Algorithms Primer

Prof. Daniel P. Palomar

ELEC5470/IEDA6100A - Convex Optimization
The Hong Kong University of Science and Technology (HKUST)
Fall 2020-21

Outline

Outline

1. **Unconstrained Optimization**
   - Gradient Descent Method
   - Newton's Method
2. **Constrained Optimization**
   - Equality Constrained Optimization
   - Gradient Projection Method
   - Interior-Point Methods (IPM)
3. **Block Coordinate Algorithms**
   - Gauss-Seidel Algorithm or Block Coordinate Descent (BCD)
   - Jacobi Algorithm

Unconstrained minimization

- Consider the following optimization problem:

$$\underset{\mathbf{x}}{\text{minimize}} \quad f(\mathbf{x})$$

  where f is convex and twice continuously differentiable.

- Optimization methods:
  - produce a sequence of points $\mathbf{x}^k \in \text{dom } f$, $k = 0, 1, \ldots$ with

$$f(\mathbf{x}^k) \to p^\star$$

    where $p^\star$ is the optimal value;
  - equivalently, can be interpreted as iterative methods to solve the optimality condition

$$\nabla f(\mathbf{x}^k) \to \mathbf{0}.$$

- Basic references: (Bertsekas 1999)[1], (Boyd and Vandenberghe 2004)[2], and (Nocedal and Wright 2006)[3].

[1] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.

[2] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[3] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer Verlag, 2006.

Outline

Descent methods

- Descent methods obtain the iterates as follows:

$$\mathbf{x}^{k+1} = \mathbf{x}^k + t^k \Delta \mathbf{x}^k,$$

where $\Delta \mathbf{x}$ is the **search direction** and t is the **stepsize**, satisfying $f(\mathbf{x}^{k+1}) < f(\mathbf{x}^k)$.

- From convexity, the descent condition implies $\nabla f(\mathbf{x})^T \Delta \mathbf{x} < 0$.

Algorithm 1: Descent method

Set $k = 0$ and initialize $\mathbf{x}^0 \in \mathrm{dom}\, f$

repeat

1. Determine a descent direction $\Delta \mathbf{x}^k$.
2. Line search: Choose a stepsize $t^k > 0$.
3. Update: $\mathbf{x}^{k+1} = \mathbf{x}^k + t^k \Delta \mathbf{x}^k$.
4. $k \leftarrow k + 1$

until convergence

return $\mathbf{x}^k$

Line search types

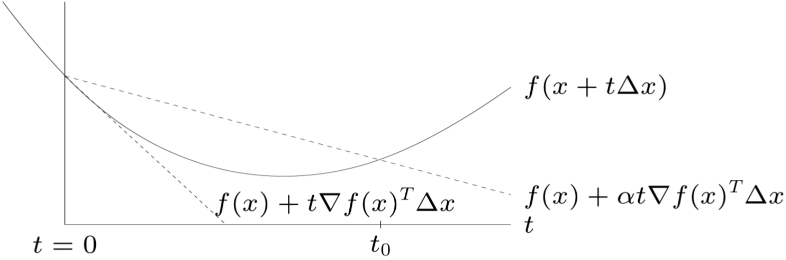
- **Exact line search**:

$$t = \arg\min_{t>0} f(\mathbf{x} + t\Delta\mathbf{x})$$

- **Backtracking line search** (parameters $\alpha \in (0, 1/2)$, $\beta \in (0, 1)$):
  - starting at $t = 1$, repeat $t \leftarrow \beta t$ until

$$f(\mathbf{x} + t\Delta\mathbf{x}) < f(\mathbf{x}) + \alpha t f(\mathbf{x})^T \Delta\mathbf{x}$$

  - graphical interpretation: backtrack until $t \leq t_0$



Gradient descent method

- Simply use the negative gradient as the direction

$$\Delta \mathbf{x} = -\nabla f(\mathbf{x})$$

in the gradient descent method, which satisfies $\nabla f(\mathbf{x})^T \Delta \mathbf{x} < 0$.

- The update is then

$$\mathbf{x}^{k+1} = \mathbf{x}^k - t^k \nabla f(\mathbf{x}^k)$$

- Stopping criterion: usually of the form $\|\nabla f(\mathbf{x})\|_2 \leq \epsilon$.
- Very simple, but often very slow; rarely used in practice.

Gradient descent method

Algorithm 2: Gradient descent method

Set $k = 0$ and initialize $\mathbf{x}^0 \in \text{dom } f$.

repeat

1. Compute the negative gradient as descent direction: $\Delta \mathbf{x}^k = -\nabla f(\mathbf{x}^k)$
2. Line search: Choose a stepsize $t^k > 0$ via exact or bracktracking line search.
3. Update: $\mathbf{x}^{k+1} = \mathbf{x}^k - t^k \nabla f(\mathbf{x}^k)$
4. $k \leftarrow k + 1$

until convergence

return $\mathbf{x}^k$

Convergence of gradient descent method*

- If the exact line search or backtracking line search is used, then every limit point of $\{\mathbf{x}^k\}$ is a stationary point and $f(\mathbf{x}^k) - p^\star \leq c^k (f(\mathbf{x}^0) - p^\star)$ (Boyd and Vandenberghe 2004)[4].

