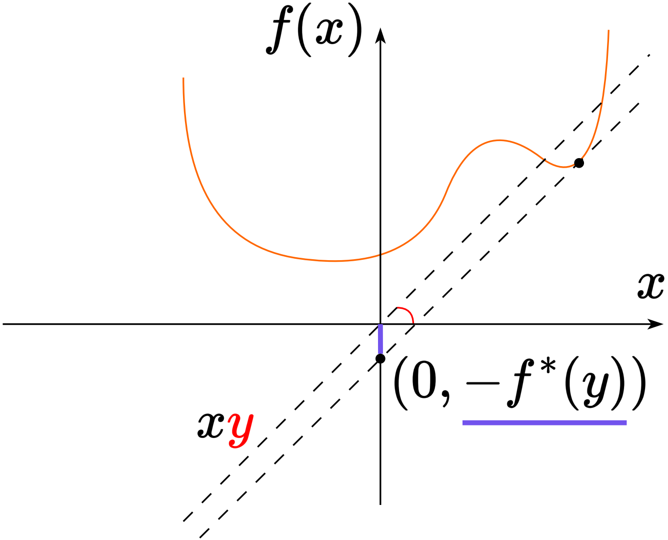
Introduction to dual methods

Daniil Merkulov

Optimization methods. MIPT

Definition

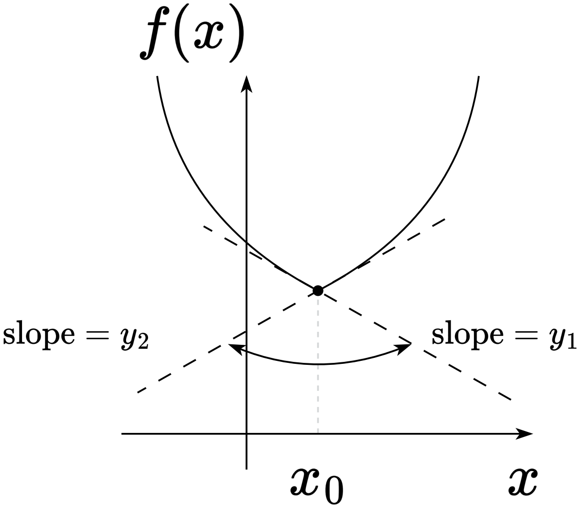


Recall that given $f : \mathbb{R}^n \to \mathbb{R}$, the function defined by

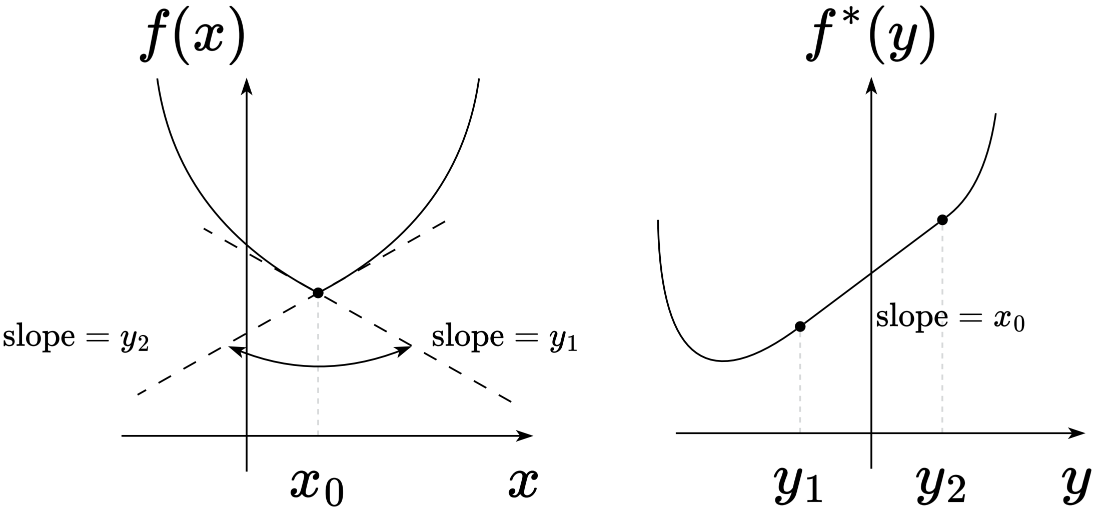
$$f^*(y) = \max_x \left[y^T x - f(x) \right]$$

is called its conjugate.

Geometrical intution



Geometrical intution



Conjugate function properties

Recall that given $f : \mathbb{R}^n \to \mathbb{R}$, the function defined by

$$f^*(y) = \max_x \left[y^T x - f(x)\right]$$

is called its conjugate.

- Conjugates appear frequently in dual programs, since

$$-f^*(y) = \min_x \left[f(x) - y^T x\right]$$

Conjugate function properties

Recall that given $f : \mathbb{R}^n \to \mathbb{R}$, the function defined by

$$f^*(y) = \max_x \left[y^T x - f(x) \right]$$

is called its conjugate.

- Conjugates appear frequently in dual programs, since

$$-f^*(y) = \min_x \left[f(x) - y^T x \right]$$

- If f is closed and convex, then $f^{**} = f$. Also,

$$x \in \partial f^*(y) \Leftrightarrow y \in \partial f(x) \Leftrightarrow x \in \arg\min_z \left[f(z) - y^T z \right]$$

Conjugate function properties

Recall that given $f : \mathbb{R}^n \to \mathbb{R}$, the function defined by

$$f^*(y) = \max_x \left[y^T x - f(x) \right]$$

is called its conjugate.

