Quasi-Newton Methods

CME 307 / MS&E 311 /OIT 676 Optimization

Wenzhi Gao 

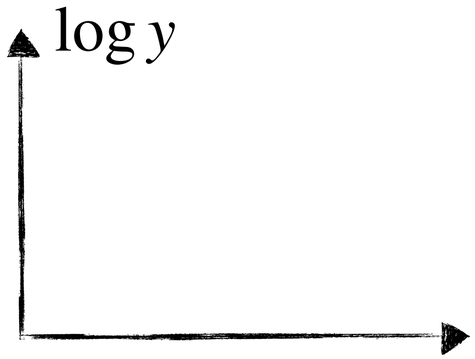
ICME, Stanford University

Review

L-smoothness $\quad \|\nabla f(x) - \nabla f(y)\| \le L\|x - y\| \qquad \|\nabla^2 f(x)\| \le L$

Descent lemma $\quad f(x) \le f(y) + \langle \nabla f(y), x - y\rangle + \frac{L}{2}\|x - y\|^2$

Gradient descent $\quad f(x + d) \le f(x) + \underbrace{\langle \nabla f(x), d\rangle + \frac{L}{2}\|d\|^2}_{d^\star = -\frac{1}{L}\nabla f(x)}$



$$f(x - \frac{1}{L}\nabla f(x)) \le f(x) - \frac{1}{2L}\|\nabla f(x)\|^2$$

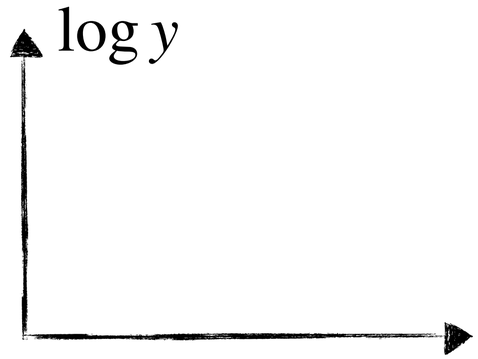
$$f(x^{k+1}) \le f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|^2$$

Review

L-smoothness $\quad \|\nabla f(x) - \nabla f(y)\| \le L\|x-y\| \qquad \|\nabla^2 f(x)\| \le L$

Descent lemma $\quad f(x) \le f(y) + \langle \nabla f(y), x-y\rangle + \frac{L}{2}\|x-y\|^2$

Gradient descent $\quad f(x+d) \le f(x) + \underbrace{\langle \nabla f(x), d\rangle + \frac{L}{2}\|d\|^2}_{d^\star = -\frac{1}{L}\nabla f(x)}$



PL $\quad \|\nabla f(x)\|^2 \ge 2\mu[f(x) - f(x^\star)]$

$$f(x^{k+1}) - f(x^\star) \le (1-\frac{\mu}{L})[f(x^k) - f(x^\star)]$$

$$f(x - \frac{1}{L}\nabla f(x)) \le f(x) - \frac{1}{2L}\|\nabla f(x)\|^2$$

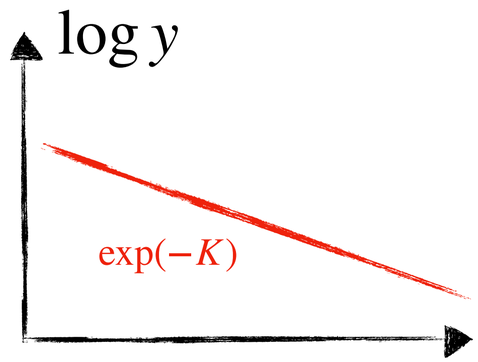
$$f(x^{k+1}) \le f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|^2$$

Review

L-smoothness $\quad \|\nabla f(x) - \nabla f(y)\| \le L\|x-y\| \qquad \|\nabla^2 f(x)\| \le L$

Descent lemma $\quad f(x) \le f(y) + \langle \nabla f(y), x-y\rangle + \frac{L}{2}\|x-y\|^2$

Gradient descent $\quad f(x+d) \le f(x) + \underbrace{\langle \nabla f(x), d\rangle + \frac{L}{2}\|d\|^2}_{d^\star = -\frac{1}{L}\nabla f(x)}$



PL $\quad \|\nabla f(x)\|^2 \ge 2\mu[f(x) - f(x^\star)]$

$$f(x^{k+1}) - f(x^\star) \le (1-\frac{\mu}{L})[f(x^k) - f(x^\star)]$$

$$f(x - \frac{1}{L}\nabla f(x)) \le f(x) - \frac{1}{2L}\|\nabla f(x)\|^2$$

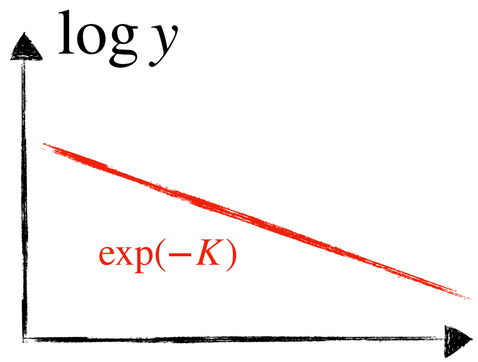
$$f(x^{k+1}) \le f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|^2$$

Review

L-smoothness $\quad \|\nabla f(x) - \nabla f(y)\| \le L\|x-y\| \qquad \|\nabla^2 f(x)\| \le L$

Descent lemma $\quad f(x) \le f(y) + \langle \nabla f(y), x-y\rangle + \frac{L}{2}\|x-y\|^2$

Gradient descent $\quad f(x+d) \le f(x) + \underbrace{\langle \nabla f(x), d\rangle + \frac{L}{2}\|d\|^2}_{d^\star = -\frac{1}{L}\nabla f(x)}$



PL $\quad \|\nabla f(x)\|^2 \ge 2\mu[f(x) - f(x^\star)]$

$$f(x - \tfrac{1}{L}\nabla f(x)) \le f(x) - \tfrac{1}{2L}\|\nabla f(x)\|^2$$

$$f(x^{k+1}) \le f(x^k) - \tfrac{1}{2L}\|\nabla f(x^k)\|^2$$

$$f(x^{k+1}) - f(x^\star) \le (1 - \tfrac{\mu}{L})[f(x^k) - f(x^\star)]$$

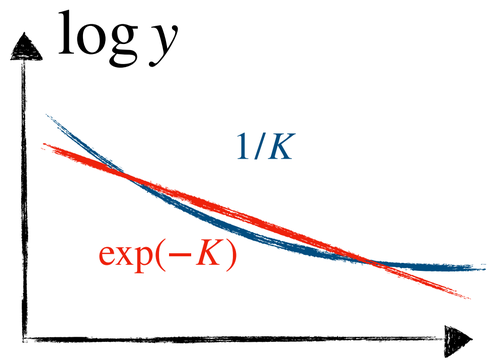
$$\tfrac{1}{2L}\sum_{k=1}^{K} \|\nabla f(x^k)\|^2 \le f(x^1) - f(x^\star)$$

Review

L-smoothness $\quad \|\nabla f(x) - \nabla f(y)\| \le L\|x-y\| \qquad \|\nabla^2 f(x)\| \le L$

Descent lemma $\quad f(x) \le f(y) + \langle \nabla f(y), x-y\rangle + \frac{L}{2}\|x-y\|^2$

Gradient descent $\quad f(x+d) \le f(x) + \underbrace{\langle \nabla f(x), d\rangle + \frac{L}{2}\|d\|^2}_{d^\star = -\frac{1}{L}\nabla f(x)}$



PL $\quad \|\nabla f(x)\|^2 \ge 2\mu[f(x) - f(x^\star)]$

$$f(x - \tfrac{1}{L}\nabla f(x)) \le f(x) - \tfrac{1}{2L}\|\nabla f(x)\|^2$$

$$f(x^{k+1}) \le f(x^k) - \tfrac{1}{2L}\|\nabla f(x^k)\|^2$$

$$f(x^{k+1}) - f(x^\star) \le (1 - \tfrac{\mu}{L})[f(x^k) - f(x^\star)]$$

$$\tfrac{1}{2L}\sum_{k=1}^{K} \|\nabla f(x^k)\|^2 \le f(x^1) - f(x^\star)$$

$$\min_k \|\nabla f(x^k)\|^2 \le \frac{2L(f(x^1) - f(x^\star))}{K}$$

Newton's method

Gradient descent $$f(x+d) \leq f(x) + \underbrace{\langle \nabla f(x), d\rangle + \frac{L}{2}\|d\|^2}_{d^\star = -\frac{1}{L}\nabla f(x)}$$