- Other simpler choices for the computation of the stepsize include:
  - constant stepsize: $t^k = t, \qquad k = 0, 1, \ldots$
  - dimishing stepsize rule: $t^k \to 0$ with $\sum_{k=0}^{\infty} t^k = \infty$.
- Other convergence results (Bertsekas 1999)[5]:
  - For the gradient descent with a sufficiently small constant stepsize, every limit point of $\{\mathbf{x}^k\}$ is a stationary point.
  - For the dimishing stepsize rule, every limit point of $\{\mathbf{x}^k\}$ is a stationary point.

[4] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.
[5] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.

Example: Quadratic function

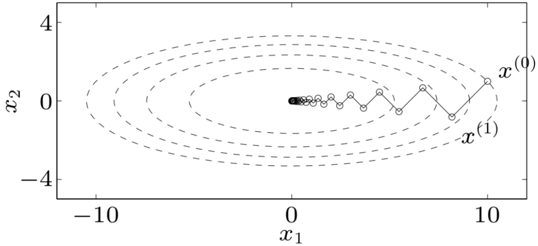
- Consider

$$f(\mathbf{x}) = \frac{1}{2}\left(x_1^2 + \gamma x_2^2\right) \qquad (\gamma > 0)$$

with exact line search, starting at $\mathbf{x}^0 = (\gamma, 1)$:

$$x_1^k = \gamma\left(\frac{\gamma-1}{\gamma+1}\right)^k, \qquad x_2^k = \left(-\frac{\gamma-1}{\gamma+1}\right)^k$$

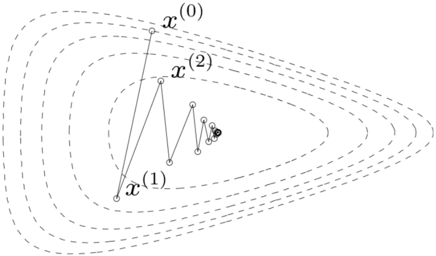
- Very slow if $\gamma \gg 1$ or $\gamma \ll 1$.
- Example for $\gamma = 10$:



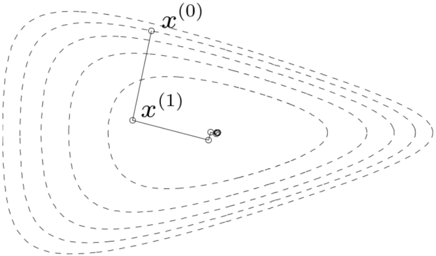
Example: Non-quadratic function

- Consider

$$f(\mathbf{x}) = e^{x_1+3x_2-0.1} + e^{x_1-3x_2-0.1} + e^{-x_1-0.1}$$



backtracking line search



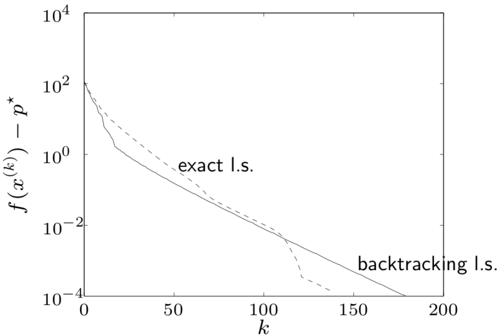
exact line search

Exact vs backtraking line search

- Consider a big problem in $\mathbb{R}^{100}$:

$$f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} - \sum_{i=1}^{500} \log(b_i - \mathbf{a}_i^T\mathbf{x})$$

- Both exact line search and backtraking line search achieve a similar linear convergence (i.e., straight line on a semilog plot):



Outline

Newton step

- Newton's method uses the following direction:

$$\Delta \mathbf{x}_{\text{nt}} = -\nabla^2 f(\mathbf{x})^{-1} \nabla f(\mathbf{x}),$$

where $\nabla^2 f(\mathbf{x})$ is the Hessian of f, which satisfies the descent condition $\nabla f(\mathbf{x})^T \Delta \mathbf{x}_{\text{nt}} < 0$.

- Interpretations:
  - $\mathbf{x} + \Delta \mathbf{x}_{\text{nt}}$ minimizes the second order approximation around $\mathbf{x}$

$$\hat{f}(\mathbf{x} + \mathbf{v}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^T \mathbf{v} + \frac{1}{2} \mathbf{v}^T \nabla^2 f(\mathbf{x}) \mathbf{v}$$

  - $\mathbf{x} + \Delta \mathbf{x}_{\text{nt}}$ solves the linearized (first order approximation) of the optimality condition $\nabla f(\mathbf{x}) = \mathbf{0}$ around $\mathbf{x}$

$$\nabla f(\mathbf{x} + \mathbf{v}) \approx \nabla \hat{f}(\mathbf{x} + \mathbf{v}) = \nabla f(\mathbf{x}) + \nabla^2 f(\mathbf{x}) \mathbf{v} = \mathbf{0}$$

Newton decrement

- The quantity
$$\lambda(\mathbf{x}) = \left(\nabla f(\mathbf{x})^T \nabla^2 f(\mathbf{x})^{-1} \nabla f(\mathbf{x})\right)^{1/2}$$
is a meaure of the proximity of $\mathbf{x}$ to $\mathbf{x}^\star$.