- Conjugates appear frequently in dual programs, since

$$-f^*(y) = \min_x \left[f(x) - y^T x \right]$$

- If f is closed and convex, then $f^{**} = f$. Also,

$$x \in \partial f^*(y) \Leftrightarrow y \in \partial f(x) \Leftrightarrow x \in \arg\min_z \left[f(z) - y^T z \right]$$

- If f is strictly convex, then

$$\nabla f^*(y) = \arg\min_z \left[f(z) - y^T z \right]$$

Conjugate function properties (proofs)

We will show that $x \in \partial f^*(y) \Leftrightarrow y \in \partial f(x)$, assuming that f is convex and closed.

- **Proof of** $\Leftarrow$: Suppose $y \in \partial f(x)$. Then $x \in M_y$, the set of maximizers of $y^T z - f(z)$ over z. But

$$f^*(y) = \max_z \{y^T z - f(z)\} \quad \text{and} \quad \partial f^*(y) = \text{cl}(\text{conv}(\bigcup_{z \in M_y} \{z\})).$$

  Thus $x \in \partial f^*(y)$.

Conjugate function properties (proofs)

We will show that $x \in \partial f^*(y) \Leftrightarrow y \in \partial f(x)$, assuming that f is convex and closed.

- **Proof of** $\Leftarrow$: Suppose $y \in \partial f(x)$. Then $x \in M_y$, the set of maximizers of $y^T z - f(z)$ over z. But

$$
f^*(y) = \max_z \{y^T z - f(z)\} \quad \text{and} \quad \partial f^*(y) = \mathrm{cl}(\mathrm{conv}(\bigcup_{z \in M_y} \{z\})).
$$

  Thus $x \in \partial f^*(y)$.

- **Proof of** $\Rightarrow$: From what we showed above, if $x \in \partial f^*(y)$, then $y \in \partial f^*(x)$, but $f^{**} = f$.

Conjugate function properties (proofs)

We will show that $x \in \partial f^*(y) \Leftrightarrow y \in \partial f(x)$, assuming that f is convex and closed.

- **Proof of** $\Leftarrow$: Suppose $y \in \partial f(x)$. Then $x \in M_y$, the set of maximizers of $y^T z - f(z)$ over z. But

$$f^*(y) = \max_z \{y^T z - f(z)\} \quad \text{and} \quad \partial f^*(y) = \mathrm{cl}(\mathrm{conv}(\bigcup_{z \in M_y} \{z\})).$$

  Thus $x \in \partial f^*(y)$.

- **Proof of** $\Rightarrow$: From what we showed above, if $x \in \partial f^*(y)$, then $y \in \partial f^*(x)$, but $f^{**} = f$.

Conjugate function properties (proofs)

We will show that $x \in \partial f^*(y) \Leftrightarrow y \in \partial f(x)$, assuming that f is convex and closed.

- **Proof of** $\Leftarrow$: Suppose $y \in \partial f(x)$. Then $x \in M_y$, the set of maximizers of $y^T z - f(z)$ over z. But

$$f^*(y) = \max_z \{y^T z - f(z)\} \quad \text{and} \quad \partial f^*(y) = \text{cl}(\text{conv}(\bigcup_{z \in M_y} \{z\})).$$

  Thus $x \in \partial f^*(y)$.

- **Proof of** $\Rightarrow$: From what we showed above, if $x \in \partial f^*(y)$, then $y \in \partial f^*(x)$, but $f^{**} = f$.

Clearly $y \in \partial f(x) \Leftrightarrow x \in \arg\min_z \{f(z) - y^T z\}$

Lastly, if f is strictly convex, then we know that $f(z) - y^T z$ has a unique minimizer over z, and this must be $\nabla f^*(y)$.

Dual (sub)gradient method

Even if we can't derive dual (conjugate) in closed form, we can still use dual-based gradient or subgradient methods.

Consider the problem:

$$\min_x \quad f(x) \quad \text{subject to} \quad Ax = b$$

Dual (sub)gradient method

Even if we can't derive dual (conjugate) in closed form, we can still use dual-based gradient or subgradient methods.

Consider the problem:

$$\min_x \quad f(x) \quad \text{subject to} \quad Ax = b$$

Its dual problem is:

$$\max_u \quad -f^*(-A^T u) - b^T u$$

where f^* is the conjugate of f. Defining $g(u) = -f^*(-A^T u) - b^T u$, note that:

$$\partial g(u) = A \partial f^*(-A^T u) - b$$

Dual (sub)gradient method

Even if we can't derive dual (conjugate) in closed form, we can still use dual-based gradient or subgradient methods.

Consider the problem:

$$\min_x \quad f(x) \quad \text{subject to} \quad Ax = b$$

Its dual problem is:

$$\max_u \quad -f^*(-A^T u) - b^T u$$

where f^* is the conjugate of f. Defining $g(u) = -f^*(-A^T u) - b^T u$, note that:

$$\partial g(u) = A\partial f^*(-A^T u) - b$$

Therefore, using what we know about conjugates

$$\partial g(u) = Ax - b \quad \text{where} \quad x \in \arg\min_z \left[f(z) + u^T Az\right]$$

Dual (sub)gradient method

Even if we can't derive dual (conjugate) in closed form, we can still use dual-based gradient or subgradient methods.