Second-order expansion

$$f(x+d) = f(x) + \underbrace{\langle \nabla f(x), d\rangle + \frac{1}{2}\langle d, \nabla^2 f(x) d\rangle}_{d^\star = -\,[\nabla^2 f(x)]^{-1}\nabla f(x)} + \mathcal{O}(\|d\|^3)$$

When f is convex quadratic, expansion is global and convergence in one step

Newton's method

Gradient descent $$f(x+d) \leq f(x) + \underbrace{\langle \nabla f(x), d\rangle + \frac{L}{2}\|d\|^2}_{d^\star = -\frac{1}{L}\nabla f(x)}$$

Second-order expansion

$$f(x+d) = f(x) + \underbrace{\langle \nabla f(x), d\rangle + \frac{1}{2}\langle d, \nabla^2 f(x) d\rangle}_{d^\star = -\,[\nabla^2 f(x)]^{-1}\nabla f(x)} + \mathcal{O}(\|d\|^3)$$

When f is convex quadratic, expansion is global and convergence in one step

When f is not quadratic, it is still (locally) a quadratic

Local quadratic convergence

"Local" means when $x \to x^\star$

$$f(x) = f(x^\star) + \langle \nabla f(x^\star), x - x^\star \rangle + \frac{1}{2}\langle x - x^\star, \nabla^2 f(x^\star)(x - x^\star) \rangle + \mathcal{O}(\|x - x^\star\|^3)$$

Local quadratic convergence

"Local" means when $x \to x^\star$

$$f(x) = f(x^\star) + \frac{1}{2}\langle x - x^\star, \nabla^2 f(x^\star)(x - x^\star)\rangle + \mathcal{O}(\|x - x^\star\|^3)$$

Local quadratic convergence

"Local" means when $x \to x^\star$

$$f(x) = f(x^\star) + \frac{1}{2}\langle x - x^\star, \nabla^2 f(x^\star)(x - x^\star)\rangle + \mathcal{O}(\|x - x^\star\|^3)$$

Newton's step converge in one step on the local quadratic

$\mathcal{O}(\|x - x^\star\|^3)$ slows it down $\Rightarrow$ superlinear convergence

Local quadratic convergence

"Local" means when $x \to x^\star$

$$f(x) = f(x^\star) + \frac{1}{2}\langle x - x^\star, \nabla^2 f(x^\star)(x - x^\star)\rangle + \mathcal{O}(\|x - x^\star\|^3)$$

Newton's step converge in one step on the local quadratic

$\mathcal{O}(\|x - x^\star\|^3)$ slows it down $\Rightarrow$ superlinear convergence

$\mathcal{O}(\|x - x^\star\|^3)$ can be quantified under Lipschitz Hessian

$$\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq H\|x - y\|$$

Local quadratic convergence

"Local" means when $x \to x^\star$

$$f(x) = f(x^\star) + \frac{1}{2}\langle x - x^\star, \nabla^2 f(x^\star)(x - x^\star)\rangle + \mathcal{O}(\|x - x^\star\|^3)$$

Newton's step converge in one step on the local quadratic

$\mathcal{O}(\|x - x^\star\|^3)$ slows it down $\Rightarrow$ superlinear convergence

$\mathcal{O}(\|x - x^\star\|^3)$ can be quantified under Lipschitz Hessian

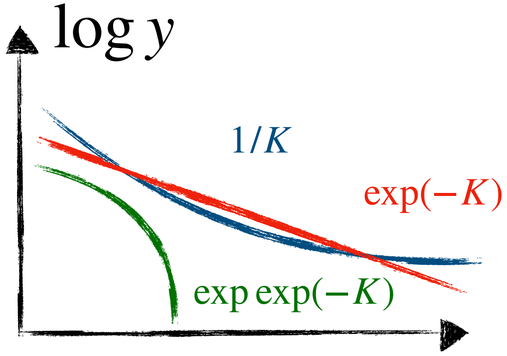
$$\|\nabla^3 f(x)\| \leq H$$

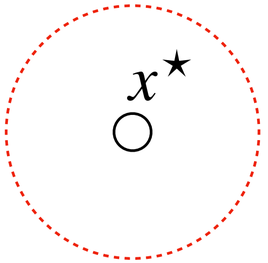
$$\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq H\|x - y\|$$

Local quadratic convergence

Theorem. Let $x^+ = x - [\nabla^2 f(x)]^{-1} \nabla f(x)$ and $\nabla^2 f(x) \succeq \mu I$

$$\|x^+ - x^\star\| \leq \frac{H}{\mu} \|x - x^\star\|^2$$

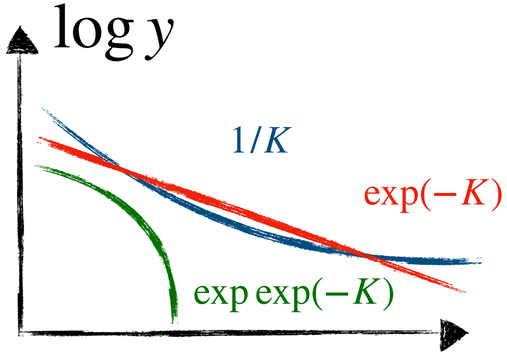




Local quadratic convergence

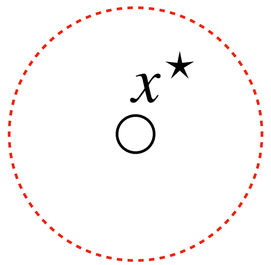
Theorem. Let $x^+ = x - [\nabla^2 f(x)]^{-1} \nabla f(x)$ and $\nabla^2 f(x) \succeq \mu I$

$$\|x^+ - x^\star\| \leq \frac{H}{\mu} \|x - x^\star\|^2$$



$$10^{-2} \to 10^{-4} \to 10^{-8} \to 10^{-16}$$

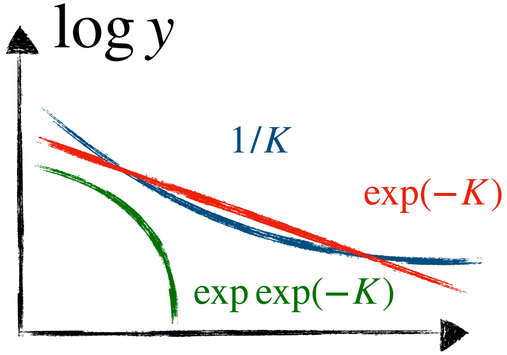
- When $\|x - x^\star\| \leq \frac{\mu}{H}$, the algorithm keeps converging



Local quadratic convergence

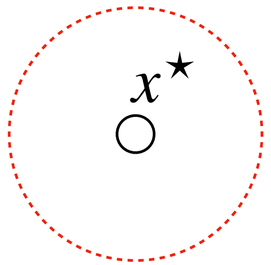
Theorem. Let $x^+ = x - [\nabla^2 f(x)]^{-1} \nabla f(x)$ and $\nabla^2 f(x) \succeq \mu I$

$$\|x^+ - x^\star\| \le \frac{H}{\mu} \|x - x^\star\|^2$$



$$10^{-2} \to 10^{-4} \to 10^{-8} \to 10^{-16}$$

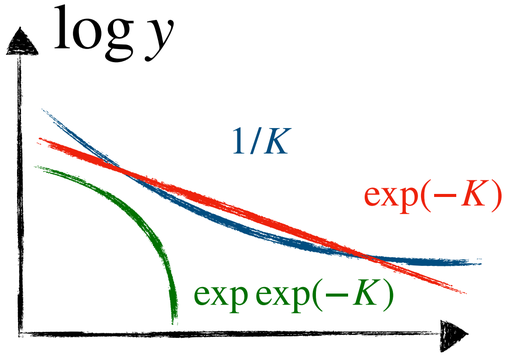
- When $\|x - x^\star\| \le \frac{\mu}{H}$, the algorithm keeps converging
- Otherwise, Newton's method can diverge



Local quadratic convergence

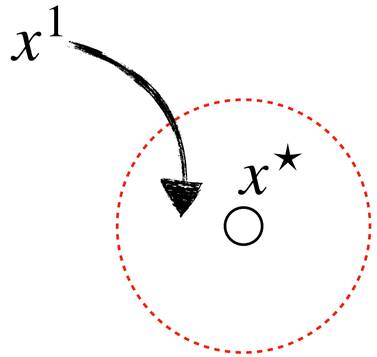
Theorem. Let $x^+ = x - [\nabla^2 f(x)]^{-1} \nabla f(x)$ and $\nabla^2 f(x) \succeq \mu I$

$$\|x^+ - x^\star\| \leq \frac{H}{\mu} \|x - x^\star\|^2$$



$$10^{-2} \to 10^{-4} \to 10^{-8} \to 10^{-16}$$

- When $\|x - x^\star\| \leq \frac{\mu}{H}$, the algorithm keeps converging
- Otherwise, Newton's method can diverge
- The process that ensures $x \to x^\star$ is known as globalization
  line-search, trust-region …



Review

- Smoothness and gradient descent
- Order of convergence (sublinear/linear/superlinear)
- Newton's method and globalization

Today's goal: invent a quasi-Newton method by yourself

How to strategically solve homework questions?