- It gives an estimate of $f(\mathbf{x}) - p^\star$, using a quadratic approximation $\hat{f}$:
$$f(\mathbf{x}) - \inf_{\mathbf{y}} \hat{f}(\mathbf{y}) = \frac{1}{2}\lambda(\mathbf{x})^2.$$

- It's basically free to compute given the Newton step $\Delta\mathbf{x}_{\text{nt}} = -\nabla^2 f(\mathbf{x})^{-1} \nabla f(\mathbf{x})$:
$$\lambda(\mathbf{x})^2 = -\nabla f(\mathbf{x})^T \Delta\mathbf{x}_{\text{nt}}.$$

Newton's method

Algorithm 3: Newton's method

Set $k = 0$, initialize $\mathbf{x}^0 \in \text{dom}\, f$, choose tolerance $\epsilon > 0$.

repeat

1. Compute Newton step and decrement:

$$\Delta \mathbf{x}_{\text{nt}}^k = -\nabla^2 f(\mathbf{x}^k)^{-1} \nabla f(\mathbf{x}^k) \quad \text{and} \quad \lambda(\mathbf{x}^k)^2 = -\nabla f(\mathbf{x}^k)^T \Delta \mathbf{x}_{\text{nt}}^k.$$

2. Stopping criterion: **quit** if $\lambda(\mathbf{x}^k)^2/2 \leq \epsilon$ and **return** $\mathbf{x}^k$.
3. Line search: Choose a stepsize $t^k > 0$ via bracktracking line search.
4. Update: $\mathbf{x}^{k+1} = \mathbf{x}^k + t^k \Delta \mathbf{x}_{\text{nt}}^k$.
5. $k \leftarrow k + 1$

Converge of Newton's method*

Newton's method can be divided into two phases:

- **damped Newton phase**: $(\|\nabla f(\mathbf{x})\|_2 \geq \eta)$
  - most iterations require backtracking steps
  - function value decreases by at least γ
- **quadratically convergent phase**: $(\|\nabla f(\mathbf{x})\|_2 < \eta)$
  - all iterations use stepsize $t = 1$
  - $\|\nabla f(\mathbf{x})\|_2$ converges to zero quadratically.

Conclusion: number of iterations until $f(\mathbf{x}) - p^\star \leq \epsilon$ is bounded above by

$$\frac{f(\mathbf{x}^0) - p^\star}{\gamma} + \log_2 \log_2(\epsilon_0/\epsilon)$$

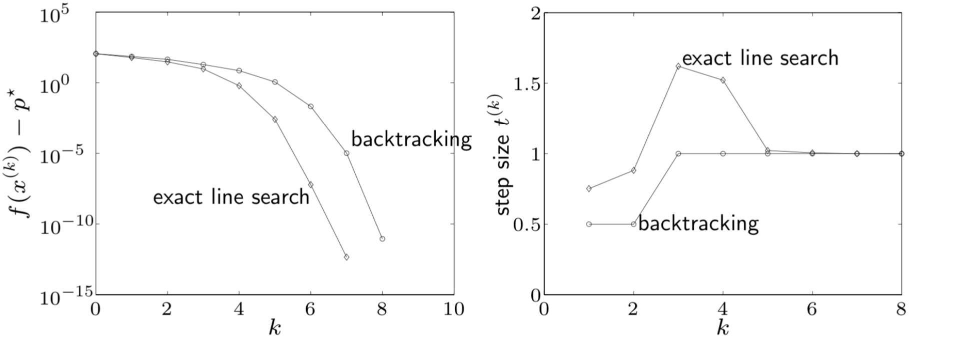
where γ and ϵ_0 are constants that depend on the smoothness of f and $\mathbf{x}^0$ (Boyd and Vandenberghe 2004)[6].

---

[6] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

Example

Example in $\mathbb{R}^{100}$:



- backtracking parameters: $\alpha = 0.01$, $\beta = 0.5$
- backtracking line search almost as fast as exact line search (and much simpler)
- the two phases of the algorithm can be clearly appreciated.

