Smoothing for non-smooth optimization, lecture 4

Last time: *Smoothing techniques*

- A "fixed" smoothing approach
- Excessive gap technique

Today: *Applications*

- Game theory: computation of Nash equilibrium
- Sparse principal component analysis

Recap

Consider

$$\min_{x\in Q_1}\max_{y\in Q_2}\{\langle c,x\rangle-\langle b,y\rangle+\langle Ax,y\rangle\}=\max_{y\in Q_2}\min_{x\in Q_1}\{\langle c,x\rangle-\langle b,y\rangle+\langle Ax,y\rangle\}.$$

Write these problems as

$$\min\{f(x):x\in Q_1\}=\max\{\phi(y):y\in Q_2\}$$

for

$$f(x)=\langle c,x\rangle+\max_{y\in Q_2}\{\langle Ax,y\rangle-\langle b,y\rangle\}$$

and

$$\phi(y)=-\langle b,y\rangle+\min_{x\in Q_1}\{\langle c,x\rangle+\langle Ax,y\rangle\}.$$

Smoothing techniques

- Assume d_i is a prox-function for Q_i with modulus ρ_i and max value D_i.
- In N iterations get $x^N\in Q_1$, $y^N\in Q_2$ such that

$$0\le f(\mathbf{x}^N)-\phi(\mathbf{y}^N)\le\frac{4\|A\|}{N+1}\sqrt{\frac{D_1D_2}{\rho_1\rho_2}}$$

- Each iteration requires elementary operations and the solution of three problems of the form

$$\min\{d_i(z)-\langle g,z\rangle:z\in Q_i\}$$

- All of this holds for any choice of norms in E_i, not necessarily the Euclidean norm.

Matrix games

Games in strategic form:

- Each player has a finite set of pure strategies
- A simultaneous choice of strategies determines the payoff of each player

Equilibrium: choice of strategies for each player so that no player wishes to deviate

Theorem 1 (Nash, 1950) *Under suitable circumstances such an equilibrium exists (may involve randomization).*

Consider a two-person, zero-sum game:

- $A \in \mathbb{R}^{m\times n}$: Player 2's payoff matrix
- $x \in \Delta_n$: set of mixed strategies of Player 1.
- $y \in \Delta_m$: set of mixed strategies of Player 2.

Player 1's problem: $\min\limits_{x\in\Delta_n} \max\limits_{j} (Ax)_j$

Player 2's problem: $\max\limits_{y\in\Delta_m} \min\limits_{i} (A^\top y)_i$

Nash equilibrium problem:

$$\min_{x\in\Delta_n}\max_{y\in\Delta_m}\langle Ax, y\rangle = \max_{y\in\Delta_m}\min_{x\in\Delta_n}\langle Ax, y\rangle.$$

To apply smoothing techniques need prox-functions for $Q_1 = \Delta_n, Q_2 = \Delta_m$.

Euclidean distance:

$$d_1(x) := \frac{1}{2}\sum_{i=1}^{n}\left(x_i - \frac{1}{n}\right)^2,\ \ d_2(x) := \frac{1}{2}\sum_{j=1}^{m}\left(y_j - \frac{1}{m}\right)^2.$$

It is easy to see that

$$D_1 = \frac{1}{2} - \frac{1}{2n} \le \frac{1}{2},\ D_2 = \frac{1}{2} - \frac{1}{m} \le \frac{1}{2}.$$

Also, for the Euclidean norms in $\mathbb{R}^n, \mathbb{R}^m$ we have

$$\rho_1 = \rho_2 = 1,$$

and

$$\|A\| = \sigma_{\max}(A) = \sqrt{\lambda_{\max}(A^\top A)}.$$

So in N iterations get $x^N \in \Delta_n,\ y^N \in \Delta_m$ such that

$$0 \le f(\mathbf{x}^N) - \phi(\mathbf{y}^N) \le \frac{\sqrt{\lambda_{\max}(A^\top A)}}{N+1}$$

Entropy:

$$d_1(x) := \sum_{i=1}^{n} x_i \log x_i + \log n,\ \ d_2(x) := \sum_{j=1}^{m} y_j \log y_j + \log m.$$

