Matrix Approximations - II

Sanjiv Kumar, Google Research, NY

EECS-6898, Columbia University - Fall, 2010

Sampling Based Methods

So Far…

- Methods that primarily depend on matrix-vector products
- Not suitable when input matrix is large and dense
  - Kernel Matrix $n = 20M$ ⇨ 1600 TB 200,000, 8GB machines!!
- Matrices may be so big that storage becomes a big problem
- One may want to reduce the computational cost significantly

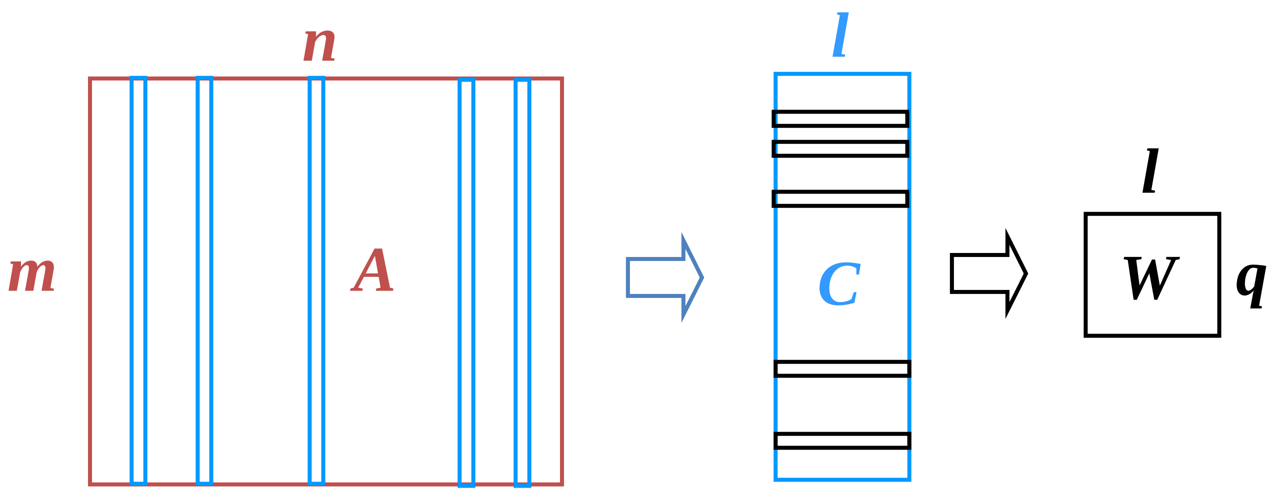
Sampling-Based Methods

- Sample a few columns or rows or both according to some distribution (without replacement in practice)
- Approximate the desired quantity by manipulating just the sampled vectors
  - No need to even create the entire matrix !!
- If done carefully, the error in approximation can be bounded

Can approximate multiplication, low-rank matrix, singular values, singular vectors

Sampling Based Methods

Sample l columns/rows randomly



Main Issues

- How to sample columns and rows?
  - Uniformly?
  - From a fixed non-uniform distribution e.g., column/row norm?
  - From adaptive distribution: distribution changes after picking a sample subset
- What is the algorithm and how much error are we making?

Overview

1. Approximate Matrix Multiplication
   – Sample columns of one matrix and rows from the other
2. Column-sampling methods for spectral decomposition
   – Methods that use decomposition of entire sampled columns
   – Methods that further sample the rows from the sampled columns
3. Low-rank approximation
   – Spectral reconstructions $A_k = U_k \Sigma_k V_k^T$
   – Matrix Projection $A_k = U_k U_k^T A$
4. Sampling Techniques
5. Ensemble Methods
   – How to combine multiple approximations to yield more accurate one

Sampling Based Methods: Matrix Multiplication

We want to approximate

$$\underset{m\times n}{A}\,\underset{n\times p}{B} \approx \underset{m\times l}{C}\,\underset{l\times p}{R} \qquad l << n$$

Sampling Based Methods: Matrix Multiplication

We want to approximate

$$\underset{m \times n}{A}\,\underset{n \times p}{B} \approx \underset{m \times l}{C}\,\underset{l \times p}{R} \qquad l << n$$

Basic Idea

1. Sample l columns from A and form a submatrix C
2. Pick the corresponding rows from B and form a submatrix R
3. Scale the submatrices appropriately
4. Output the multiplication of two scaled submatrices

Sampling Based Methods: Matrix Multiplication

We want to approximate

$$\underset{m\times n}{A}\,\underset{n\times p}{B} \approx \underset{m\times l}{C}\,\underset{l\times p}{R} \qquad l << n$$