Consider the problem:

$$\min_x \quad f(x) \quad \text{subject to} \quad Ax = b$$

Its dual problem is:

$$\max_u \quad -f^*(-A^T u) - b^T u$$

where f^* is the conjugate of f. Defining $g(u) = -f^*(-A^T u) - b^T u$, note that:

$$\partial g(u) = A \partial f^*(-A^T u) - b$$

Therefore, using what we know about conjugates

$$\partial g(u) = Ax - b \quad \text{where} \quad x \in \arg\min_z \left[f(z) + u^T A z \right]$$

Dual ascent method for maximizing dual objective:

$$x_k \in \arg\min_x \left[f(x) + (u_{k-1})^T A x \right]$$
$$u_k = u_{k-1} + \alpha_k (A x_k - b)$$

- Step sizes α_k, $k = 1, 2, 3, \ldots$, are chosen in standard ways.

Dual (sub)gradient method

Even if we can't derive dual (conjugate) in closed form, we can still use dual-based gradient or subgradient methods.

Consider the problem:

$$\min_x \quad f(x) \quad \text{subject to} \quad Ax = b$$

Its dual problem is:

$$\max_u \quad -f^*(-A^T u) - b^T u$$

where f^* is the conjugate of f. Defining $g(u) = -f^*(-A^T u) - b^T u$, note that:

$$\partial g(u) = A\partial f^*(-A^T u) - b$$

Therefore, using what we know about conjugates

$$\partial g(u) = Ax - b \quad \text{where} \quad x \in \arg\min_z \left[f(z) + u^T Az\right]$$

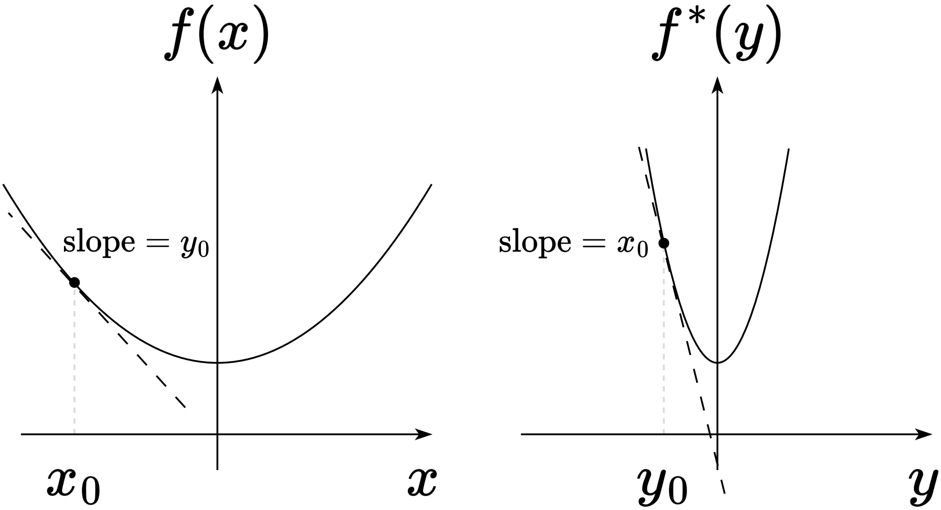
Dual ascent method for maximizing dual objective:

$$x_k \in \arg\min_x \left[f(x) + (u_{k-1})^T Ax\right]$$

$$u_k = u_{k-1} + \alpha_k(Ax_k - b)$$

- Step sizes α_k, $k = 1, 2, 3, \ldots$, are chosen in standard ways.
- Proximal gradients and acceleration can be applied as they would usually.

Slopes of f and f^*



Slopes of f and f^*

Assume that f is a closed and convex function. Then f is strongly convex with parameter $\mu \Leftrightarrow \nabla f^*$ is Lipschitz with parameter $1/\mu$.

Slopes of f and f^*

Assume that f is a closed and convex function. Then f is strongly convex with parameter $\mu \Leftrightarrow \nabla f^*$ is Lipschitz with parameter $1/\mu$.

Proof of "$\Rightarrow$": Recall, if g is strongly convex with minimizer x, then

$$g(y) \geq g(x) + \frac{\mu}{2}\|y - x\|^2, \quad \text{for all } y$$

Slopes of f and f^*

Assume that f is a closed and convex function. Then f is strongly convex with parameter $\mu \Leftrightarrow \nabla f^*$ is Lipschitz with parameter $1/\mu$.

Proof of "$\Rightarrow$": Recall, if g is strongly convex with minimizer x, then

$$g(y) \geq g(x) + \frac{\mu}{2}\|y - x\|^2, \quad \text{for all } y$$

Hence, defining $x_u = \nabla f^*(u)$ and $x_v = \nabla f^*(v)$,

$$f(x_v) - u^T x_v \geq f(x_u) - u^T x_u + \frac{\mu}{2}\|x_u - x_v\|^2$$
$$f(x_u) - v^T x_u \geq f(x_v) - v^T x_v + \frac{\mu}{2}\|x_u - x_v\|^2$$

Slopes of f and f^*

Assume that f is a closed and convex function. Then f is strongly convex with parameter $\mu \Leftrightarrow \nabla f^*$ is Lipschitz with parameter $1/\mu$.