You just got assigned a homework before the weekend

Problem 1. Matrix completion
Tabular data with missing entries is a common phenomenon in data science in machine learning. It can be useful to predict the missing values in the table from known entries. This problem is called the *matrix completion* problem, as the table of data can be represented as a matrix. Consider an example: consider an online business that has collected ratings by m users for n items. The (i, j)th entry of the matrix gives user i's rating for item j, if user i has rated item j. The goal of matrix completion in this context is to estimate the missing ratings. Table 1 gives an example of this setting.

User	Item 1	Item 2	Item 3	Item 4
User 1	5	*	3	4
User 2	4	3	*	2
User 3	*	4	5	3
User 4	3	*	4	5
User 5	2	5	*	4

Table 1: Example ratings matrix with missing values represented by an asterisk.

To estimate these missing values requires further assumptions. One natural assumption is that each user i's preferences can be represented by latent vector $u_i \in \mathbf{R}^k$, and each item's characteristics can be represented by latent vector $v_j \in \mathbf{R}^k$, so that user i's rating for item j is given by $u_i^T v_j$. This assumption guarantees the true matrix of ratings has rank k. However, k is not typically known in advance, so we prefer to solve a convex relaxation of the problem that automatically chooses the rank. Under this assumption, we can (approximately) recover the data matrix by solving the optimization problem

$$\begin{array}{ll} \text{minimize} & \|X\|_*, \\ \text{subject to} & X_{ij} = M_{ij} \quad (i,j) \in \Omega, \\ \text{variables} & X \in \mathbf{R}^{m \times n}, \end{array} \tag{1}$$

where $\|\cdot\|_*$ denotes the nuclear norm (sum of the singular values), and Ω is the set of ordered pairs corresponding to known ratings. The nuclear norm objective $\|X\|_*$ is a convex surrogate for the rank of X. Define $X^\star$ to be the solution of (1).

(a) The matrix completion problem (1) is equivalent to the following semi-definite program (SDP):

$$\begin{array}{ll} \text{minimize} & \text{trace}(Y) + \text{trace}(Z), \\ \text{subject to} & X_{ij} = M_{ij} \quad (i,j) \in \Omega, \\ & \begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} \succeq 0, \\ \text{variables} & X \in \mathbf{R}^{m \times n}, Y \in \mathbf{R}^{m \times m}, Z \in \mathbf{R}^{n \times n}. \end{array} \tag{2}$$

The equivalence between (1) and (2) is challenging to prove. Instead, we would like you to prove the following result, which is part of the equivalence proof:

Let $X \in \mathbf{R}^{m \times n}$ and $t \in \mathbf{R}$. Show that if $\|X\|_* \leq t$, then there exist $Y \in \mathbf{R}^{m \times m}$ and $Z \in \mathbf{R}^{n \times n}$ such that

$$\begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} \succeq 0, \quad \text{trace}(Y) + \text{trace}(Z) \leq 2t.$$

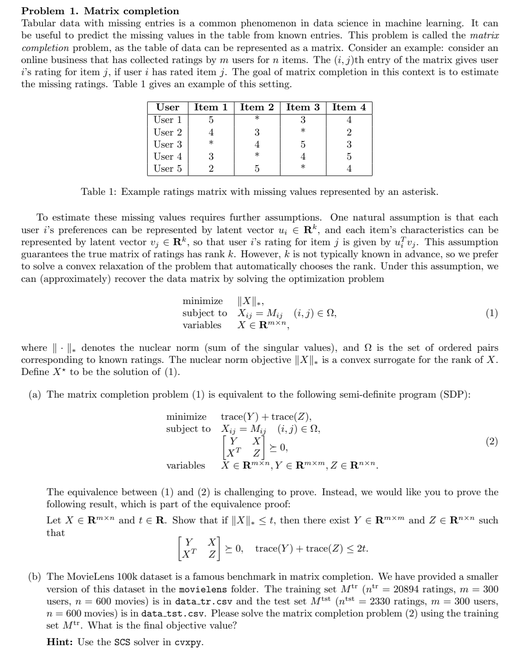
(b) The MovieLens 100k dataset is a famous benchmark in matrix completion. We have provided a smaller version of this dataset in the `movielens` folder. The training set M^{tr} ($n^{\text{tr}} = 20894$ ratings, $m = 300$ users, $n = 600$ movies) is in `data_tr.csv` and the test set M^{tst} ($n^{\text{tst}} = 2330$ ratings, $m = 300$ users, $n = 600$ movies) is in `data_tst.csv`. Please solve the matrix completion problem (2) using the training set M^{tr}. What is the final objective value?

Hint: Use the `SCS` solver in `cvxpy`.

How to strategically solve homework questions?

You just got assigned a homework before the weekend

- Evil TAs 😈 created N homework problems
- Each question takes κ units of effort ☕
- 😈 host K sessions, asking about one of them
  $K = 1$ in CME 307
- If 😈 asks about an unfinished question,
  then you have to finish it and lose 1 unit of ☕



Problem 1. Matrix completion

Tabular data with missing entries is a common phenomenon in data science in machine learning. It can be useful to predict the missing values in the table from known entries. This problem is called the *matrix completion* problem, as the table of data can be represented as a matrix. Consider an example: consider an online business that has collected ratings by m users for n items. The (i, j)th entry of the matrix gives user i's rating for item j, if user i has rated item j. The goal of matrix completion in this context is to estimate the missing ratings. Table 1 gives an example of this setting.

User	Item 1	Item 2	Item 3	Item 4
User 1	5	*	3	4
User 2	4	3	*	2
User 3	*	4	5	3
User 4	3	*	4	5
User 5	2	5	*	4

Table 1: Example ratings matrix with missing values represented by an asterisk.

To estimate these missing values requires further assumptions. One natural assumption is that each user i's preferences can be represented by latent vector $u_i \in \mathbf{R}^k$, and each item's characteristics can be represented by latent vector $v_j \in \mathbf{R}^k$, so that user i's rating for item j is given by $u_i^T v_j$. This assumption guarantees the true matrix of ratings has rank k. However, k is not typically known in advance, so we prefer to solve a convex relaxation of the problem that automatically chooses the rank. Under this assumption, we can (approximately) recover the data matrix by solving the optimization problem

$$\begin{array}{ll} \text{minimize} & \|X\|_*, \\ \text{subject to} & X_{ij} = M_{ij} \quad (i,j) \in \Omega, \\ \text{variables} & X \in \mathbf{R}^{m \times n}, \end{array} \tag{1}$$

where $\|\cdot\|_*$ denotes the nuclear norm (sum of the singular values), and Ω is the set of ordered pairs corresponding to known ratings. The nuclear norm objective $\|X\|_*$ is a convex surrogate for the rank of X. Define $X^\star$ to be the solution of (1).

(a) The matrix completion problem (1) is equivalent to the following semi-definite program (SDP):

$$\begin{array}{ll} \text{minimize} & \text{trace}(Y) + \text{trace}(Z), \\ \text{subject to} & X_{ij} = M_{ij} \quad (i,j) \in \Omega, \\ & \begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} \succeq 0, \\ \text{variables} & X \in \mathbf{R}^{m \times n}, Y \in \mathbf{R}^{m \times m}, Z \in \mathbf{R}^{n \times n}. \end{array} \tag{2}$$

The equivalence between (1) and (2) is challenging to prove. Instead, we would like you to prove the following result, which is part of the equivalence proof:

Let $X \in \mathbf{R}^{m \times n}$ and $t \in \mathbf{R}$. Show that if $\|X\|_* \leq t$, then there exist $Y \in \mathbf{R}^{m \times m}$ and $Z \in \mathbf{R}^{n \times n}$ such that

$$\begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} \succeq 0, \quad \text{trace}(Y) + \text{trace}(Z) \leq 2t.$$

