1})\right) + A^T\widehat{\lambda}^{k+1}$
  which is a necessary condition for our original problem
  - $\widehat{\lambda}^{k+1}$ in place of λ^*
- What is the challenge in Applying Dual Decomposition to this Augmented Lagrangian?

  The dual decomposition trick is not applicable rightaway owing to the presence of the augmented Lagrangian

ADMM: Best of Several Worlds

- **Extend the decomposition idea to augmented Lagrangian.**
- Iteratively solve a smaller problem with respect to x_i by fixing variables x_j for $j \neq i$.
- Consider simpler case $N = 2$ (easily generalizable to N). $f(x) = f_1(x_1) + f_2(x_2)$ and augmented Lagrangian is

$$L_\rho(x_1, x_2, \lambda) = f_1(x_1) + f_2(x_2) + \lambda^T(A_1x_1 + A_2x_2 - b) + \underline{\frac{\rho}{2}\|A_1x_1 + A_2x_2 - \mathbf{b}\|_2^2}. \quad (80)$$

Does not decompose

ADMM solves each direction alternatively

Solve for x1 keeping x2 from previous iteration

$$x_1^{t+1} = \arg\min_{x_1} L_\rho(x_1, \underline{x_2^t,} \lambda^t) \quad (81)$$

$$x_2^{t+1} = \arg\min_{x_2} L_\rho(x_1^{t+1}, x_2, \lambda^t) \quad (82)$$

$$\lambda^{t+1} = \lambda^t + \rho(A_1x_1^{t+1} + A_2x_2^{t+1} - \mathbf{b}) \quad (83)$$

- Main difference wrt dual decomposition ascent:

Earlier you did not have to use values from previous iterate for x_2...x_v while finding next iterate for x_1... Here you have to do so.

ADMM: Best of Several Worlds

- **Extend the decomposition idea to augmented Lagrangian.**
- Iteratively solve a smaller problem with respect to x_i by fixing variables x_j for $j \neq i$.
- Consider simpler case $N = 2$ (easily generalizable to N). $f(x) = f_1(x_1) + f_2(x_2)$ and augmented Lagrangian is

$$L_\rho(x_1, x_2, \lambda) = f_1(x_1) + f_2(x_2) + \lambda^T(A_1x_1 + A_2x_2 - b) + \frac{\rho}{2}\|A_1x_1 + A_2x_2 - \mathbf{b}\|_2^2. \tag{80}$$

ADMM solves each direction alternatively

$$x_1^{t+1} = \arg\min_{x_1} L_\rho(x_1, x_2^t, \lambda^t) \tag{81}$$

$$x_2^{t+1} = \arg\min_{x_2} L_\rho(x_1^{t+1}, x_2, \lambda^t) \tag{82}$$

$$\lambda^{t+1} = \lambda^t + \rho(A_1x_1^{t+1} + A_2x_2^{t+1} - \mathbf{b}) \tag{83}$$

- Main difference wrt dual decomposition ascent: ADMM updates x_i sequentially. Additional augmented term does not let us decompose the Lagrangian form into N components conditionally independent wrt λ

ADMM: Alternating Direction Method of Multipliers

Augmented lagrangian ==> Method of multipliers

1. Assume that functions f_1, f_2 are closed, proper, and convex (that is, they have closed, nonempty, and convex epigraphs)
2. Assume that the un-augmented Lagrangian $L_0(x_1, x_2, \lambda)$ has (critical) saddle points $\widehat{x}_1, \widehat{x}_2$ and $\widehat{\lambda}$ subject to
$$L_0(\widehat{x}_1, \widehat{x}_2, \lambda) \leq L_0(\widehat{x}_1, \widehat{x}_2, \widehat{\lambda}) \leq L_0(x_1, x_2, \widehat{\lambda}) \tag{84}$$
3. No need to assume that A_1, A_2 *etc.* have full column rank

Then when $t \to \infty$, one can prove that[12]

Residual convergence: $r^t = A_1 x_1^t + A_2 x_2^t - \mathbf{b} \to 0$

Objective convergence: $f_1(x_1^t) + f_2(x_2^t) \to f^*$

Dual variable convergence: $\lambda^t \to \lambda^*$

[12] `https://web.stanford.edu/~boyd/papers/pdf/admm_distr_stats.pdf`

(Log) Barrier methods

Barrier Methods for Constrained Optimization

Consider a more general constrained optimization problem

$$\min_{\mathbf{x}\in\mathbf{R}^n} f(\mathbf{x})$$
$$\text{s.t.} g_i(\mathbf{x}) \leq 0 \; i = 1 \ldots m$$
$$\text{and } A\mathbf{x} = \mathbf{b}$$