Outline

Outline

Equality constrained optimization

- Consider the following equality constrained optimization problem:

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

  where f is convex and twice continuously differentiable and $\mathbf{A} \in \mathbb{R}^{p \times n}$ is a fat full rank matrix.
- We assume $p^\star$ is finite and attained.
- The Lagrangian of this problem is

$$L(\mathbf{x}; \boldsymbol{\nu}) = f(\mathbf{x}) + \boldsymbol{\nu}^T(\mathbf{A}\mathbf{x} - \mathbf{b})$$

  with gradient

$$\nabla L(\mathbf{x}; \boldsymbol{\nu}) = \nabla f(\mathbf{x}) + \mathbf{A}^T \boldsymbol{\nu}.$$

- **Optimality conditions**: $\mathbf{x}^\star$ is optimal iff there exists a $\boldsymbol{\nu}^\star$ such that

$$\nabla f(\mathbf{x}^\star) + \mathbf{A}^T \boldsymbol{\nu}^\star = 0, \quad \mathbf{A}\mathbf{x}^\star = \mathbf{b}.$$

Eliminating equality constraints

- From linear algebra, we know that we can represent the possibly infinite solutions to $\mathbf{A}\mathbf{x} = \mathbf{b}$ as

$$\{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} = \mathbf{b}\} = \{\mathbf{F}\mathbf{z} + \mathbf{x}_0 \mid \mathbf{z} \in \mathbb{R}^{n-p}\}$$

  where $\mathbf{x}_0$ is any particular solution to $\mathbf{A}\mathbf{x} = \mathbf{b}$ and the range of $\mathbf{F} \in \mathbb{R}^{n\times(n-p)}$ is the nullspace of $\mathbf{A} \in \mathbb{R}^{p\times n}$, i.e., $\mathbf{A}\mathbf{F} = \mathbf{0}$.
- The **reduced or eliminated problem** is

$$\underset{\mathbf{z}}{\text{minimize}} \quad \tilde{f}(\mathbf{z}) = f(\mathbf{F}\mathbf{z} + \mathbf{x}_0)$$

- From the solution $\mathbf{z}^\star$, we can obtain $\mathbf{x}^\star$ and $\boldsymbol{\nu}^\star$ as

$$\mathbf{x}^\star = \mathbf{F}\mathbf{z}^\star + \mathbf{x}_0, \quad \boldsymbol{\nu}^\star = -(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\nabla f(\mathbf{x}^\star).$$

- To use Newton's method on $\tilde{f}(\mathbf{z})$ note that

$$\nabla \tilde{f}(\mathbf{z}) = \mathbf{F}^T \nabla f(\mathbf{x})$$
$$\nabla^2 \tilde{f}(\mathbf{z}) = \mathbf{F}^T \nabla^2 f(\mathbf{x}) \mathbf{F}.$$

Outline

1. Unconstrained Optimization
   - Gradient Descent Method
   - Newton's Method
2. Constrained Optimization
   - Equality Constrained Optimization
   - Gradient Projection Method
   - Interior-Point Methods (IPM)
3. Block Coordinate Algorithms
   - Gauss-Seidel Algorithm or Block Coordinate Descent (BCD)
   - Jacobi Algorithm

Gradient projection method

Consider a convex optimization problem:

$$\begin{array}{ll} \underset{\mathbf{x}}{\text{minimize}} & f(\mathbf{x}) \\ \text{subject to} & \mathbf{x} \in \mathcal{X} \end{array}$$

where $f(\cdot)$ is a convex function and $\mathcal{X}$ represents an arbitrary feasible set (defined by equality and/or inequality constraints).

- If we were to use the gradient descent method $\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha^k \nabla f(\mathbf{x}^k)$ we would possibly end up with an infeasible point $\mathbf{x}^{k+1}$.
- The gradient projection method addresses this issue by projecting onto the feasible set after taking the step (Bertsekas 1999)[7]:

$$\mathbf{x}^{k+1} = \left[\mathbf{x}^k - \alpha^k \nabla f(\mathbf{x}^k)\right]_{\mathcal{X}}$$

where $[\cdot]_{\mathcal{X}}$ denotes projection onto the set $\mathcal{X}$ defined as the solution to $\min_{\mathbf{y}} \|\mathbf{y} - \mathbf{x}\|$ subject to $\mathbf{y} \in \mathcal{X}$.

[7] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.

Gradient projection method

- A slightly more general version of the gradient projection method is to express a feasible direction as $\mathbf{d}^k = \bar{\mathbf{x}}^k - \mathbf{x}^k$ (because $\bar{\mathbf{x}}^k$ is feasible) and write the iteration as (Bertsekas 1999)[8]

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha^k \left(\bar{\mathbf{x}}^k - \mathbf{x}^k \right)$$

where

$$\bar{\mathbf{x}}^k = \left[\mathbf{x}^k - s^k \nabla f \left(\mathbf{x}^k \right) \right]_{\mathcal{X}},$$

$\alpha^k \in (0, 1]$ is a stepsize, and s^k is a positive scalar.

- Note that if we choose $\alpha^k = 1$ then the iteration simplifies to the previous expression:

$$\mathbf{x}^{k+1} = \left[\mathbf{x}^k - s^k \nabla f \left(\mathbf{x}^k \right) \right]_{\mathcal{X}}.$$

- The main limitation of the gradient projection method is to have to compute the projection at each iteration.

[8] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.

Convergence*

- Every limit point of $\{\mathbf{x}^k\}$ is a stationary point (Bertsekas 1999):[9]
  - if s^k is constant and α^k is chosen with the exact line search or backtracking line search;
  - if $\alpha^k = 1$ and s^k is chosen according to the backtracking line search;
  - if $\alpha^k = 1$ and $s^k = s$ with s sufficiently small.