Proof of "$\Rightarrow$": Recall, if g is strongly convex with minimizer x, then

$$g(y) \geq g(x) + \frac{\mu}{2}\|y - x\|^2, \quad \text{for all } y$$

Hence, defining $x_u = \nabla f^*(u)$ and $x_v = \nabla f^*(v)$,

$$f(x_v) - u^T x_v \geq f(x_u) - u^T x_u + \frac{\mu}{2}\|x_u - x_v\|^2$$
$$f(x_u) - v^T x_u \geq f(x_v) - v^T x_v + \frac{\mu}{2}\|x_u - x_v\|^2$$

Adding these together, using the Cauchy-Schwarz inequality, and rearranging shows that

$$\|x_u - x_v\|^2 \leq \frac{1}{\mu}\|u - v\|^2$$

Slopes of f and f^*

Proof of "$\Leftarrow$": for simplicity, call $g = f^*$ and $L = \frac{1}{\mu}$. As ∇g is Lipschitz with constant L, so is $g_x(z) = g(z) - \nabla g(x)^T z$, hence

$$g_x(z) \leq g_x(y) + \nabla g_x(y)^T (z - y) + \frac{L}{2}\|z - y\|_2^2$$

Slopes of f and f^*

Proof of "$\Leftarrow$": for simplicity, call $g = f^*$ and $L = \frac{1}{\mu}$. As ∇g is Lipschitz with constant L, so is $g_x(z) = g(z) - \nabla g(x)^T z$, hence

$$g_x(z) \leq g_x(y) + \nabla g_x(y)^T (z - y) + \frac{L}{2}\|z - y\|_2^2$$

Minimizing each side over z, and rearranging, gives

$$\frac{1}{2L}\|\nabla g(x) - \nabla g(y)\|^2 \leq g(y) - g(x) + \nabla g(x)^T (x - y)$$

Slopes of f and f^*

Proof of "$\Leftarrow$": for simplicity, call $g = f^*$ and $L = \frac{1}{\mu}$. As ∇g is Lipschitz with constant L, so is $g_x(z) = g(z) - \nabla g(x)^T z$, hence

$$g_x(z) \leq g_x(y) + \nabla g_x(y)^T(z-y) + \frac{L}{2}\|z-y\|_2^2$$

Minimizing each side over z, and rearranging, gives

$$\frac{1}{2L}\|\nabla g(x) - \nabla g(y)\|^2 \leq g(y) - g(x) + \nabla g(x)^T(x-y)$$

Exchanging roles of x, y, and adding together, gives

$$\frac{1}{L}\|\nabla g(x) - \nabla g(y)\|^2 \leq (\nabla g(x) - \nabla g(y))^T(x-y)$$

Slopes of f and f^*

Proof of "$\Leftarrow$": for simplicity, call $g = f^*$ and $L = \frac{1}{\mu}$. As ∇g is Lipschitz with constant L, so is $g_x(z) = g(z) - \nabla g(x)^T z$, hence

$$g_x(z) \leq g_x(y) + \nabla g_x(y)^T(z-y) + \frac{L}{2}\|z-y\|_2^2$$

Minimizing each side over z, and rearranging, gives

$$\frac{1}{2L}\|\nabla g(x) - \nabla g(y)\|^2 \leq g(y) - g(x) + \nabla g(x)^T(x-y)$$

Exchanging roles of x, y, and adding together, gives

$$\frac{1}{L}\|\nabla g(x) - \nabla g(y)\|^2 \leq (\nabla g(x) - \nabla g(y))^T(x-y)$$

Let $u = \nabla f(x)$, $v = \nabla g(y)$; then $x \in \partial g^*(u)$, $y \in \partial g^*(v)$, and the above reads $(x-y)^T(u-v) \geq \frac{\|u-v\|^2}{L}$, implying the result.

Convergence guarantees

The following results hold from combining the last fact with what we already know about gradient descent:[1]

- If f is strongly convex with parameter μ, then dual gradient ascent with constant step sizes $\alpha_k = \mu$ converges at sublinear rate $O(\frac{1}{\epsilon})$.

[1]This is ignoring the role of A, and thus reflects the case when the singular values of A are all close to 1. To be more precise, the step sizes here should be: $\frac{\mu}{\sigma_{\max}(A)^2}$ (first case) and $\frac{2}{\frac{\sigma_{\max}(A)^2}{\mu} + \frac{\sigma_{\min}(A)^2}{L}}$ (second case).

Convergence guarantees

The following results hold from combining the last fact with what we already know about gradient descent:[1]

- If f is strongly convex with parameter μ, then dual gradient ascent with constant step sizes $\alpha_k = \mu$ converges at sublinear rate $O(\frac{1}{\epsilon})$.
- If f is strongly convex with parameter μ and ∇f is Lipschitz with parameter L, then dual gradient ascent with step sizes $\alpha_k = \frac{2}{\frac{1}{\mu}+\frac{1}{L}}$ converges at linear rate $O(\log(\frac{1}{\epsilon}))$.