Algorithm

Given $A,\ B,\ 1 \le l \le n,\ \{p_i\}_{i=1}^{n}$ s.t. $\sum_i p_i = 1,\ p_i \ge 0$

fixed non-uniform distribution

Sampling Based Methods: Matrix Multiplication

We want to approximate

$$\underset{m\times n}{A}\underset{n\times p}{B} \approx \underset{m\times l}{C}\underset{l\times p}{R} \qquad l << n$$

Algorithm

Given $A,\ B,\ 1 \le l \le n,\ \{p_i\}_{i=1}^{n}$ s.t. $\sum_i p_i = 1,\ \ p_i \ge 0$

fixed non-uniform distribution

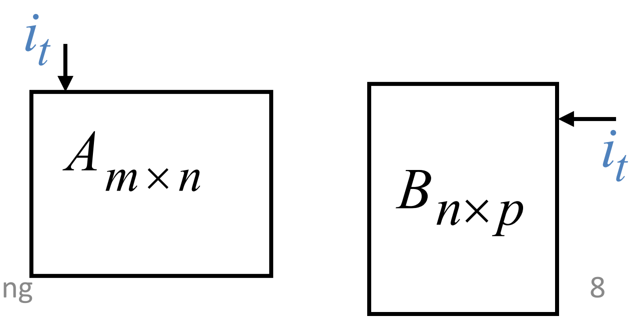
For $t = 1, \ldots, l$

- Pick $i_t \in \{1, \ldots, n\}$ with $P(i_t = k) = p_k$ independently, with replacement

  in practice, without replacement !

- Set $\underbrace{C^{(t)}}_{\text{column}} = A^{(i_t)} / \sqrt{l\, p_{i_t}}$ $\qquad \underbrace{R_{(t)}}_{\text{row}} = B_{(i_t)} / \sqrt{l\, p_{i_t}}$

Return $C,\ R$



Sampling Based Methods: Matrix Multiplication

Standard proof strategy (based on concentration of measures)

Show

$$E[\|AB - CR\|_F^2] = \sum_{i=1}^{n} \sum_{j=1}^{p} E[(AB - CR)_{ij}^2] = \text{small quantity}$$

$Var[\|AB - CR\|_F^2]$ is small for right choice of p_k

Sampling Based Methods: Matrix Multiplication

We want to approximate $AB = \sum_{t=1}^{n} A^{(t)} B_{(t)}$

$$CR = \sum_{t=1}^{l} C^{(t)} R_{(t)} = \sum_{t=1}^{l} \frac{1}{(l\, p_{i_t})} A^{(i_t)} B_{(i_t)}$$

Why is it a good approximation?

Sampling Based Methods: Matrix Multiplication

We want to approximate $AB = \sum_{t=1}^{n} A^{(t)} B_{(t)}$

$$CR = \sum_{t=1}^{l} C^{(t)} R_{(t)} = \sum_{t=1}^{l} \frac{1}{(l\, p_{i_t})} A^{(i_t)} B_{(i_t)}$$

Why is it a good approximation?

Expectation $E[(CR)_{ij}] = (AB)_{ij}$

Variance $Var[(CR)_{ij}] = \frac{1}{l} \sum_{k=1}^{n} \frac{A_{ik}^2 B_{kj}^2}{p_k} - \frac{1}{l} (AB)_{ij}^2$

Sampling Based Methods: Matrix Multiplication

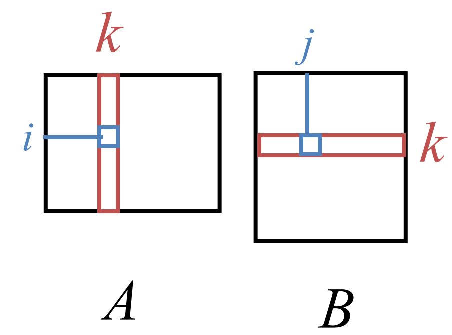
We want to approximate $AB = \sum_{t=1}^{n} A^{(t)} B_{(t)}$

$$CR = \sum_{t=1}^{l} C^{(t)} R_{(t)} = \sum_{t=1}^{l} \frac{1}{(l\, p_{i_t})} A^{(i_t)} B_{(i_t)}$$

Why is it a good approximation?