[9] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.

Outline

Inequality constrained optimization

- Consider the following equality constrained optimization problem:

$$\begin{array}{ll} \underset{\mathbf{x}}{\text{minimize}} & f_0(\mathbf{x}) \\ \text{subject to} & f_i(\mathbf{x}) \leq 0, \qquad i = 1, \ldots, m \\ & \mathbf{A}\mathbf{x} = \mathbf{b} \end{array}$$

  where all f_i is convex and twice continuously differentiable and $\mathbf{A} \in \mathbb{R}^{p \times n}$ is a fat full rank matrix.

- We assume $p^\star$ is finite and attained.
- We assume the problem is strictly feasible, hence strong duality holds and dual optimum is attained.

Indicator function

- We can reformulate the original problem with inequality constraints

$$\begin{array}{ll} \underset{\mathbf{x}}{\text{minimize}} & f_0(\mathbf{x}) \\ \text{subject to} & f_i(\mathbf{x}) \leq 0, \qquad i = 1, \dots, m \\ & \mathbf{A}\mathbf{x} = \mathbf{b} \end{array}$$

via the **indicator function** $I_{-}(\cdot)$:

$$\begin{array}{ll} \underset{\mathbf{x}}{\text{minimize}} & f_0(\mathbf{x}) + \sum_{i=1}^{m} I_{-}(f_i(\mathbf{x})) \\ \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{b} \end{array}$$

where

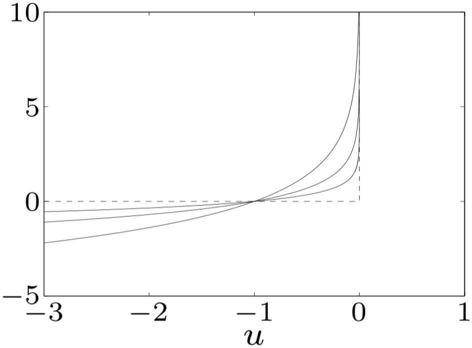
$$I_{-}(u) = \begin{cases} 0 & \text{if} \quad u \leq 0 \\ \infty & \text{otherwise.} \end{cases}$$

Logarithmic barrier

- Then we can approximate the indicator function via the **logarithmic barrier**:

$$\begin{array}{ll}\underset{\mathbf{x}}{\text{minimize}} & f_0(\mathbf{x}) - (1/t)\sum_{i=1}^{m}\log(-f_i(\mathbf{x})) \\ \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{b}\end{array}$$

  which is an equality constrained smooth problem.
- For $t > 0$, $-(1/t)\log(-u)$ is a smooth approximation of $I_-(u)$, which improves as $t \to \infty$.



Logarithmic barrier function

- The logarithmic barrier function is

$$\phi(\mathbf{x}) = -\sum_{i=1}^{m} \log(-f_i(\mathbf{x}))$$

with $\text{dom}\phi = \{x \mid f_1(\mathbf{x}) < 0, \ldots, f_m(\mathbf{x}) < 0\}$.

- It is convex (follows from composition rules).
- Twice continuously differentiable, with derivatives:

$$\nabla\phi(\mathbf{x}) = \sum_{i=1}^{m} \frac{1}{-f_i(\mathbf{x})} \nabla f_i(\mathbf{x})$$

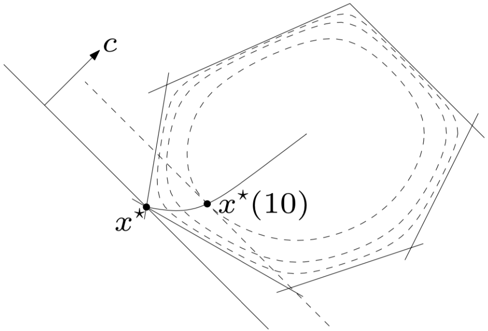
$$\nabla^2\phi(\mathbf{x}) = \sum_{i=1}^{m} \frac{1}{f_i(\mathbf{x})^2} \nabla f_i(\mathbf{x}) \nabla f_i(\mathbf{x})^T + \sum_{i=1}^{m} \frac{1}{-f_i(\mathbf{x})} \nabla^2 f_i(\mathbf{x})$$

Central path

- For $t > 0$, define $\mathbf{x}^\star(t)$ as the solution of

$$\begin{array}{ll} \underset{\mathbf{x}}{\text{minimize}} & tf_0(\mathbf{x}) + \phi(\mathbf{x}) \\ \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{b}. \end{array}$$

- The central path is the curve $\{\mathbf{x}^\star(t) \mid t > 0\}$.
- For example, central path of an LP:



Dual points on central path*

- Central path: $\mathbf{x} = \mathbf{x}^\star(t)$ if there exists a $\mathbf{w}$ such that

$$t\nabla f_0(\mathbf{x}) + \sum_{i=1}^{m} \frac{1}{-f_i(\mathbf{x})} \nabla f_i(\mathbf{x}) + \mathbf{A}^T\mathbf{w} = 0, \quad \mathbf{A}\mathbf{x} = \mathbf{b}$$