[1]This is ignoring the role of A, and thus reflects the case when the singular values of A are all close to 1. To be more precise, the step sizes here should be: $\frac{\mu}{\sigma_{\text{max}}(A)^2}$ (first case) and $\frac{2}{\frac{\sigma_{\text{max}}(A)^2}{\mu}+\frac{\sigma_{\text{min}}(A)^2}{L}}$ (second case).

Convergence guarantees

The following results hold from combining the last fact with what we already know about gradient descent:[1]

- If f is strongly convex with parameter μ, then dual gradient ascent with constant step sizes $\alpha_k = \mu$ converges at sublinear rate $O(\frac{1}{\epsilon})$.
- If f is strongly convex with parameter μ and ∇f is Lipschitz with parameter L, then dual gradient ascent with step sizes $\alpha_k = \frac{2}{\frac{1}{\mu} + \frac{1}{L}}$ converges at linear rate $O(\log(\frac{1}{\epsilon}))$.

[1] This is ignoring the role of A, and thus reflects the case when the singular values of A are all close to 1. To be more precise, the step sizes here should be: $\frac{\mu}{\sigma_{\text{max}}(A)^2}$ (first case) and $\frac{2}{\frac{\sigma_{\text{max}}(A)^2}{\mu} + \frac{\sigma_{\text{min}}(A)^2}{L}}$ (second case).

Convergence guarantees

The following results hold from combining the last fact with what we already know about gradient descent:[1]

- If f is strongly convex with parameter μ, then dual gradient ascent with constant step sizes $\alpha_k = \mu$ converges at sublinear rate $O(\frac{1}{\epsilon})$.
- If f is strongly convex with parameter μ and ∇f is Lipschitz with parameter L, then dual gradient ascent with step sizes $\alpha_k = \frac{2}{\frac{1}{\mu}+\frac{1}{L}}$ converges at linear rate $O(\log(\frac{1}{\epsilon}))$.

Note that this describes convergence in the dual. (Convergence in the primal requires more assumptions)

[1]This is ignoring the role of A, and thus reflects the case when the singular values of A are all close to 1. To be more precise, the step sizes here should be: $\frac{\mu}{\sigma_{\text{max}}(A)^2}$ (first case) and $\frac{2}{\frac{\sigma_{\text{max}}(A)^2}{\mu}+\frac{\sigma_{\text{min}}(A)^2}{L}}$ (second case).

Dual decomposition

Consider

$$\min_x \sum_{i=1}^{B} f_i(x_i) \quad \text{subject to} \quad Ax = b$$

Dual decomposition

Consider

$$\min_x \sum_{i=1}^{B} f_i(x_i) \quad \text{subject to} \quad Ax = b$$

Here $x = (x_1, \dots, x_B) \in \mathbb{R}^n$ divides into B blocks of variables, with each $x_i \in \mathbb{R}^{n_i}$. We can also partition A accordingly:

$$A = [A_1 \dots A_B], \text{ where } A_i \in \mathbb{R}^{m \times n_i}$$

Dual decomposition

Consider

$$\min_x \sum_{i=1}^B f_i(x_i) \quad \text{subject to} \quad Ax = b$$

Here $x = (x_1, \dots, x_B) \in \mathbb{R}^n$ divides into B blocks of variables, with each $x_i \in \mathbb{R}^{n_i}$. We can also partition A accordingly:

$$A = [A_1 \dots A_B], \text{ where } A_i \in \mathbb{R}^{m \times n_i}$$

Simple but powerful observation, in calculation of subgradient, is that the minimization decomposes into B separate problems:

$$x^{\text{new}} \in \arg\min_x \left(\sum_{i=1}^B f_i(x_i) + u^T Ax \right)$$

$$\Rightarrow x_i^{\text{new}} \in \arg\min_{x_i} \left(f_i(x_i) + u^T A_i x_i \right), \quad i = 1, \dots, B$$

$$x_i^k \in \arg\min_{x_i} \left(f_i(x_i) + (u^{k-1})^T A_i x_i \right), \quad i = 1, \dots, B$$

$$u^k = u^{k-1} + \alpha_k \left(\sum_{i=1}^B A_i x_i^k - b \right)$$

Dual decomposition

Consider

$$\min_x \sum_{i=1}^B f_i(x_i) \quad \text{subject to} \quad Ax = b$$

Here $x = (x_1, \ldots, x_B) \in \mathbb{R}^n$ divides into B blocks of variables, with each $x_i \in \mathbb{R}^{n_i}$. We can also partition A accordingly:

$$A = [A_1 \ldots A_B], \text{ where } A_i \in \mathbb{R}^{m \times n_i}$$

Simple but powerful observation, in calculation of subgradient, is that the minimization decomposes into B separate problems:

$$x^{\text{new}} \in \arg\min_x \left(\sum_{i=1}^B f_i(x_i) + u^T A x \right)$$

$$\Rightarrow x_i^{\text{new}} \in \arg\min_{x_i} \left(f_i(x_i) + u^T A_i x_i \right), \quad i = 1, \ldots, B$$

$$x_i^k \in \arg\min_{x_i} \left(f_i(x_i) + (u^{k-1})^T A_i x_i \right), \quad i = 1, \ldots, B$$

$$u^k = u^{k-1} + \alpha_k \left(\sum_{i=1}^B A_i x_i^k - b \right)$$

Can think of these steps as:

- **Broadcast:** Send u to each of the B processors, each optimizes in parallel to find x_i.