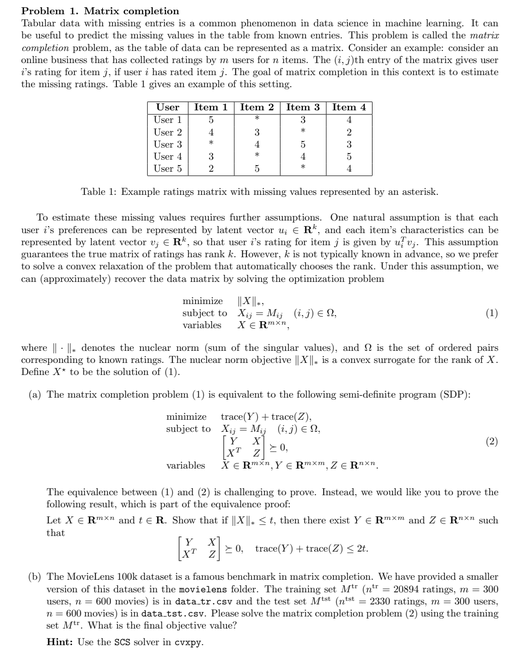
(b) The MovieLens 100k dataset is a famous benchmark in matrix completion. We have provided a smaller version of this dataset in the `movielens` folder. The training set M^{tr} ($n^{\text{tr}} = 20894$ ratings, $m = 300$ users, $n = 600$ movies) is in `data_tr.csv` and the test set M^{tst} ($n^{\text{tst}} = 2330$ ratings, $m = 300$ users, $n = 600$ movies) is in `data_tst.csv`. Please solve the matrix completion problem (2) using the training set M^{tr}. What is the final objective value?

Hint: Use the `SCS` solver in `cvxpy`.

How to strategically solve homework questions?

You just got assigned a homework before the weekend

- Evil TAs 😈 created N homework problems
- Each question takes κ units of effort ☕
- 😈 host K sessions, asking about one of them
  $K = 1$ in CME 307
- If 😈 asks about an unfinished question,
  then you have to finish it and lose 1 unit of ☕



Problem 1. Matrix completion

Tabular data with missing entries is a common phenomenon in data science in machine learning. It can be useful to predict the missing values in the table from known entries. This problem is called the *matrix completion* problem, as the table of data can be represented as a matrix. Consider an example: consider an online business that has collected ratings by m users for n items. The (i, j)th entry of the matrix gives user i's rating for item j, if user i has rated item j. The goal of matrix completion in this context is to estimate the missing ratings. Table 1 gives an example of this setting.

User	Item 1	Item 2	Item 3	Item 4
User 1	5	*	3	4
User 2	4	3	*	2
User 3	*	4	5	3
User 4	3	*	4	5
User 5	2	5	*	4

Table 1: Example ratings matrix with missing values represented by an asterisk.

To estimate these missing values requires further assumptions. One natural assumption is that each user i's preferences can be represented by latent vector $u_i \in \mathbf{R}^k$, and each item's characteristics can be represented by latent vector $v_j \in \mathbf{R}^k$, so that user i's rating for item j is given by $u_i^T v_j$. This assumption guarantees the true matrix of ratings has rank k. However, k is not typically known in advance, so we prefer to solve a convex relaxation of the problem that automatically chooses the rank. Under this assumption, we can (approximately) recover the data matrix by solving the optimization problem

$$\begin{array}{ll} \text{minimize} & \|X\|_*, \\ \text{subject to} & X_{ij} = M_{ij} \quad (i,j) \in \Omega, \\ \text{variables} & X \in \mathbf{R}^{m \times n}, \end{array} \tag{1}$$

where $\|\cdot\|_*$ denotes the nuclear norm (sum of the singular values), and Ω is the set of ordered pairs corresponding to known ratings. The nuclear norm objective $\|X\|_*$ is a convex surrogate for the rank of X. Define $X^\star$ to be the solution of (1).

(a) The matrix completion problem (1) is equivalent to the following semi-definite program (SDP):

$$\begin{array}{ll} \text{minimize} & \text{trace}(Y) + \text{trace}(Z), \\ \text{subject to} & X_{ij} = M_{ij} \quad (i,j) \in \Omega, \\ & \begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} \succeq 0, \\ \text{variables} & X \in \mathbf{R}^{m \times n}, Y \in \mathbf{R}^{m \times m}, Z \in \mathbf{R}^{n \times n}. \end{array} \tag{2}$$

The equivalence between (1) and (2) is challenging to prove. Instead, we would like you to prove the following result, which is part of the equivalence proof:

Let $X \in \mathbf{R}^{m \times n}$ and $t \in \mathbf{R}$. Show that if $\|X\|_* \leq t$, then there exist $Y \in \mathbf{R}^{m \times m}$ and $Z \in \mathbf{R}^{n \times n}$ such that

$$\begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} \succeq 0, \quad \text{trace}(Y) + \text{trace}(Z) \leq 2t.$$

(b) The MovieLens 100k dataset is a famous benchmark in matrix completion. We have provided a smaller version of this dataset in the `movielens` folder. The training set M^{tr} ($n^{\text{tr}} = 20894$ ratings, $m = 300$ users, $n = 600$ movies) is in `data_tr.csv` and the test set M^{tst} ($n^{\text{tst}} = 2330$ ratings, $m = 300$ users, $n = 600$ movies) is in `data_tst.csv`. Please solve the matrix completion problem (2) using the training set M^{tr}. What is the final objective value?

Hint: Use the `SCS` solver in `cvxpy`.

You want to enjoy a good weekend 🏖️

What to do?

N questions, K sessions, κ ☕ per problem

Option 1. Finish all the questions over the weekend. It costs

What to do?

N questions, K sessions, κ ☕ per problem

Option 1. Finish all the questions over the weekend. It costs

$N\kappa$ units of ☕

What to do?

N questions, K sessions, κ ☕ per problem

Option 1. Finish all the questions over the weekend. It costs

$N\kappa$ units of ☕

- A lot of effort and the loss of a weekend 💔
- But TA sessions will be no problem

Average effort over K sessions:

What to do?

N questions, K sessions, κ ☕ per problem

Option 1. Finish all the questions over the weekend. It costs

$N\kappa$ units of ☕

- A lot of effort and the loss of a weekend 💔
- But TA sessions will be no problem

Average effort over K sessions:

$$\frac{N\kappa + \sum_{k=1}^{K} 0}{K} = \frac{N\kappa}{K}$$

What to do?

N questions, K sessions, κ ☕ per problem

Option 2. Slack off and enjoy ⛱. It costs

What to do?

N questions, K sessions, κ ☕ per problem

Option 2. Slack off and enjoy ⛱️. It costs

0 units of ☕

What to do?

N questions, K sessions, κ ☕ per problem

Option 2. Slack off and enjoy ⛱️. It costs

$$0 \text{ units of } ☕$$

- No effort over the weekend
- But TA sessions can be thrilling

What to do?

N questions, K sessions, κ ☕ per problem

Option 2. Slack off and enjoy ⛱. It costs

0 units of ☕

- No effort over the weekend
- But TA sessions can be thrilling

What happens after all K sessions?

Slack off

N questions, K sessions, κ ☕ per problem

Let g_k be the effort spent on the session k

Let V_k denote the #unsolved questions before session k. We have $V_1 = N$

In session k, two possibilities:

Slack off

N questions, K sessions, κ ☕ per problem

Let g_k be the effort spent on the session k

Let V_k denote the #unsolved questions before session k. We have $V_1 = N$

In session k, two possibilities:

- TA asks about the question you happened to solve 😊

$$g_k = 0 \qquad\qquad V_{k+1} = V_k$$

Slack off

N questions, K sessions, κ ☕ per problem

Let g_k be the effort spent on the session k

Let V_k denote the #unsolved questions before session k. We have $V_1 = N$

In session k, two possibilities:

- TA asks about the question you happened to solve 😊

$$g_k = 0 \qquad\qquad V_{k+1} = V_k$$

- TA asks about the question you didn't solve 😨

$$g_k = \kappa + 1 \qquad\qquad V_{k+1} = V_k - 1$$

Cost of slack off

N questions, K sessions, κ ☕ per problem

V unsolved problems

😊 $g_k = 0$

$V_{k+1} = V_k - 0$

😨 $g_k = \kappa + 1$

$V_{k+1} = V_k - 1$

N questions, K sessions, κ ☕ per problem

V unsolved problems

Cost of slack off

😊 $g_k = 0$

$V_{k+1} = V_k - 0$

$$V_{k+1} = V_k - \frac{g_k}{\kappa + 1}$$

😨 $g_k = \kappa + 1$

$V_{k+1} = V_k - 1$

N questions, K sessions, κ ☕ per problem
V unsolved problems

Cost of slack off

😊 $g_k = 0$

$V_{k+1} = V_k - 0$

😨 $g_k = \kappa + 1$

$V_{k+1} = V_k - 1$

$$V_{k+1} = V_k - \frac{g_k}{\kappa + 1} \quad \Rightarrow \quad g_k = (\kappa + 1)(V_k - V_{k+1})$$