Expectation $\quad E[(CR)_{ij}] = (AB)_{ij}$

Variance $\quad Var[(CR)_{ij}] = \frac{1}{l} \sum_{k=1}^{n} \frac{A_{ik}^2 B_{kj}^2}{p_k} - \frac{1}{l}(AB)_{ij}^2$



Proof: Let $X_t = \left(\frac{A^{(i_t)} B_{(i_t)}}{(l\, p_{i_t})} \right)_{ij}$

$$E[X_t] = \sum_{k=1}^{n} p_k \frac{A_{ik} B_{kj}}{(l\, p_k)} = \frac{1}{l}(AB)_{ij}$$

$$E[X_t^2] = \sum_{k=1}^{n} \frac{A_{ik}^2 B_{kj}^2}{(l^2\, p_k)}$$

Sampling Based Methods: Matrix Multiplication

We want to approximate $AB = \sum_{t=1}^{n} A^{(t)} B_{(t)}$

$$CR = \sum_{t=1}^{l} C^{(t)} R_{(t)} = \sum_{t=1}^{l} \frac{1}{(l\, p_{i_t})} A^{(i_t)} B_{(i_t)}$$

Why is it a good approximation?

Expectation $E[(CR)_{ij}] = (AB)_{ij}$

Variance $Var[(CR)_{ij}] = \frac{1}{l} \sum_{k=1}^{n} \frac{A_{ik}^2 B_{kj}^2}{p_k} - \frac{1}{l}(AB)_{ij}^2$

Proof: Let $X_t = \left(\frac{A^{(i_t)} B_{(i_t)}}{(l\, p_{i_t})} \right)_{ij}$

$$E[X_t] = \sum_{k=1}^{n} p_k \frac{A_{ik} B_{kj}}{(l\, p_k)} = \frac{1}{l}(AB)_{ij}$$

$$E[X_t^2] = \sum_{k=1}^{n} \frac{A_{ik}^2 B_{kj}^2}{(l^2\, p_k)}$$

$$\boxed{(CR)_{ij} = \sum_{t=1}^{l} X_t}$$

$$Var(X_t) = E[X_t^2] - E[X_t]^2$$

Next, get

$E[\|AB - CR\|_F^2]$

Sampling Based Methods: Matrix Multiplication

We want to find

$$E[\|AB - CR\|_F^2] = \sum_{i=1}^{n} \sum_{j=1}^{p} E[(AB - CR)_{ij}^2]$$

Sampling Based Methods: Matrix Multiplication

We want to find

$$E[\|AB - CR\|_F^2] = \sum_{i=1}^{n}\sum_{j=1}^{p} E[(AB - CR)_{ij}^2]$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{p} Var[(CR)_{ij}]$$

$$Var[(CR)_{ij}] = \frac{1}{l}\sum_{k=1}^{n}\frac{A_{ik}^2 B_{kj}^2}{p_k} - \frac{1}{l}(AB)_{ij}^2$$

$$= \frac{1}{l}\sum_{k=1}^{n}\frac{1}{p_k}\left|A^{(k)}\right|^2\left|B_{(k)}\right|^2 - \frac{1}{l}\|AB\|_F^2$$

Sampling Based Methods: Matrix Multiplication

We want to find

$$E[\|AB - CR\|_F^2] = \sum_{i=1}^{n}\sum_{j=1}^{p} E[(AB - CR)_{ij}^2]$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{p} Var[(CR)_{ij}]$$

$$Var[(CR)_{ij}] = \frac{1}{l}\sum_{k=1}^{n}\frac{A_{ik}^2 B_{kj}^2}{p_k} - \frac{1}{l}(AB)_{ij}^2$$

$$= \frac{1}{l}\sum_{k=1}^{n}\frac{1}{p_k}\left|A^{(k)}\right|^2\left|B_{(k)}\right|^2 - \frac{1}{l}\|AB\|_F^2$$

Find p_k that minimizes above $\quad p_k = \left|A^{(k)}\right|\left|B_{(k)}\right| \Big/ \sum_{k'=1}^{n}\left|A^{(k)}\right|\left|B_{(k)}\right|$

with positivity and unit sum constraints

$$\boxed{= \frac{1}{l}\left(\sum_{k=1}^{n}\left|A^{(k)}\right|\left|B_{(k)}\right|\right)^2 - \frac{1}{l}\|AB\|_F^2}$$

$$\leq \frac{1}{l}\|A\|_F^2\|B\|_F^2$$

Theoretical Guarantee

Given $A,\ B,\ 1 \le l \le n,\ \{p_i\}_{i=1}^{n}$ s.t. $\sum_i p_i = 1,\ p_i \ge 0$ and $\beta \le 1$