- Therefore, $\mathbf{x}^\star(t)$ minimizes the Lagrangian

$$L(\mathbf{x}; \boldsymbol{\lambda}^\star(t), \boldsymbol{\nu}^\star(t)) = f_0(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i^\star(t) f_i(\mathbf{x}) + \boldsymbol{\nu}^\star(t)^T(\mathbf{A}\mathbf{x} - \mathbf{b})$$

  where we define $\lambda_i^\star(t) = 1/(-tf_i(\mathbf{x}^\star(t)))$ and $\boldsymbol{\nu}^\star(t) = \mathbf{w}/t$.
- This confirms the intuitive idea that $f_0(\mathbf{x}^\star(t)) \to p^\star$ if $t \to \infty$:

$$\begin{aligned} p^\star &\geq g(\boldsymbol{\lambda}^\star(t), \boldsymbol{\nu}^\star(t)) \\ &= L(\mathbf{x}^\star(t); \boldsymbol{\lambda}^\star(t), \boldsymbol{\nu}^\star(t)) \\ &= f_0(\mathbf{x}^\star(t)) - m/t. \end{aligned}$$

Interpretation via KKT conditions*

$\mathbf{x} = \mathbf{x}^\star(t)$, $\mathbf{x} = \mathbf{x}^\star(t)$ satisfy

1. Primal feasibility: $f_i(\mathbf{x}) \leq 0, \; i = 1, \ldots, m, \quad \mathbf{A}\mathbf{x} = \mathbf{b}$
2. Dual feasibility: $\boldsymbol{\lambda} \geq \mathbf{0}$
3. Approximate complementary slackness: $-\lambda_i f_i(\mathbf{x}) = 1/t, \; i = 1, \ldots, m$
4. Gradient of Lagrangian with respect to $\mathbf{x}$ vanishes:

$$\nabla f_0(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i \nabla f_i(\mathbf{x}) + \mathbf{A}^T \boldsymbol{\nu} = 0.$$

- The difference with the KKT conditions of the original problem is that condition 3 replaces $\lambda_i f_i(\mathbf{x}) = 0$.

Barrier method

Algorithm 4: Barrier method

Set $k = 0$, initial $\mathbf{x}^0$ strictly feasible, $t^0 > 0$, $\mu > 1$, tolerance $\epsilon > 0$.

repeat

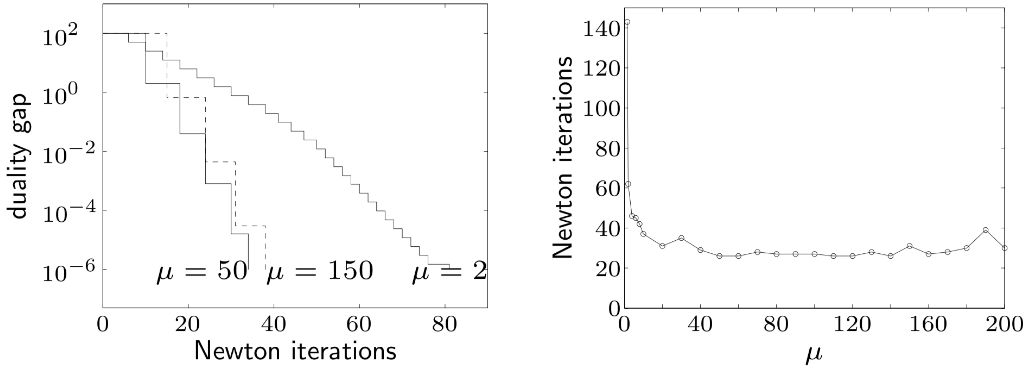
1. Centering step: Compute $\mathbf{x}^\star(t^k)$ by minimizing $t^k f_0(\mathbf{x}) + \phi(\mathbf{x})$ subject to $\mathbf{A}\mathbf{x} = \mathbf{b}$.
2. Stopping criterion: **quit** if $m/t < \epsilon$ and **return** $\mathbf{x}^\star(t^k)$.
3. Increase t: $t^{k+1} \leftarrow \mu t^k$
4. $k \leftarrow k + 1$

- Terminates with $f_0(\mathbf{x}) - p^\star \leq \epsilon$ (follows from $f_0(\mathbf{x}^\star(t)) - p^\star \leq m/t$).
- Centering usually with Newton's method (starting at the current $\mathbf{x}$).
- Choice of μ involves a trade-off: large μ means fewer outer iterations, but more inner (Newton) iterations; typical values are $\mu = 10 \sim 20$.
- For convergence analysis see (Boyd and Vandenberghe 2004)[10].