Possibly reformulations of this problem include:

$$\min_x f(x) + \lambda B(x)$$

where B is a **barrier function** like

1. $B(x) = \max_i \min_{u \in \{g_i(\mathbf{x}) \leq 0\}} \|\mathbf{x} - u\|^2$
2. $B(x) = \sum I_{g_i}(\mathbf{x})$ (Recap: Projected Gradient Descent was built on this and a linear approximation to $f(\mathbf{x})$)
3. $B(x) = \phi_{g_i}(\mathbf{x}) = -\frac{1}{t}\log\left(-g_i(\mathbf{x})\right)$

(We saw quadratic barrier in Augmented Lagrangian method)

Barrier Method: Example

As a very simple example, consider the following inequality constrained optimization problem.

$$\begin{array}{ll} \text{minimize} & x^2 \\ \text{subject to} & x \geq 1 \end{array}$$

Optimal solution is 1

The logarithmic barrier formulation of this problem is

$$\text{minimize} \quad x^2 - \mu \ln (x - 1)$$

The unconstrained minimizer for this convex logarithmic barrier function is $\widehat{\mathbf{x}}(\mu) = \frac{1}{2} + \frac{1}{2}\sqrt{1 + 2\mu}$. As $\mu \to 0$, the optimal point of the logarithmic barrier problem approaches the actual point of optimality $\widehat{\mathbf{x}} = 1$ (which, as we can see, lies on the boundary of the feasible region). The generalized idea, that as $\mu \to 0$, $f(\widehat{\mathbf{x}}) \to p^*$ (where p^* is the optimal for primal) will be proved next.

Barrier Method and Linear Program

For details of either approach to solve LP look at Section 4.7 of https://www.cse.iitb.ac.in/~cs709/notes/BasicsOfConvexOptimization.pdf Especially look at its two subsections 4.7.1 and 4.7.2

Recap:

Problem type	**Objective Function**	**Constraints**	$L^*(\lambda)$	**Dual constraints**	**Strong duality**
Linear Program	$\mathbf{c}^T\mathbf{x}$	$A\mathbf{x} \leq \mathbf{b}$	$-\mathbf{b}^T\lambda$	$A^T\lambda + \mathbf{c} = \mathbf{0}$	Feasible primal

What are necessary conditions at primal-dual optimality?

- .. You have feasiblity conditions
- .. And Complementary slackness conditions

Simplex tries to enforce feasibility while maintaining complementary slackness [Section 4.7.1 of BasicsOfConvexOptimization.pdf]

Barrier methods (Interior point) for LP tries to enforce this eventually, while maintaining feasibility [See Section 4.7.2 of notes/BasicsOfConvexOptimization.pdf]

Log Barrier (Interior Point) Method

- The log barrier function is defined as

$$B(x) = \phi_{g_i}(\mathbf{x}) = -\frac{1}{t}\log\left(-g_i(\mathbf{x})\right)$$

- It looks like an approximation of $\sum I_{g_i}(x)$
- $f(\mathbf{x}) + \sum_i \phi_{g_i}(\mathbf{x})$
  is convex if f and g_i are convex
- Let λ_i be lagrange multiplier associated with inequality constraint $g_i(\mathbf{x}) \leq 0$
- We've taken care of the inequality constraints, lets also consider an equality constraint $A\mathbf{x} = \mathbf{b}$ with corresponding langrage multipler (vector) ν

We have already seen several algorithms for handling linear constraints. These include Project Gradient Descent, Dual Ascent and ADMM. For another approach that can use Gauss Elimination to find the family of solutions to Ax = b see Section 4.6.1 of https://www.cse.iitb.ac.in/~cs709/notes/BasicsOfConvexOptimization.pdf

Log Barrier Method (contd.)

- Our objective becomes

$$\min_x f(x) + \sum_i \left(-\frac{1}{t}\right) \log\left(-g_i(x)\right)$$

$$\text{s.t. } Ax = b$$

- At different values of t, we get different $x^*(t)$
- Let $\underline{\lambda_i^*(t)} =$ identify nature of lambda that aligns with the barrier
- First-order necessary conditions for optimality (and strong duality)[13] at $x^*(t), \lambda_i^*(t)$:

[13] of original problem