It is easy to see that

$$D_1 = \log n,\ D_2 = \log m.$$

Also, for the 1-norms in $\mathbb{R}^n, \mathbb{R}^m$ we have

$$\rho_1 = \rho_2 = 1,$$

and

$$\|A\| = \max\left\{|A|_{ij}\right\}.$$

$|A_{ij}|$

So in N iterations get $x^N \in \Delta_n,\ y^N \in \Delta_m$ such that

$$0 \le f(\mathbf{x}^N) - \phi(\mathbf{y}^N) \le \frac{4\sqrt{\log n \log m}\,\max|A_{ij}|}{N+1}$$

What about the subproblems at each iteration? Need to solve

$$\min \{d_1(x) - \langle g, x\rangle : x \in \Delta_n\}.$$

For the entropy $d_1(x) = \sum_{i=1}^{n} x_i \log x_i + \log n$, get a closed-form solution

$$x_i = \frac{e^{g_i}}{\sum_{k=1}^{n} e^{g_k}},\ i = 1, \ldots, n$$

For the Euclidean distance $d_1(x) = \frac{1}{2}\sum_{i=1}^{n}(x_i - 1/n)^2$, it follows from the KKT conditions that the solution is

$$x_i = (g_i - \lambda)^+,\ i = 1, \ldots, n$$

for $\lambda \in \mathbb{R}$ such that $\sum_{i=1}^{n}(g_i - \lambda)^+ = 1$.

Can find λ by sorting $g_i,\ i = 1, \ldots, n$.

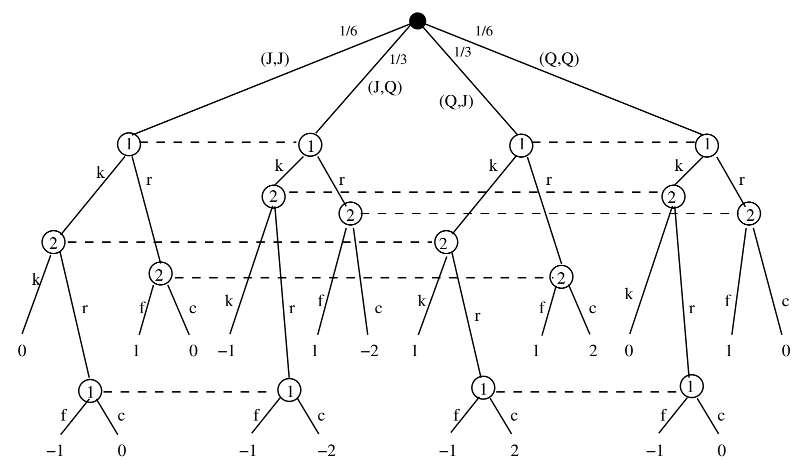
Sequential games

Games that involve turn-taking, chance moves, and imperfect information.

Example 2 (Simplified poker)

- Opening: players bet 1 each
- One card is dealt to each player
- Player 1 can check or raise
  - If Player 1 checks then Player 2 can check or raise
  - If Player 2 checks there is a showdown (higher wins)
  - If Player 2 raises then Player 1 can fold, or call (showdown)
- If Player 1 raises then Player 2 can fold, or call (showdown)

Game tree representation



The sequence form

- With perfect recall, can formulate the Nash equilibrium problem in terms of sequences of moves.
- Strategies $\leftrightarrow$ set of realization plans

Example 3 (simplified poker)
Player 1's sequences:

$$S = \left\{\emptyset, k^J, r^J, k^Q, r^Q, k^J f^J, k^J c^J, k^Q f^Q, k^Q c^Q\right\}$$

Set of realization plans: $\{x : Ex = e,\ x \geq 0\}$, for

$$E = \begin{bmatrix} 1 & & & & & & & & \\ -1 & 1 & 1 & & & & & & \\ -1 & & & 1 & 1 & & & & \\ & -1 & & & & 1 & 1 & & \\ & & & -1 & & & & 1 & 1 \end{bmatrix}, \quad e = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$x = (1, \frac{1}{3}, \frac{2}{3}, \frac{1}{2}, \frac{1}{2}, \frac{1}{6}, \frac{1}{6}, \frac{1}{8}, \frac{3}{8})$