[10] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

Example

Example with an LP ($m = 100$ inequalities, $n = 50$ variables):



- starts with **x** on central path ($t^0 = 1$, duality gap 100)
- terminates when $t = 10^8$ (gap 10^{-6})

Feasibility and phase I methods

- Recall that the barrier method requires a strictly feasible initial point $\mathbf{x}^0$.
- **Feasibility problem**: find $\mathbf{x}$ such that

$$f_i(\mathbf{x}) \leq 0, \;\; i = 1, \ldots, m, \quad \mathbf{A}\mathbf{x} = \mathbf{b}$$

- How can we find a feasible point?
- **Phase I method**:

$$\begin{array}{ll} \underset{\mathbf{x},s}{\text{minimize}} & s \\ \text{subject to} & f_i(\mathbf{x}) \leq s, \qquad i = 1, \ldots, m \\ & \mathbf{A}\mathbf{x} = \mathbf{b} \end{array}$$

- If the solution $(\mathbf{x}^\star, s^\star)$ satisfies $s^\star < 0$, then $\mathbf{x}^\star$ is strictly feasible in the original problem; otherwise, the original problem is infeasible.
- To solve the phase I problem we can use the barrier method.
- But how do we obtain a stricly feasible point for the phase I method?

Primal-dual interior-point methods

- Primal-dual IPMs are more efficient than the primal barrier method when high accuracy is needed.
- The idea is to update the primal and dual variables at each iterations; so no distinction between inner and outer iterations.
- Often exhibit superlinear asymptotic convergence.
- Search directions can be interpreted as Newton directions for modified KKT conditions.
- Can start at infeasible points.
- Cost per iteration same as barrier method.

Outline

Feasible Cartesian product structure

- Consider a general optimization problem

$$\begin{array}{ll}\underset{\mathbf{x}}{\text{minimize}} & f(\mathbf{x}) \\ \text{subject to} & \mathbf{x} \in \mathcal{X}\end{array}$$

where the optimization variable can be separated into N blocks

$$\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$$

and the feasible set has a **Cartesian product** structure

$$\mathcal{X} = \prod_{i=1}^{N} \mathcal{X}_i.$$

- The problem can be written with decoupled constrains as

$$\begin{array}{lll}\underset{\mathbf{x}}{\text{minimize}} & f(\mathbf{x}_1, \ldots, \mathbf{x}_N) & \\ \text{subject to} & \mathbf{x}_i \in \mathcal{X}_i & i = 1, \ldots, N.\end{array}$$

Outline

Block Coordinate Descent (BCD)

- The **Block Coordinate Descent (BCD) algorithm**, also called nonlinear **Gauss-Seidel algorithm**, optimizes $f(\mathbf{x}_1, \ldots, \mathbf{x}_N)$ sequentially.
- At iteration k, for $i = 1, \ldots, N$:

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i \in \mathcal{X}_i} f\left(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_{i-1}^{k+1}, \mathbf{x}_i, \mathbf{x}_{i+1}^{k} \ldots, \mathbf{x}_{N+1}^{k}\right)$$

- Observe that at each iteration k the blocks are optimized sequentially.
- Merits of BCD:
  1. each subproblem may be much easier to solve, or even may have a closed-form solution;
  2. the objective value is nonincreasing along the BCD updates;
  3. it allows parallel or distributed implementations.

Convergence of BCD*

- Suppose that i) $f(\cdot)$ is continuously differentiable over $\mathcal{X}$ and ii) each block optimization is strictly convex. Then, every limit point of the sequence $\{\mathbf{x}^k\}$ is a stationary point (Bertsekas 1999)[11], (Bertsekas and Tsitsiklis 1997)[12].
- If $\mathcal{X}$ is convex, then the strict convexity of each block optimization can be relaxed to simply having a unique solution.
- Convergence generalizations: it converges in any of the following cases (Grippo and Sciandrone 2000)[13]:
  - the two-block case $N = 2$;
  - $f(\cdot)$ is component-wise strictly quasi-convex w.r.t. $N - 2$ components;
  - $f(\cdot)$ is pseudo-convex.

[11] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.

[12] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

[13] L. Grippo and M. Sciandrone, "On the convergence of the block nonlinear Gauss–Seidel method under convex constraints," *Oper. Res. Lett.*, vol. 26, no. 3, pp. 127–136, 2000.

Application of BCD: $\ell_2 - \ell_1$ optimization problem

- Consider the convex problem

$$\underset{\mathbf{x}}{\text{minimize}} \quad f(\mathbf{x}) \triangleq \frac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1$$

- We can use BCD on each element of $\mathbf{x} = (x_1, \ldots, x_N)$.
- The optimization w.r.t. each block x_i is

$$\underset{x_i}{\text{minimize}} \quad f_i(x_i) \triangleq \frac{1}{2}\|\tilde{\mathbf{y}}_i - \mathbf{a}_i x_i\|_2^2 + \lambda|x_i|$$

where $\tilde{\mathbf{y}}_i \triangleq \mathbf{y} - \sum_{j \neq i} \mathbf{a}_j x_j$.

- The optimal x_i has a closed-form update:

$$x_i^\star = \text{soft}_\lambda\left(\mathbf{a}_i^T \tilde{\mathbf{y}}_i\right) / \|\mathbf{a}_i\|^2$$

where $\text{soft}_\lambda(u) \triangleq \text{sign}(u)\left[|u| - \lambda\right]_+$ is the **soft-thresholding** operator ($[\cdot]_+ \triangleq \max\{\cdot, 0\}$).