Dual decomposition

Consider

$$\min_x \sum_{i=1}^B f_i(x_i) \quad \text{subject to} \quad Ax = b$$

Here $x = (x_1, \dots, x_B) \in \mathbb{R}^n$ divides into B blocks of variables, with each $x_i \in \mathbb{R}^{n_i}$. We can also partition A accordingly:

$$A = [A_1 \dots A_B], \text{ where } A_i \in \mathbb{R}^{m \times n_i}$$

Simple but powerful observation, in calculation of subgradient, is that the minimization decomposes into B separate problems:

$$x^{\text{new}} \in \arg\min_x \left(\sum_{i=1}^B f_i(x_i) + u^T Ax \right)$$

$$\Rightarrow x_i^{\text{new}} \in \arg\min_{x_i} \left(f_i(x_i) + u^T A_i x_i \right), \quad i = 1, \dots, B$$

$$x_i^k \in \arg\min_{x_i} \left(f_i(x_i) + (u^{k-1})^T A_i x_i \right), \quad i = 1, \dots, B$$

$$u^k = u^{k-1} + \alpha_k \left(\sum_{i=1}^B A_i x_i^k - b \right)$$

Can think of these steps as:

- **Broadcast:** Send u to each of the B processors, each optimizes in parallel to find x_i.
- **Gather:** Collect $A_i x_i$ from each processor, update the global dual variable u.

Inequality constraints

Consider the optimization problem:

$$\min_x \sum_{i=1}^{B} f_i(x_i) \quad \text{subject to} \quad \sum_{i=1}^{B} A_i x_i \leq b$$

Inequality constraints

Consider the optimization problem:

$$\min_x \sum_{i=1}^B f_i(x_i) \quad \text{subject to} \quad \sum_{i=1}^B A_i x_i \leq b$$

Using **dual decomposition**, specifically the **projected subgradient method**, the iterative steps can be expressed as:

- The primal update step:

$$x_i^k \in \arg\min_{x_i} \left[f_i(x_i) + \left(u^{k-1}\right)^T A_i x_i \right], \quad i = 1, \ldots, B$$

Inequality constraints

Consider the optimization problem:

$$
\min_x \sum_{i=1}^B f_i(x_i) \quad \text{subject to} \quad \sum_{i=1}^B A_i x_i \leq b
$$

Using **dual decomposition**, specifically the **projected subgradient method**, the iterative steps can be expressed as:

- The primal update step:

$$
x_i^k \in \arg\min_{x_i} \left[f_i(x_i) + \left(u^{k-1}\right)^T A_i x_i \right], \quad i = 1, \ldots, B
$$

- The dual update step:

$$
u^k = \left(u^{k-1} + \alpha_k \left(\sum_{i=1}^B A_i x_i^k - b \right) \right)_+
$$

where $(u)_+$ denotes the positive part of u, i.e., $(u_+)_i = \max\{0, u_i\}$, for $i = 1, \ldots, m$.

Price Coordination Interpretation (Vandenberghe)

- **System Overview**: Consider a system with B units, where each unit independently chooses its decision variable x_i, which determines how to allocate its goods.

Price Coordination Interpretation (Vandenberghe)

- **System Overview**: Consider a system with B units, where each unit independently chooses its decision variable x_i, which determines how to allocate its goods.
- **Resource Constraints**: These are limits on shared resources, represented by the rows of A. Each component of the dual variable u_j represents the price of resource j.

Price Coordination Interpretation (Vandenberghe)

- **System Overview**: Consider a system with B units, where each unit independently chooses its decision variable x_i, which determines how to allocate its goods.
- **Resource Constraints**: These are limits on shared resources, represented by the rows of A. Each component of the dual variable u_j represents the price of resource j.
- **Dual Update Rule**:

$$u_j^{\text{new}} = (u_j - ts_j)_+, \quad j = 1, \dots, m$$

where $s = b - \sum_{i=1}^{B} A_i x_i$ represents the slacks.

Price Coordination Interpretation (Vandenberghe)

- **System Overview**: Consider a system with B units, where each unit independently chooses its decision variable x_i, which determines how to allocate its goods.
- **Resource Constraints**: These are limits on shared resources, represented by the rows of A. Each component of the dual variable u_j represents the price of resource j.
- **Dual Update Rule**:

$$u_j^{\text{new}} = (u_j - ts_j)_+, \quad j = 1, \dots, m$$

  where $s = b - \sum_{i=1}^{B} A_i x_i$ represents the slacks.
- **Price Adjustments**:

Price Coordination Interpretation (Vandenberghe)

- **System Overview**: Consider a system with B units, where each unit independently chooses its decision variable x_i, which determines how to allocate its goods.
- **Resource Constraints**: These are limits on shared resources, represented by the rows of A. Each component of the dual variable u_j represents the price of resource j.
- **Dual Update Rule**:

$$u_j^{\text{new}} = (u_j - ts_j)_+, \quad j = 1, \dots, m$$

  where $s = b - \sum_{i=1}^{B} A_i x_i$ represents the slacks.
- **Price Adjustments**:
  - **Increase price** u_j if resource j is over-utilized ($s_j < 0$).