N questions, K sessions, κ ☕ per problem
V unsolved problems

Cost of slack off

😊 $g_k = 0$

$V_{k+1} = V_k - 0$

😨 $g_k = \kappa + 1$

$V_{k+1} = V_k - 1$

$$V_{k+1} = V_k - \frac{g_k}{\kappa + 1} \quad \Rightarrow \quad g_k = (\kappa + 1)(V_k - V_{k+1})$$

- Whenever you are penalized, you solve more questions
- What's the worst-case average effort?

$$\frac{0 + \sum_{k=1}^{K} g_k}{K}$$

N questions, K sessions, κ ☕ per problem
V unsolved problems

Cost of slack off

😊 $g_k = 0$

$V_{k+1} = V_k - 0$

😨 $g_k = \kappa + 1$

$V_{k+1} = V_k - 1$

$$V_{k+1} = V_k - \frac{g_k}{\kappa + 1} \quad \Rightarrow \quad g_k = (\kappa + 1)(V_k - V_{k+1})$$

- Whenever you are penalized, you solve more questions
- What's the worst-case average effort?

$$\frac{0 + \sum_{k=1}^{K} g_k}{K} = \frac{(\kappa + 1) \sum_{k=1}^{K} (V_k - V_{k+1})}{K}$$

N questions, K sessions, κ ☕ per problem
V unsolved problems

Cost of slack off

😊 $g_k = 0$

$V_{k+1} = V_k - 0$

😨 $g_k = \kappa + 1$

$V_{k+1} = V_k - 1$

$$V_{k+1} = V_k - \frac{g_k}{\kappa + 1} \quad \Rightarrow \quad g_k = (\kappa + 1)(V_k - V_{k+1})$$

- Whenever you are penalized, you solve more questions
- What's the worst-case average effort?

$$\frac{0 + \sum_{k=1}^{K} g_k}{K} = \frac{(\kappa + 1)\sum_{k=1}^{K}(V_k - V_{k+1})}{K} = \frac{(\kappa + 1)(V_1 - V_{K+1})}{K}$$

N questions, K sessions, κ ☕ per problem
V unsolved problems

Cost of slack off

😊 $g_k = 0$

$V_{k+1} = V_k - 0$

😨 $g_k = \kappa + 1$

$V_{k+1} = V_k - 1$

$$V_{k+1} = V_k - \frac{g_k}{\kappa+1} \quad \Rightarrow \quad g_k = (\kappa+1)(V_k - V_{k+1})$$

- Whenever you are penalized, you solve more questions
- What's the worst-case average effort?

$$\frac{0 + \sum_{k=1}^{K} g_k}{K} = \frac{(\kappa+1)\sum_{k=1}^{K}(V_k - V_{k+1})}{K} = \frac{(\kappa+1)(V_1 - V_{K+1})}{K} \leq \frac{(\kappa+1)N}{K}$$

Slack off or not to slack off

	Work over weekend	Slack-off
Weekend ☕	$N\kappa$	0
Session ☕	0	$\sum_k g_k \leq (\kappa+1)N$
Average effort	$\frac{N\kappa}{K}$	$\frac{N(\kappa+1)}{K}$
Peak effort	$N\kappa$	$1+\kappa$

Slack off or not to slack off

	Work over weekend	Slack-off
Weekend ☕	$N\kappa$	0
Session ☕	0	$\sum_k g_k \leq (\kappa+1)N$
Average effort	$\frac{N\kappa}{K}$	$\frac{N(\kappa+1)}{K}$
Peak effort	$N\kappa$	$1+\kappa$

- Peak effort is greatly reduced when N is large
- Average effort is increased by N/K, fine if $K \gg 1$ and κ is large
- If lucky, you don't have to do all questions

Slack off or not to slack off

	Work over weekend	Slack-off
Weekend ☕	$N\kappa$	0
Session ☕	0	$\sum_k g_k \le (\kappa+1)N$
Average effort	$\frac{N\kappa}{K}$	$\frac{N(\kappa+1)}{K}$
Peak effort	$N\kappa$	$1+\kappa$

- Peak effort is greatly reduced when N is large
- Average effort is increased by N/K, fine if $K \gg 1$ and κ is large
- If lucky, you don't have to do all questions

This is quasi-Newton method

Quasi-Newton method

- Amortization
  Split a large bulk of effort into small blocks

Weekend vs. slack-off

- Learning from losses/mistakes
  Make up for it when slack-off causes problems

$$V_{k+1} = V_k - \frac{g_k}{\kappa + 1}$$

Quasi-Newton uses this idea to compute A^{-1}

Quadratic minimization

Positive definite quadratic minimization problem with $LI \succeq A \succeq \mu I, \kappa = L/\mu$

$$\min_{x \in \mathbb{R}^n} \quad f(x) = \frac{1}{2}\langle x, Ax \rangle - \langle b, x \rangle$$

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

preconditioner

- The problem is equivalent to solving $\nabla f(x) = Ax - b = 0$
- Gradient descent with $P = A^{-1}$ solves the problem in one step

$$x - A^{-1}\nabla f(x) = x - A^{-1}A(x - x^\star) = x^\star$$

- We want fast convergence (e.g. $\frac{\|\nabla f(x^{K+1})\|^2}{\|\nabla f(x^1)\|^2}$) after K iterations

$$\min_{x \in \mathbb{R}^n} \quad f(x) = \frac{1}{2}\langle x, Ax \rangle - \langle b, x \rangle$$
$$x^+ = x - P\nabla f(x)$$

What to do?

Option 1. Compute the perfect preconditioner A^{-1} over the weekend. It costs

$$\Omega(n^3) \text{ flops and } \Omega(n^2) \text{ memory}$$

- For dense matrices, computationally intensive for $n > 10^5$
- But once finished, we can obtain perfect progress in x

$$\min_{x\in\mathbb{R}^n} \quad f(x) = \frac{1}{2}\langle x, Ax\rangle - \langle b, x\rangle$$
$$x^+ = x - P\nabla f(x)$$

What to do?

Option 1. Compute the perfect preconditioner A^{-1} over the weekend. It costs

$$\Omega(n^3) \text{ flops and } \Omega(n^2) \text{ memory}$$

- For dense matrices, computationally intensive for $n > 10^5$
- But once finished, we can obtain perfect progress in x

Let $g_k = \dfrac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} = \left\|(I - AP_k)\dfrac{\nabla f(x^k)}{\|\nabla f(x^k)\|}\right\|^2$ denote the contraction

$$g_k = 0 \qquad \Rightarrow \qquad \frac{1}{K}\sum_{k=1}^{K} g_k = \frac{1}{K}\sum_{k=1}^{K} 0 = 0$$

What to do?

N questions, K sessions, κ ☕ per problem

Option 2. Slack off and use whatever P. It costs

$$0 \text{ flops}$$

- No effort over the weekend
- But using a bad P can cause divergence

What happens after all K iterations?

Slack off

In iteration k, two possibilities:

Slack off

In iteration k, two possibilities:

😊 P_k happens to produce nice contraction $g_k \approx 0$

Slack off

In iteration k, two possibilities:

- 😊 P_k happens to produce nice contraction $g_k \approx 0$
- 😨 P_k causes serious divergence $g_k \gg 0$

Slack off

In iteration k, two possibilities:

😊 P_k happens to produce nice contraction $g_k \approx 0$

😨 P_k causes serious divergence $g_k \gg 0$

P_k does not work well for x^k (😈), so it needs fixing

Repairing bad P

$$g_k = \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} = \left\|(I - AP_k)\frac{\nabla f(x^k)}{\|\nabla f(x^k)\|}\right\|^2$$

Let V_k denote $\mathrm{dist}(P_k, A^{-1}) := \|I - AP_k\|_F^2$ before iteration k.

Repairing bad P

$$g_k = \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} = \left\|(I - AP_k)\frac{\nabla f(x^k)}{\|\nabla f(x^k)\|}\right\|^2$$

Let V_k denote $\mathrm{dist}(P_k, A^{-1}) := \|I - AP_k\|_F^2$ before iteration k.