if $\quad p_k \ge \beta \left| A^{(k)} \right| \left| B_{(k)} \right| \Big/ \sum_{k'=1}^{n} \left| A^{(k)} \right| \left| B_{(k)} \right|$

then $\quad \boxed{E[\|AB - CR\|_F^2] \le (1/\beta l)\|A\|_F^2 \|B\|_F^2}$

Let $\delta \in (0,1)$ and $\eta = 1 + \sqrt{(8/\beta)\log(1/\delta)}$, then with probability at least $1-\delta$

$$\boxed{\|AB - CR\|_F^2 \le (\eta^2 / \beta l)\|A\|_F^2 \|B\|_F^2}$$

Proof based on showing that changing one column/row does not change the product CR by much, and then applying concentration of measures: either Doob Martingale or Mcdiarmid’s inequality

Implementation Details

- How to sample?

Uniform Random: just one pass over A and B

Data-dependent sampling: based on column/row norms of A and B

- Two passes necessary
- First pass: compute and store $\left|A^{(k)}\right|$ and $\left|B_{(k)}\right|$ $\quad k = 1,\ldots,n$
- Second pass: sample from A and B with $p_k = \alpha\left|A^{(k)}\right|\left|B_{(k)}\right|$

Implementation Details

- How to sample?

  Uniform Random: just one pass over A and B

  Data-dependent sampling: based on column/row norms of A and B

  - Two passes necessary
  - First pass: compute and store $\left|A^{(k)}\right|$ and $\left|B_{(k)}\right|$ $k = 1,\ldots,n$
  - Second pass: sample from A and B with $p_k = \alpha\left|A^{(k)}\right|\left|B_{(k)}\right|$

- Special case $B = A^T$

$$AA^T \approx CC^T \qquad p_k = \left|A^{(k)}\right|^2 \Big/ \|A\|_F^2$$

$$\boxed{E[\left\|AA^T - CC^T\right\|_F] \le (1/\sqrt{\beta l})\|A\|_F^2}$$

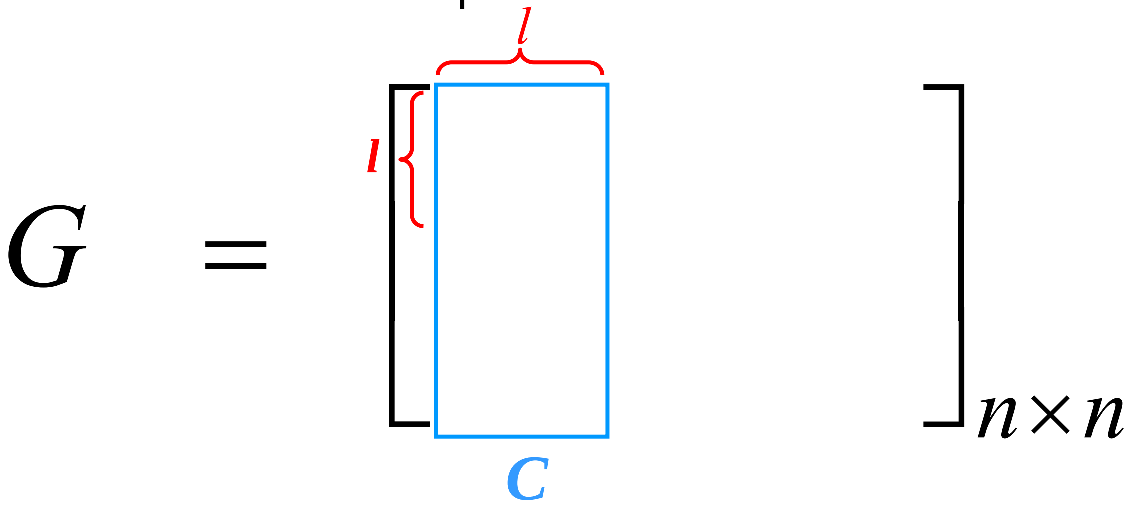
Overview

1. Approximate Matrix Multiplication
   - Sample columns of one matrix and rows from the other
2. Column-sampling methods for spectral decomposition
   - Methods that use decomposition of entire sampled columns
   - Methods that further sample the rows from the sampled columns
3. Low-rank approximation
   - Spectral reconstructions $A_k = U_k \Sigma_k V_k{}^T$
   - Matrix Projection $A_k = U_k U_k{}^T A$
4. Sampling Techniques
5. Ensemble Methods
   - How to combine multiple approximations to yield more accurate one

Approximate Spectral Decomposition

Let’s focus on decomposition of a symmetric matrix - arise commonly in machine learning applications (other cases possible)