Price Coordination Interpretation (Vandenberghe)

- **System Overview**: Consider a system with B units, where each unit independently chooses its decision variable x_i, which determines how to allocate its goods.
- **Resource Constraints**: These are limits on shared resources, represented by the rows of A. Each component of the dual variable u_j represents the price of resource j.
- **Dual Update Rule**:

$$u_j^{\text{new}} = (u_j - ts_j)_+, \quad j = 1, \ldots, m$$

  where $s = b - \sum_{i=1}^{B} A_i x_i$ represents the slacks.
- **Price Adjustments**:
  - **Increase price** u_j if resource j is over-utilized ($s_j < 0$).
  - **Decrease price** u_j if resource j is under-utilized ($s_j > 0$).

Price Coordination Interpretation (Vandenberghe)

- **System Overview**: Consider a system with B units, where each unit independently chooses its decision variable x_i, which determines how to allocate its goods.
- **Resource Constraints**: These are limits on shared resources, represented by the rows of A. Each component of the dual variable u_j represents the price of resource j.
- **Dual Update Rule**:

$$u_j^{\text{new}} = (u_j - ts_j)_+, \quad j = 1, \ldots, m$$

  where $s = b - \sum_{i=1}^{B} A_i x_i$ represents the slacks.
- **Price Adjustments**:
  - **Increase price** u_j if resource j is over-utilized ($s_j < 0$).
  - **Decrease price** u_j if resource j is under-utilized ($s_j > 0$).
  - **Never let prices get negative**; hence the use of the positive part notation $(\cdot)_+$.

Augmented Lagrangian method aka method of multipliers

Dual ascent disadvantage: convergence requires strong conditions. Augmented Lagrangian method transforms the primal problem:

$$
\begin{aligned}
&\min_x f(x) + \frac{\rho}{2}\|Ax - b\|^2 \\
&\text{s.t. } Ax = b
\end{aligned}
$$

Augmented Lagrangian method aka method of multipliers

Dual ascent disadvantage: convergence requires strong conditions. Augmented Lagrangian method transforms the primal problem:

$$
\begin{aligned}
&\min_x f(x) + \frac{\rho}{2}\|Ax - b\|^2 \\
&\text{s.t. } Ax = b
\end{aligned}
$$

where $\rho > 0$ is a parameter. This formulation is clearly equivalent to the original problem. The problem is strongly convex if matrix A has full column rank.

Augmented Lagrangian method aka method of multipliers

Dual ascent disadvantage: convergence requires strong conditions. Augmented Lagrangian method transforms the primal problem:

$$
\begin{split}
&\min_x f(x) + \frac{\rho}{2}\|Ax - b\|^2 \\
&\text{s.t. } Ax = b
\end{split}
$$

where $\rho > 0$ is a parameter. This formulation is clearly equivalent to the original problem. The problem is strongly convex if matrix A has full column rank.

Dual gradient ascent: The iterative updates are given by:

$$
\begin{split}
x_k &= \arg\min_x \left[f(x) + (u_{k-1})^T Ax + \frac{\rho}{2}\|Ax - b\|^2 \right] \\
u_k &= u_{k-1} + \rho(Ax_k - b)
\end{split}
$$

Augmented Lagrangian method aka method of multipliers

Notice step size choice $\alpha_k = \rho$ in dual algorithm. Why?

Since x_k minimizes the function:

$$f(x) + (u_{k-1})^T Ax + \frac{\rho}{2}\|Ax - b\|^2$$

over x, we have the stationarity condition:

$$0 \in \partial f(x_k) + A^T \left(u_{k-1} + \rho(Ax_k - b)\right)$$

which simplifies to:

$$\partial f(x_k) + A^T u_k$$

Augmented Lagrangian method aka method of multipliers

Notice step size choice $\alpha_k = \rho$ in dual algorithm. Why?

Since x_k minimizes the function:

$$f(x) + (u_{k-1})^T Ax + \frac{\rho}{2}\|Ax - b\|^2$$

over x, we have the stationarity condition:

$$0 \in \partial f(x_k) + A^T \left(u_{k-1} + \rho(Ax_k - b)\right)$$

which simplifies to:

$$\partial f(x_k) + A^T u_k$$

This represents the stationarity condition for the original primal problem; under mild conditions, $Ax_k - b \to 0$ as $k \to \infty$, so the KKT conditions are satisfied in the limit and x_k, u_k converge to the solutions.

- **Advantage:** The augmented Lagrangian gives better convergence.

Augmented Lagrangian method aka method of multipliers

Notice step size choice $\alpha_k = \rho$ in dual algorithm. Why?

Since x_k minimizes the function:

$$f(x) + (u_{k-1})^T Ax + \frac{\rho}{2}\|Ax - b\|^2$$

over x, we have the stationarity condition:

$$0 \in \partial f(x_k) + A^T\left(u_{k-1} + \rho(Ax_k - b)\right)$$

which simplifies to:

$$\partial f(x_k) + A^T u_k$$

This represents the stationarity condition for the original primal problem; under mild conditions, $Ax_k - b \to 0$ as $k \to \infty$, so the KKT conditions are satisfied in the limit and x_k, u_k converge to the solutions.