Repairing P means reducing $\|I - AP\|_F^2$, and we have

$$g_k = \left\|(I - AP)\frac{\nabla f(x^k)}{\|\nabla f(x^k)\|}\right\|^2 > 0$$

- We need P_{k+1} so that $\|I - AP_{k+1}\|_F^2 < \|I - AP_k\|_F^2$
- Update from P_k to P_{k+1} should be inexpensive

Repairing bad P

$$g_k = \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} = \left\|(I - AP_k)\frac{\nabla f(x^k)}{\|\nabla f(x^k)\|}\right\|^2$$

$$P_+ = P + \eta uv^\top, \quad \|u\| = \|v\| = 1$$

Repairing bad P

$$g_k = \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} = \left\|(I - AP_k)\frac{\nabla f(x^k)}{\|\nabla f(x^k)\|}\right\|^2$$

$$P_+ = P + \eta uv^\top, \quad \|u\| = \|v\| = 1$$

$$\min_\eta \|I - A(P + \eta uv^\top)\|_F^2 = \|I - AP - \eta Auv^\top\|_F^2$$

Repairing bad P

$$g_k = \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} = \left\|(I - AP_k)\frac{\nabla f(x^k)}{\|\nabla f(x^k)\|}\right\|^2$$

$$P_+ = P + \eta uv^\top, \quad \|u\| = \|v\| = 1$$

$$\min_\eta \|I - A(P + \eta uv^\top)\|_F^2 = \|I - AP - \eta Auv^\top\|_F^2$$

Some tedius derivation

$$\eta^\star = \frac{\langle Au, (I - AP)v)\rangle}{\|Au\|^2} \qquad u = \frac{(I - AP)v}{\|(I - AP)v\|}$$

Repairing bad P

$$g_k = \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} = \left\|(I - AP_k)\frac{\nabla f(x^k)}{\|\nabla f(x^k)\|}\right\|^2$$

$$P_+ = P + \eta uv^\top, \quad \|u\| = \|v\| = 1$$

$$\min_\eta \|I - A(P + \eta uv^\top)\|_F^2 = \|I - AP - \eta Auv^\top\|_F^2$$

Some tedius derivation

$$\eta^\star = \frac{\langle Au, (I - AP)v)\rangle}{\|Au\|^2} \qquad u = \frac{(I - AP)v}{\|(I - AP)v\|}$$

and

$$\|AP_+ - I\|_F^2 \leq \|AP - I\|_F^2 - \frac{1}{\kappa}\|(I - AP)v\|^2$$

Repairing bad P

$$g_k = \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} = \left\|(I - AP_k)\frac{\nabla f(x^k)}{\|\nabla f(x^k)\|}\right\|^2$$

$$V_k = \|I - AP_k\|_F^2$$

For any $\|v\| = 1$

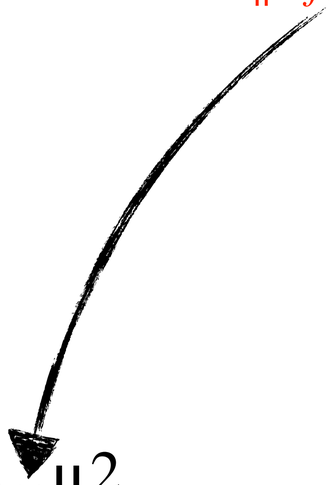
$$\|AP_{k+1} - I\|_F^2 \leq \|AP_k - I\|_F^2 - \frac{1}{\kappa}\|(I - AP_k)v\|^2$$

How to choose v to ensure useful repair when $g_k \gg 0$?

Repairing bad P

$$g_k = \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} = \left\|(I - AP_k)\frac{\nabla f(x^k)}{\|\nabla f(x^k)\|}\right\|^2$$

$$V_k = \|I - AP_k\|_F^2$$



For any $\|v\| = 1$

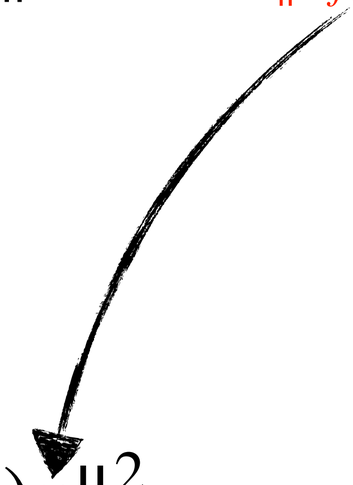
$$\|AP_{k+1} - I\|_F^2 \leq \|AP_k - I\|_F^2 - \frac{1}{\kappa}\|(I - AP_k)v\|^2$$

How to choose v to ensure useful repair when $g_k \gg 0$?

Repairing bad P

$$g_k = \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} = \left\|(I - AP_k)\frac{\nabla f(x^k)}{\|\nabla f(x^k)\|}\right\|^2$$

$$V_k = \|I - AP_k\|_F^2$$



For any $\|v\| = 1$

$$\|AP_{k+1} - I\|_F^2 \leq \|AP_k - I\|_F^2 - \frac{1}{\kappa}\|(I - AP_k)v\|^2$$

How to choose v to ensure useful repair when $g_k \gg 0$?

$$v = \frac{\nabla f(x^k)}{\|\nabla f(x^k)\|} \quad \Rightarrow \quad V_{k+1} \leq V_k - \frac{1}{\kappa} g_k$$

Algorithm

Initialize: slack off 🏖️ and have some random P_1 and x^1

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

Get contraction

$$g_k = \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2}$$

$$P_{k+1} = P_k + \eta^\star u(v)^\top$$

Repair P_k

$$V_{k+1} \leq V_k - \frac{1}{\kappa} g_k$$

$$v = \frac{\nabla f(x^k)}{\|\nabla f(x^k)\|} \quad u = \frac{(I - AP_k)v^k}{\|(I - AP_k)v^k\|} \quad \eta^\star = \frac{\langle Au, (I - AP_k)v)\rangle}{\|Au\|^2}$$

What's the cost of slack-off?

Cost of slack-off

Slack-off for CME 307

Slack-off for A^{-1}

Weekend/factorization

Session/Gradient descent

Avg. effort/contraction

Peak effort

Cost of slack-off

	Slack-off for CME 307	Slack-off for A^{-1}
Weekend/factorization	0	0
Session/Gradient descent		
Avg. effort/contraction		
Peak effort		

Cost of slack-off

	Slack-off for CME 307	Slack-off for A^{-1}
Weekend/factorization	0	0
Session/Gradient descent	$V_{k+1} = V_k - \frac{1}{\kappa+1} g_k$	$V_{k+1} \leq V_k - \frac{1}{\kappa} g_k$
Avg. effort/contraction		
Peak effort		

Cost of slack-off

	Slack-off for CME 307	Slack-off for A^{-1}
Weekend/factorization	0	0
Session/Gradient descent	$V_{k+1} = V_k - \frac{1}{\kappa+1} g_k$	$V_{k+1} \leq V_k - \frac{1}{\kappa} g_k$
Avg. effort/contraction	$\frac{(\kappa+1)V_1}{K}$	$\frac{\kappa V_1}{K}$
Peak effort		

Cost of slack-off

	Slack-off for CME 307	Slack-off for A^{-1}
Weekend/factorization	0	0
Session/Gradient descent	$V_{k+1} = V_k - \frac{1}{\kappa+1} g_k$	$V_{k+1} \leq V_k - \frac{1}{\kappa} g_k$
Avg. effort/contraction	$\frac{(\kappa+1)V_1}{K}$	$\frac{\kappa V_1}{K}$
Peak effort		n^2

Cost of slack-off

	Slack-off for CME 307	Slack-off for A^{-1}
Weekend/factorization	0	0
Session/Gradient descent	$V_{k+1} = V_k - \frac{1}{\kappa+1} g_k$	$V_{k+1} \leq V_k - \frac{1}{\kappa} g_k$
Avg. effort/contraction	$\frac{(\kappa+1)V_1}{K}$	$\frac{\kappa V_1}{K}$
Peak effort		n^2

The average contraction satisfies $\frac{1}{K}\sum_{k=1}^{K} g_k \leq \frac{\kappa V_1}{K}$

Cost of slack-off

	Slack-off for CME 307	Slack-off for A^{-1}
Weekend/factorization	0	0
Session/Gradient descent	$V_{k+1} = V_k - \frac{1}{\kappa+1} g_k$	$V_{k+1} \leq V_k - \frac{1}{\kappa} g_k$
Avg. effort/contraction	$\frac{(\kappa+1)V_1}{K}$	$\frac{\kappa V_1}{K}$
Peak effort		n^2

The average contraction satisfies $\frac{1}{K}\sum_{k=1}^{K} g_k \leq \frac{\kappa V_1}{K}$

But what we really want is

$$\frac{\|\nabla f(x^{K+1})\|^2}{\|\nabla f(x^1)\|^2}$$

Cost of slack-off

	Slack-off for CME 307	Slack-off for A^{-1}
Weekend/factorization	0	0
Session/Gradient descent	$V_{k+1} = V_k - \frac{1}{\kappa+1} g_k$	$V_{k+1} \leq V_k - \frac{1}{\kappa} g_k$
Avg. effort/contraction	$\frac{(\kappa+1)V_1}{K}$	$\frac{\kappa V_1}{K}$
Peak effort		n^2