Nash equilibrium via sequence form

Assume

- Q_1, Q_2: realization plans of Players 1 and 2
- A: Player 2's payoff matrix

Nash equilibrium

$$\min_{x \in Q_1} \max_{y \in Q_2} \langle Ax, y\rangle = \max_{y \in Q_2} \min_{x \in Q_1} \langle Ax, y\rangle.$$

- In matrix games Q_1, Q_2 are simplices
- In sequential games Q_1, Q_2 are complexes

Texas Hold'em Poker (with limits):

Game tree has $\sim 10^{18}$ nodes.

- **Gilpin and Sandholm (2005–):**
  Can approximate by solving abstractions (simpler sequential games)
- The closer the abstraction, the better

Rhode Island Hold'em: created for AI research

- Each player pays an ante of 5 chips.
- Each player is dealt a single card, placed face down.
- First betting round: Each player may check, or bet if no bets have been placed. If a bet has been placed, then the player may fold, call, or raise. The players are limited to 3 raises per betting round. In this betting round, the bets are 10 chips.
- A community card is dealt face up and a second betting round take place with bets equal to 20 chips.
- Another community card is dealt face up and a final betting round takes place at this point, with bets equal to 20 chips.

If neither player folds, then the showdown takes place. Both players turn over their cards. The player who has the best 3-card poker hand takes the pot. In the event of a draw, the pot is split evenly. The ranking of hands is given below.

Hands

- Straight flush: e.g., J, Q, K of spades
- Three of a kind: e.g., 8, 8, 8 of spades, hearts, diamonds
- Straight: e.g., J, Q, K
- Flush: e.g., 2, 5, 7 diamonds
- Pair: e.g., 2, 2, 8
- High card

Hoda, Gilpin, Sandholm, P. (2006–):
Apply smoothing techniques to solve large sequential games.

Theorem 4 (HGP 2006) *Any prox-function for the simplex yields a prox-function for any complex.*

Remarks

- Provide estimates of relevant ρ, D
- Subproblem (for Q complex):

$$\min \{d(z) - \langle g, z \rangle : z \in Q\}.$$

  can be recursively solved via solving subproblems over simplices
- Most expensive work per iteration: matrix-vector products

$$x \mapsto Ax, \; y \mapsto A^\top y$$

Complexity results

From Smoothing and HGP Theorem get:

Theorem 5

- *For the entropy induced prox-function:*

  $\left\lceil (4G^2/\epsilon) \max |A_{ij}| \right\rceil$ *itns* $\rightsquigarrow (\bar{x}, \bar{y}) \in Q_1 \times Q_2$ *such that*

$$\max_{y \in Q_2} \langle Ay, \bar{x} \rangle - \min_{x \in Q_1} \langle A\bar{y}, x \rangle \leq \epsilon$$

  G*: size of the game tree*
- *For the Euclidean induced prox-function:*

  $\left\lceil (4G/\epsilon) \lambda_{\max}^{1/2}(A^\top A) \right\rceil$ *itns* $\rightsquigarrow (\bar{x}, \bar{y}) \in Q_1 \times Q_2$ *such that*

$$\max_{y \in Q_2} \langle Ay, \bar{x} \rangle - \min_{x \in Q_1} \langle A\bar{y}, x \rangle \leq \epsilon$$

Computational experience

Test problems, size of A

- Rhode Island Hold'em poker, $1M \times 1M$.
- Abstractions of Texas Hold'em poker:

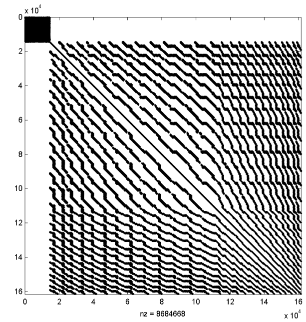
  $81 \times 81, 1041 \times 1041,$
  $10421 \times 10421, 160k \times 160k,$
  $13M \times 13M, 100M \times 100M$