- **Advantage:** The augmented Lagrangian gives better convergence.
- **Disadvantage:** We lose decomposability! (Separability is ruined)

Alternating Direction Method of Multipliers (ADMM)

Alternating direction method of multipliers or ADMM aims for the best of both worlds. Consider the following optimization problem:

Minimize the function:

$$\min_{x,z} f(x) + g(z)$$

$$\text{s.t. } Ax + Bz = c$$

Alternating Direction Method of Multipliers (ADMM)

Alternating direction method of multipliers or ADMM aims for the best of both worlds. Consider the following optimization problem:

Minimize the function:

$$
\begin{aligned}
&\min_{x,z} f(x) + g(z) \\
&\text{s.t. } Ax + Bz = c
\end{aligned}
$$

We augment the objective to include a penalty term for constraint violation:

$$
\begin{aligned}
&\min_{x,z} f(x) + g(z) + \frac{\rho}{2}\|Ax + Bz - c\|^2 \\
&\text{s.t. } Ax + Bz = c
\end{aligned}
$$

Alternating Direction Method of Multipliers (ADMM)

Alternating direction method of multipliers or ADMM aims for the best of both worlds. Consider the following optimization problem:

Minimize the function:

$$
\begin{aligned}
&\min_{x,z} f(x) + g(z) \\
&\text{s.t. } Ax + Bz = c
\end{aligned}
$$

We augment the objective to include a penalty term for constraint violation:

$$
\begin{aligned}
&\min_{x,z} f(x) + g(z) + \frac{\rho}{2}\|Ax + Bz - c\|^2 \\
&\text{s.t. } Ax + Bz = c
\end{aligned}
$$

where $\rho > 0$ is a parameter. The augmented Lagrangian for this problem is defined as:

$$
L_\rho(x, z, u) = f(x) + g(z) + u^T(Ax + Bz - c) + \frac{\rho}{2}\|Ax + Bz - c\|^2
$$

Alternating Direction Method of Multipliers (ADMM)

ADMM repeats the following steps, for $k = 1, 2, 3, \ldots$:

1. Update x:

$$x_k = \arg\min_x L_\rho(x, z_{k-1}, u_{k-1})$$

Alternating Direction Method of Multipliers (ADMM)

ADMM repeats the following steps, for $k = 1, 2, 3, \ldots$:

1. Update x:

$$x_k = \arg\min_x L_\rho(x, z_{k-1}, u_{k-1})$$

2. Update z:

$$z_k = \arg\min_z L_\rho(x_k, z, u_{k-1})$$

Alternating Direction Method of Multipliers (ADMM)

ADMM repeats the following steps, for $k = 1, 2, 3, \ldots$:

1. Update x:

$$x_k = \arg\min_x L_\rho(x, z_{k-1}, u_{k-1})$$

2. Update z:

$$z_k = \arg\min_z L_\rho(x_k, z, u_{k-1})$$

3. Update u:

$$u_k = u_{k-1} + \rho(Ax_k + Bz_k - c)$$

Alternating Direction Method of Multipliers (ADMM)

ADMM repeats the following steps, for $k = 1, 2, 3, \ldots$:

1. Update x:

$$x_k = \arg\min_x L_\rho(x, z_{k-1}, u_{k-1})$$

2. Update z:

$$z_k = \arg\min_z L_\rho(x_k, z, u_{k-1})$$

3. Update u:

$$u_k = u_{k-1} + \rho(Ax_k + Bz_k - c)$$

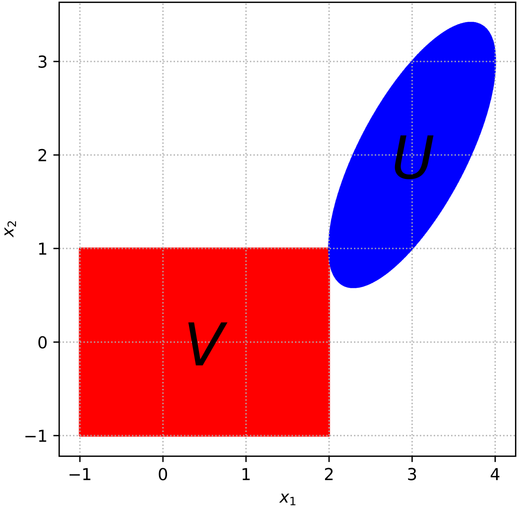
Alternating Direction Method of Multipliers (ADMM)

ADMM repeats the following steps, for $k = 1, 2, 3, \ldots$:

1. Update x:
$$x_k = \arg\min_x L_\rho(x, z_{k-1}, u_{k-1})$$
2. Update z:
$$z_k = \arg\min_z L_\rho(x_k, z, u_{k-1})$$
3. Update u:
$$u_k = u_{k-1} + \rho(Ax_k + Bz_k - c)$$

Note: The usual method of multipliers would replace the first two steps by a joint minimization:
$$(x^{(k)}, z^{(k)}) = \arg\min_{x,z} L_\rho(x, z, u^{(k-1)})$$

Example: Alternating Projections



Consider finding a point in the intersection of convex sets $U, V \subseteq \mathbb{R}^n$:

$$\min_x I_U(x) + I_V(x)$$

To transform this problem into ADMM form, we express it as:

$$\min_{x,z} I_U(x) + I_V(z) \quad \text{subject to} \quad x - z = 0$$

Each ADMM cycle involves two projections:

$$x_k = \arg\min_x P_U(z_{k-1} - w_{k-1})$$
$$z_k = \arg\min_z P_V(x_k + w_{k-1})$$
$$w_k = w_{k-1} + x_k - z_k$$

Sources

- Ryan Tibshirani. Convex Optimization 10-725