The average contraction satisfies $\frac{1}{K}\sum_{k=1}^{K} g_k \leq \frac{\kappa V_1}{K}$

But what we really want is

$$\frac{\|\nabla f(x^{K+1})\|^2}{\|\nabla f(x^1)\|^2} = \prod_{k=1}^{K} \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2}$$

Cost of slack-off

	Slack-off for CME 307	Slack-off for A^{-1}
Weekend/factorization	0	0
Session/Gradient descent	$V_{k+1} = V_k - \frac{1}{\kappa+1} g_k$	$V_{k+1} \leq V_k - \frac{1}{\kappa} g_k$
Avg. effort/contraction	$\frac{(\kappa+1)V_1}{K}$	$\frac{\kappa V_1}{K}$
Peak effort		n^2

The average contraction satisfies $\frac{1}{K}\sum_{k=1}^{K} g_k \leq \frac{\kappa V_1}{K}$

But what we really want is

$$\frac{\|\nabla f(x^{K+1})\|^2}{\|\nabla f(x^1)\|^2} = \prod_{k=1}^{K} \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} = \prod_{k=1}^{K} g_k$$

Cost of slack-off

	Slack-off for CME 307	Slack-off for A^{-1}
Weekend/factorization	0	0
Session/Gradient descent	$V_{k+1} = V_k - \frac{1}{\kappa+1} g_k$	$V_{k+1} \leq V_k - \frac{1}{\kappa} g_k$
Avg. effort/contraction	$\frac{(\kappa+1)V_1}{K}$	$\frac{\kappa V_1}{K}$
Peak effort		n^2

The average contraction satisfies $\frac{1}{K}\sum_{k=1}^{K} g_k \leq \frac{\kappa V_1}{K}$

But what we really want is

$$\frac{\|\nabla f(x^{K+1})\|^2}{\|\nabla f(x^1)\|^2} = \prod_{k=1}^{K} \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} = \prod_{k=1}^{K} g_k \leq \left(\frac{1}{K}\sum_{k=1}^{K} g_k\right)^K$$

Cost of slack-off

	Slack-off for CME 307	Slack-off for A^{-1}
Weekend/factorization	0	0
Session/Gradient descent	$V_{k+1} = V_k - \frac{1}{\kappa+1} g_k$	$V_{k+1} \leq V_k - \frac{1}{\kappa} g_k$
Avg. effort/contraction	$\frac{(\kappa+1)V_1}{K}$	$\frac{\kappa V_1}{K}$
Peak effort		n^2

The average contraction satisfies $\frac{1}{K}\sum_{k=1}^{K} g_k \leq \frac{\kappa V_1}{K}$

But what we really want is

$$\frac{\|\nabla f(x^{K+1})\|^2}{\|\nabla f(x^1)\|^2} = \prod_{k=1}^{K} \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} = \prod_{k=1}^{K} g_k \leq \left(\frac{1}{K}\sum_{k=1}^{K} g_k\right)^K \leq \left(\frac{\kappa V_1}{K}\right)^K$$

Cost of slack-off

	Slack-off for CME 307	Slack-off for A^{-1}
Weekend/factorization	0	0
Session/Gradient descent	$V_{k+1} = V_k - \frac{1}{\kappa+1} g_k$	$V_{k+1} \leq V_k - \frac{1}{\kappa} g_k$
Avg. effort/contraction	$\frac{(\kappa+1)V_1}{K}$	$\frac{\kappa V_1}{K}$
Peak effort		n^2

The average contraction satisfies $\frac{1}{K}\sum_{k=1}^{K} g_k \leq \frac{\kappa V_1}{K}$

But what we really want is

$$\frac{\|\nabla f(x^{K+1})\|^2}{\|\nabla f(x^1)\|^2} = \prod_{k=1}^{K} \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} = \prod_{k=1}^{K} g_k \leq \left(\frac{1}{K}\sum_{k=1}^{K} g_k\right)^K \leq \left(\frac{\kappa V_1}{K}\right)^K$$

Superlinear convergence

Idea so far

- Slack off and don't compute A^{-1}
- Go to iteration $x^{k+1} = x^k - P_k \nabla f(x^k)$ and receive contraction
- Exact minimization reduces V_k

$$\min_{\eta} \|I - A(P + \eta uv^\top)\|_F^2$$

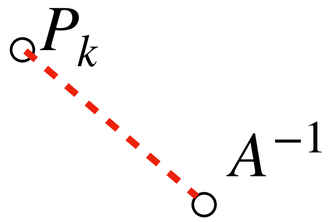
Idea so far

- Slack off and don't compute A^{-1}
- Go to iteration $x^{k+1} = x^k - P_k \nabla f(x^k)$ and receive contraction
- Exact minimization reduces V_k

$$\min_{\eta} \|I - A(P + \eta uv^\top)\|_F^2$$

Is there a more systematic way to reduce V_k?

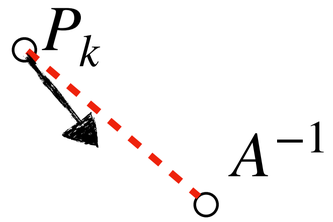
Descent direction



Let $V_k = \|P_k - A^{-1}\|_F^2$ or some smooth distance function

How to reduce a smooth function?

Descent direction



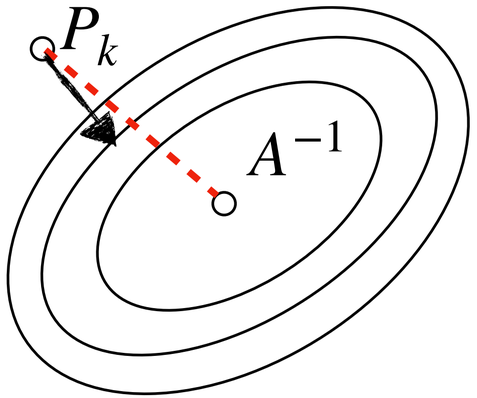
Let $V_k = \|P_k - A^{-1}\|_F^2$ or some smooth distance function

How to reduce a smooth function?

OK to have a descent direction

$$\langle G, \nabla_P \left[\frac{1}{2}\|P - A^{-1}\|_F^2\right]\rangle = \langle G, P - A^{-1}\rangle < 0$$

Descent direction



Let $V_k = \|P_k - A^{-1}\|_F^2$ or some smooth distance function

How to reduce a smooth function?

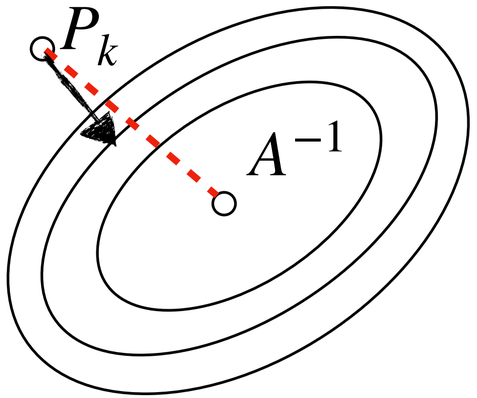
OK to have a descent direction

$$\langle G, \nabla_P \left[\frac{1}{2}\|P - A^{-1}\|_F^2\right]\rangle = \langle G, P - A^{-1}\rangle < 0$$

Given convex $l(P)$ minimized by A^{-1}

$$\langle -\nabla l(P), P - A^{-1}\rangle \leq l(A^{-1}) - l(P) \leq 0$$

Descent direction



Let $V_k = \|P_k - A^{-1}\|_F^2$ or some smooth distance function

How to reduce a smooth function?

OK to have a descent direction

$$\langle G, \nabla_P \left[\frac{1}{2}\|P - A^{-1}\|_F^2\right]\rangle = \langle G, P - A^{-1}\rangle < 0$$

Given convex $l(P)$ minimized by A^{-1}

$$\langle -\nabla l(P), P - A^{-1}\rangle \leq l(A^{-1}) - l(P) \leq 0$$

Negative gradient $-\nabla l(P)$ is a descent direction for $\|P - A^{-1}\|_F^2$

Finding a convex function

What convex function is minimized A^{-1}?