Efficient matrix representation

- Payoff matrix in poker games admits a concise representation. For example, for a three-round game
$$A = \begin{bmatrix} A_1 & & \\ & A_2 & \\ & & A_3 \end{bmatrix}$$
where $A_i = F_i \otimes B_i,\ i = 1, 2$ and $A_3 = F_3 \otimes B_3 + S \otimes W$ for smaller matrices F_i, B_i, S, W.
- Do not need to form A explicitly.
- Instead have subroutines that compute $x \mapsto A^\mathsf{T} x,\ y \mapsto Ay$.

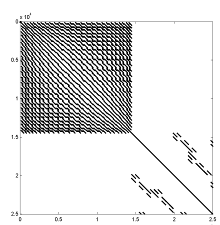
About the $160K \times 160K$ instance

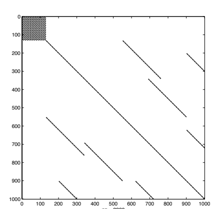
Matrix A



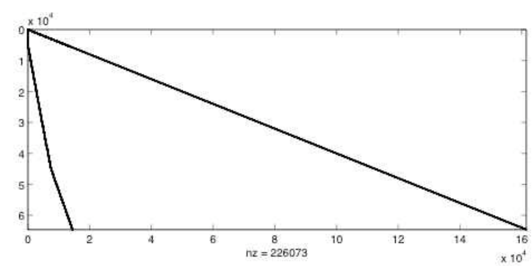
nnz = 8684668

More about the $160k \times 160k$ **problem** $25k \times 25k$ and $1k \times 1k$ upper-left blocks of A



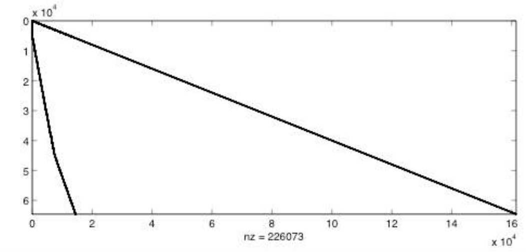


Matrix E



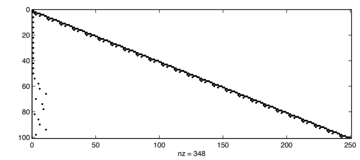
nnz = 226073

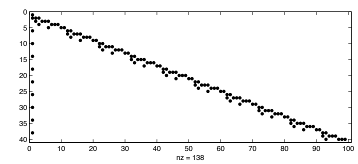
Matrix F



nnz = 226073

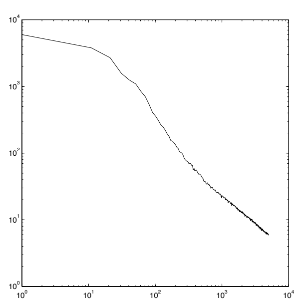
Upper-left blocks of E





Path of the iterates' gap

$$\max_{y \in Q_2} \langle y, Ax^k \rangle - \min_{x \in Q_1} \langle y^k, Ax \rangle$$



Largest instance attempted so far:

$$\begin{array}{rl} A: & 101,192,201 \times 101,192,221 \\ E: & 40,476,881 \times 101,192,201 \\ F: & 40,476,881 \times 101,192,221 \end{array}$$

number of non-zeros in A: $2,927,336,725,318$

Implementation

- Based on EGT technique
- Machine: 1.65GHz IBM eServer p5 570 with 64 gigabytes of RAM
- Concise representation requires only 2.49 GB of RAM.
- Entire algorithm uses about 30 GB of RAM.
- Each iteration takes a few hours (it has run for months)

Poker players

Poker is a central challenge problem in AI. Some reasons:

- Imperfect information: the other players' cards are hidden, future events
- Bluffing and other deceptive strategies are needed in a good player
- Interest in developing automatic poker players

Gilpin, Sandholm, Sorensen 2007

- A poker player based on the four-round abstraction.
- Use the approximate equilibrium found by our algorithm.