Soft-thresholding operator

- Consider the problem

$$\underset{x_i}{\text{minimize}} \quad \frac{1}{2}\|\tilde{\mathbf{y}}_i - \mathbf{a}_i x_i\|_2^2 + \lambda |x_i|$$

- Assuming $x_i > 0$, the objective becomes $\frac{1}{2}\|\mathbf{a}_i\|^2 x_i^2 - \tilde{\mathbf{y}}_i^T \mathbf{a}_i x_i + \lambda x_i$ and setting the gradient to zero we get

$$x_i = \left(\tilde{\mathbf{y}}_i^T \mathbf{a}_i - \lambda\right) / \|\mathbf{a}_i\|^2$$

  which implies $\tilde{\mathbf{y}}_i^T \mathbf{a}_i > \lambda > 0$.
- Assuming $x_i < 0$, the objective becomes $\frac{1}{2}\|\mathbf{a}_i\|^2 x_i^2 - \tilde{\mathbf{y}}_i^T \mathbf{a}_i x_i - \lambda x_i$ and setting the gradient to zero we get

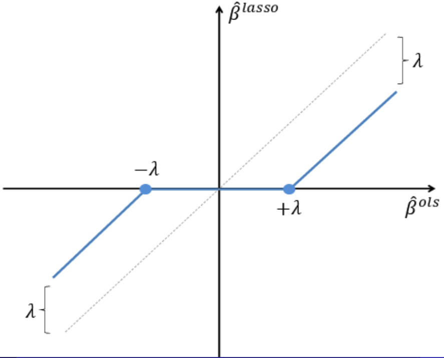
$$x_i = \left(\tilde{\mathbf{y}}_i^T \mathbf{a}_i + \lambda\right) / \|\mathbf{a}_i\|^2$$

  which implies $\tilde{\mathbf{y}}_i^T \mathbf{a}_i < -\lambda < 0$.
- The last case is when $\tilde{\mathbf{y}}_i^T \mathbf{a}_i \in [-\lambda, \lambda]$ (equivalently, $|\tilde{\mathbf{y}}_i^T \mathbf{a}_i| \leq \lambda$), in which case $x_i = 0$.

Soft-thresholding operator

- Recall that
  - if $\tilde{\mathbf{y}}_i^T\mathbf{a}_i > \lambda$: $x_i = \left(\tilde{\mathbf{y}}_i^T\mathbf{a}_i - \lambda\right)/\|\mathbf{a}_i\|^2 = \left(|\tilde{\mathbf{y}}_i^T\mathbf{a}_i| - \lambda\right)/\|\mathbf{a}_i\|^2$
  - if $\tilde{\mathbf{y}}_i^T\mathbf{a}_i < -\lambda$: $x_i = \left(\tilde{\mathbf{y}}_i^T\mathbf{a}_i + \lambda\right)/\|\mathbf{a}_i\|^2 = -\left(|\tilde{\mathbf{y}}_i^T\mathbf{a}_i| - \lambda\right)/\|\mathbf{a}_i\|^2$
- Together with the case x_i when $|\tilde{\mathbf{y}}_i^T\mathbf{a}_i| \leq \lambda$, we can finally write the solution in a compact form:

$$x_i = \text{sign}(\tilde{\mathbf{y}}_i^T\mathbf{a}_i)\left[|\tilde{\mathbf{y}}_i^T\mathbf{a}_i| - \lambda\right]_+ /\|\mathbf{a}_i\|^2.$$



Outline

Jacobi Algorithm

- The **Jacobi algorithm** is similar to the Gauss-Seiden algorithm but, instead of sequentially, it optimizes $f(\mathbf{x}_1, \ldots, \mathbf{x}_N)$ in parallel.
- At iteration k, for $i = 1, \ldots, N$:

$$\mathbf{x}_i = \arg\min_{\mathbf{x}_i} f\left(\mathbf{x}_1^k, \ldots, \mathbf{x}_{i-1}^k, \mathbf{x}_i, \mathbf{x}_{i+1}^k \ldots, \mathbf{x}_{N+1}^k\right)$$

- Observe that at each iteration k all the blocks are optimized in parallel.
- Convergence is more difficult to establish.
- If the mapping defined by $T(\mathbf{x}) = \mathbf{x} - \gamma\nabla f(\mathbf{x})$ is a contraction for some γ, then $\{\mathbf{x}^k\}$ converges to solution $\mathbf{x}^\star$ geometrically (Bertsekas 1999)[14].

[14] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.

Thanks

For more information visit:

https://www.danielppalomar.com

References I

Bertsekas, D. P. (1999). *Nonlinear programming*. Athena Scientific.

Bertsekas, D. P., & Tsitsiklis, J. N. (1997). *Parallel and distributed computation: Numerical methods*. Athena Scientific.

Boyd, S. P., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Grippo, L., & Sciandrone, M. (2000). On the convergence of the block nonlinear Gauss–Seidel method under convex constraints. *Oper. Res. Lett.*, *26*(3), 127–136.

Nocedal, J., & Wright, S. J. (2006). *Numerical optimization*. Springer Verlag.