Finding a convex function

What convex function is minimized A^{-1}?

$$x - A^{-1} \nabla f(x) = x^\star$$

Finding a convex function

What convex function is minimized A^{-1}?

$$x - A^{-1}\nabla f(x) = x^\star$$

$$l(P) = \frac{1}{2}\|x - P\nabla f(x) - x^\star\|^2$$

Finding a convex function

What convex function is minimized A^{-1}?

$$l(P) = \frac{1}{2}\|x - P\nabla f(x) - x^\star\|^2$$

$$x - A^{-1}\nabla f(x) = x^\star$$

$$z - A^{-1}\nabla f(z) = x^\star$$

Finding a convex function

What convex function is minimized A^{-1}?

$$l(P) = \frac{1}{2}\|x - P\nabla f(x) - x^\star\|^2$$

$$x - A^{-1}\nabla f(x) = x^\star \quad \searrow$$

$$z - A^{-1}\nabla f(z) = x^\star \quad \nearrow$$

Finding a convex function

What convex function is minimized A^{-1}?

$$l(P) = \frac{1}{2}\|x - P\nabla f(x) - x^\star\|^2$$

$$x - A^{-1}\nabla f(x) = x^\star \quad \twoheadrightarrow$$
$$z - A^{-1}\nabla f(z) = x^\star \quad \twoheadrightarrow$$

$$\underbrace{x - z}_{s} = A^{-1}(\underbrace{\nabla f(x) - \nabla f(z)}_{y})$$

Finding a convex function

What convex function is minimized A^{-1}?

$$l(P) = \frac{1}{2}\|x - P\nabla f(x) - x^\star\|^2$$

$$x - A^{-1}\nabla f(x) = x^\star \quad \twoheadrightarrow$$
$$z - A^{-1}\nabla f(z) = x^\star \quad \twoheadrightarrow$$

$$\underbrace{x - z}_{s} = A^{-1}(\underbrace{\nabla f(x) - \nabla f(z)}_{y})$$

Define function $l_{x,z}(P) := \mathbb{I}\{Py = s\}$ or $l_{x,z}(P) := \|Py - s\|^2$

Finding a convex function

What convex function is minimized A^{-1}?

$$l(P) = \frac{1}{2}\|x - P\nabla f(x) - x^\star\|^2$$

$$x - A^{-1}\nabla f(x) = x^\star \quad \twoheadrightarrow \quad \underbrace{x - z}_{s} = A^{-1}(\underbrace{\nabla f(x) - \nabla f(z)}_{y})$$
$$z - A^{-1}\nabla f(z) = x^\star \quad \twoheadrightarrow$$

Define function $l_{x,z}(P) := \mathbb{I}\{Py = s\}$ or $l_{x,z}(P) := \|Py - s\|^2$

- Learning from the (penalized) secant equation

Finding a convex function

What convex function is minimized A^{-1}?

$$l(P) = \frac{1}{2}\|x - P\nabla f(x) - x^\star\|^2$$

$$x - A^{-1}\nabla f(x) = x^\star \quad \twoheadrightarrow$$
$$z - A^{-1}\nabla f(z) = x^\star \quad \twoheadrightarrow$$

$$\underbrace{x - z}_{s} = A^{-1}(\underbrace{\nabla f(x) - \nabla f(z)}_{y})$$

Define function $l_{x,z}(P) := \mathbb{I}\{Py = s\}$ or $l_{x,z}(P) := \|Py - s\|^2$

- Learning from the (penalized) secant equation
- $P_+ = \arg\min_Q \{l_{x,z}(Q) + \frac{1}{2\eta}\|Q - P\|_F^2\}$ recovers Broyden's quasi-Newton method

Quasi-Newton methods

Initialize: slack off ⛱️ and have some random P_1 and x^1

line-search

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

Get contraction

$$P_{k+1} = \arg\min_{P} \{\mathbb{I}\{Py^k = s^k\} + \|P - P_k\|_F^2\}$$

Repair P_k

$$s_k = x^{k+1} - x^k \qquad y_k = \nabla f(x^{k+1}) - \nabla f(x^k)$$

Quasi-Newton methods

Initialize: slack off 🏖️ and have some random P_1 and x^1

line-search

$$x^{k+1} = x^k - tP_k \nabla f(x^k)$$ Get contraction

$$P_{k+1} = \arg\min_P \{\mathbb{I}\{Py^k = s^k\} + \|P - P_k\|_F^2\}$$ Repair P_k

$$s_k = x^{k+1} - x^k \qquad y_k = \nabla f(x^{k+1}) - \nabla f(x^k)$$

- Additional constraints on P gives various QN variants

Quasi-Newton methods

Initialize: slack off ⛱️ and have some random P_1 and x^1

line-search

$$x^{k+1} = x^k - tP_k \nabla f(x^k)$$ Get contraction

$$P_{k+1} = \arg\min_P \{\mathbb{I}\{Py^k = s^k\} + \|P - P_k\|_F^2\}$$ Repair P_k

$$s_k = x^{k+1} - x^k \qquad y_k = \nabla f(x^{k+1}) - \nabla f(x^k)$$

- Additional constraints on P gives various QN variants
- The distance $\|P - P_k\|_F^2$ can be further generalized

Distance ⇒ divergence

Euclidean $\|P - A^{-1}\|_F^2$ generalizes to Bregman divergences

$$V_d(P, Q) = d(P) - d(Q) - \langle \nabla d(Q), P - Q \rangle$$

- Frobenius norm $d(P) = \frac{1}{2}\|P\|_F^2$
- $\log\det$ divergence $d(P) = -\log\det P$

DFP (BFGS) takes $\mathbb{I}\{Py = s\}$ and $\log\det$ divergence

$$P_{k+1} = \arg\min_{P \succ 0} V_d(P, P_k) \quad \text{subject to} \quad Py^k = s^k$$

Distance ⇒ divergence

Euclidean $\|P - A^{-1}\|_F^2$ generalizes to Bregman divergences

$$V_d(P, Q) = d(P) - d(Q) - \langle \nabla d(Q), P - Q \rangle$$

- Frobenius norm $d(P) = \frac{1}{2}\|P\|_F^2$
- $\log\det$ divergence $d(P) = -\log\det P$

$$P_{k+1} = P_k - \frac{P_k y_k y_k^\top P_k}{y_k^\top P_k y_k} + \frac{s_k s_k^\top}{y_k^\top s_k}.$$

DFP (BFGS) takes $\mathbb{I}\{Py = s\}$ and $\log\det$ divergence

$$P_{k+1} = \arg\min_{P \succ 0} V_d(P, P_k) \quad \text{subject to} \quad Py^k = s^k$$

History of quasi-Newton methods



Davidon



Powell



Broyden, Fletcher, Goldfarb, Shanno





Nocedal, Byrd

~1960 SR1

1970 PSB

1980 L-BFGS 👍

2026 in class

1959 DFP

1970 BFGS 👍

Summary

Summary

- Amortization and learning in quasi-Newton

Summary

- Amortization and learning in quasi-Newton
- Superlinear convergence

Summary

- Amortization and learning in quasi-Newton
- Superlinear convergence

Next lecture: Interior point method

Convergence analysis through potential function

The algorithm admits an either-or intuition

$$V_{k+1} \le V_k - \frac{1}{\kappa} g_k$$

$$\|\nabla f(x^{k+1})\|^2 = g_k \|\nabla f(x^k)\|^2$$

😊 Either g_k is already small, and the contraction is nice

😊 Or g_k is large, but we manage to repair P_k

Define the potential function $\varphi(x, P) = \log \|f(x)\|^2 + \kappa \|I - AP\|_F^2$. Then

$$\varphi(x^{k+1}, P_{k+1}) - \varphi(x^k, P_k) \le \log g_k - g_k \le -1$$

and #iterations to ε-accuracy is $\kappa \|I - AP\|_F^2 + \log(\frac{1}{\varepsilon})$

Another convex function

What convex function is minimized by A^{-1}?

$$x - A^{-1}\nabla f(x) = x^\star \quad \Rightarrow \quad A^{-1} \in \arg\min_P \; f(x - P\nabla f(x))$$

Let $u_x(P) := f(x - P\nabla f(x))$

- $\nabla_P u_x(P) = -\nabla f(x - P\nabla f(x))\nabla f(x)^\top$ is $L\|\nabla f(x)\|^2$-Lipschitz
- GD $P_+ = P - \eta \nabla_P u_x(P)$ with $\eta = \frac{1}{L\|\nabla f(x)\|^2}$ gives

$$x^{k+1} = x^k - P_k \nabla f(x^k)$$
$$P_{k+1} = P_k - \eta \nabla_P u_{x^k}(P_k)$$

$$\|P_+ - A^{-1}\|_F^2 \le \|P - A^{-1}\|_F^2 - \frac{1}{L^2} g \qquad \Rightarrow \qquad \left(\frac{L^2 V_1}{K}\right)^K \text{ convergence}$$