Principal component analysis

Suppose $C \in \mathbb{S}^n$ is a covariance matrix. Then there exist $P \in \mathbb{R}^{n \times n}$ orthogonal such that

$$C = Q\text{Diag}(\lambda(C))Q^\top = \sum_{i=1}^{n} \lambda_i(C) p_i p_i^\top$$

Principal components: $p_1, \ldots, p_n$

Can find first principal component by solving:

$$\begin{array}{ll} \max & x^\top C x \\ \text{s.t.} & x^\top x = 1 \end{array}$$

$X := xx^\top$

Semidefinite programming (SDP) reformulation:
Put $X := xx^\top$, get

$$\begin{array}{ll} \max & C \bullet X \\ \text{s.t.} & I \bullet X = 1 \\ & X \succeq 0. \end{array}$$

$= C \cdot xx^\top = x^\top C x$

$I \cdot X = X_{ii} = x_i^2$

Sparse first principal component:

$$\begin{array}{ll} \max & x^\top C x \\ \text{s.t.} & x^\top x = 1 \\ & \text{card}(x) \le k \end{array}$$

SDP relaxation

$$\begin{array}{ll} \max & C \bullet X - \delta \mathbf{1} \bullet |X| \\ \text{s.t.} & I \bullet X = 1 \\ & X \succeq 0. \end{array}$$

For simplicity assume $\delta = 1$.

Then the SDP

$$\begin{array}{ll} \max & C \bullet X - \mathbf{1} \bullet |X| \\ \text{s.t.} & I \bullet X = 1 \\ & X \succeq 0 \end{array}$$

can be written as

$$\max_{X \in Q_1} \min_{Y \in Q_2} \{\langle C, X\rangle - \langle X, Y\rangle\}$$

where

$$Q_1 = \{X \in \mathbf{S}^n : X \succeq 0, \langle I, X\rangle = 1\}, \quad Q_2 = \left\{Y \in \mathbf{S}^n : |Y_{ij}| \le 1\right\},$$

and $\langle \cdot, \cdot \rangle$ is the trace inner product: $\langle X, S\rangle = \text{trace}(XS) = X \bullet S$.

Prox-function for Q_1:

$$d_1(X) = \sum_{i=1}^n \lambda_i(X) \log \lambda_i(X) + \log n$$

Prox-function for Q_2:

$$d_2(Y) = \frac{1}{2}\langle Y, Y\rangle$$

It is easy to see that $D_1 = \log n,\ D_2 = n^2/2$.

For suitable norms we get $\rho_1 = \rho_2 = 1$ and $\|A\| = 1$.

Thus in N iterations get $\bar{X} \in Q_1$ such that $C \bullet \bar{X} - \mathbf{1} \bullet |\bar{X}|$ is within

$$\frac{2\sqrt{2}n\sqrt{\log n}}{N+1}$$

of the optimal SDP value.

Subproblems at each iteration:

- $\min \left\{ \sum_{i=1}^n \lambda_i(X) \log \lambda_i(X) + \log n - G \bullet X : X \succeq 0,\ I \bullet X = 1 \right\}$
- $\min \left\{ \frac{1}{2} U \bullet U - \langle G, U\rangle : |U_{ij}| \le 1 \right\}$

Solution to second one:

$$U_{ij} = \text{sign}(G_{ij}) \min\left\{|G_{ij}|, 1\right\}.$$

Solution to the first one (similar to entropy):

- Compute eigenvalue decomposition: $G = V\text{Diag}(\lambda(G))V^\mathsf{T}$
- Let $h_i := \frac{e^{\lambda_i(G)}}{\sum_{k=1}^n e^{\lambda_k(G)}},\ i = 1, \ldots, n$
- Let $X = V\text{Diag}(h)V^\mathsf{T}$

References for today's material

- S. Hoda, A. Gilpin, J. Peña, and T. Sandholm, "A gradient-based algorithm for finding Nash equilibria in extensive form games," In preparation.
- A. D'Aspremont, L. El Ghaoui, M. Jordan, and G. Lankcriet, "A direct formulation for sparse PCA using semidefinite programming," To appear in SIAM Review.