Sample l columns without replacement



- Column-Sampling Approximation – SVD of C
- Nystrom Approximation – SVD of W

Column-Sampling Approximation

$$\underset{n\times l}{C} = \underset{n\times l}{U_c} \underset{l\times l}{\Sigma_c} \underset{l\times l}{V_c^T} \qquad O(nl^2)$$

Column-Sampling Approximation

Suppose l columns were sampled uniformly

$$C = U_c \Sigma_c V_c^T \qquad O(nl^2)$$

$$\boxed{\tilde{U}_G = U_c = C V_c \Sigma_c^{-1}}$$

$$\boxed{\tilde{\Sigma}_G = \sqrt{\frac{n}{l}} \Sigma_c}$$

Column-Sampling Approximation

Suppose l columns were sampled uniformly

$$C = U_c \Sigma_c V_c^T$$

$O(nl^2)$

$n \sim 20M, l \sim 10K$

$$\boxed{\tilde{U}_G = U_c = C V_c \Sigma_c^{-1}}$$

$$\boxed{\tilde{\Sigma}_G = \sqrt{\frac{n}{l}} \Sigma_c}$$

$$C^T C = V_c \Sigma_c^2 V_c^T$$

$l \times l$

$O(nl^2)$ $O(l^3)$

parallelize

For rank-k, $k \leq l$ reconstruction, pick top singular vectors and/or singular values !

Nystrom Approximation

$$G = \begin{array}{cc} & \overbrace{\qquad}^{l} \\ l\{ & \left[\begin{array}{c|c} \boxed{\begin{matrix} W \\ G_{21} \end{matrix}} & \begin{matrix} G_{21}^T \\ G_{22} \end{matrix} \end{array}\right]_{n \times n} \\ & C \end{array}$$

pseudo-inverse

$$G \approx \tilde{G} = CW^{-1}C^T \quad \text{or} \quad CW^{+}C^T$$

Reconstructs W and G_{21} i.e., C exactly!

Nystrom Approximation

$$G = \begin{array}{c c} & \overbrace{\qquad}^{l} \\ l\{ & \end{array} \begin{bmatrix} W & G_{21}^T \\ G_{21} & G_{22} \end{bmatrix}_{n \times n}$$

C

$$G \approx \tilde{G} = C W^{-1} C^T$$

$$W = U_W \Sigma_W U_W^T \qquad O(l^3)$$

$$\boxed{\tilde{\Sigma}_G = \frac{n}{l} \Sigma_W}$$

Nystrom Approximation

$$G = \begin{array}{cc} & \overbrace{}^{l} \\ l\{ & \end{array} \begin{bmatrix} W & G_{21}^T \\ G_{21} & G_{22} \end{bmatrix}_{n\times n}$$

$$C$$

$$G \approx \tilde{G} = CW^{-1}C^T$$

$$W = U_W \Sigma_W U_W^T \qquad O(l^3)$$

$$\tilde{G} = \tilde{U}_G \tilde{\Sigma}_G \tilde{U}_G^T$$

$$\boxed{\tilde{\Sigma}_G = \frac{n}{l}\Sigma_W}$$

$$\boxed{\tilde{U}_G = \sqrt{\frac{l}{n}} C U_W \Sigma_W^{-1}}$$

Nystrom Approximation

$$G = \begin{array}{cc} & \begin{array}{cc} \overbrace{\qquad\quad}^{l} & \phantom{G_{21}^T} \end{array} \\ l\left\{\right. & \begin{bmatrix} W & G_{21}^T \\ G_{21} & G_{22} \end{bmatrix}_{n\times n} \\ & \begin{array}{cc} C & \phantom{G_{22}} \end{array} \end{array}$$

$$G \approx \tilde{G} = CW^{-1}C^T$$

$$W = U_W \Sigma_W U_W^T$$

For rank-k, $k \le l$ reconstruction, pick top singular vectors and/or singular values !

$$\boxed{\tilde{\Sigma}_G = \frac{n}{l}\Sigma_W}$$

$$\boxed{\tilde{U}_G = \sqrt{\frac{l}{n}}CU_W \Sigma_W^{-1}}$$

Not Orthonormal !

$$\tilde{U}_G^T \tilde{U}_G \neq I$$

Nystrom Vs Column-Sampling

Spectral reconstruction: $\tilde{G} = \tilde{U}_G \tilde{\Sigma}_G \tilde{U}_G^T$

$$\tilde{G}_{nys} = C W^{-1} C^T$$

$$\tilde{G}_{col} = C \left(\left[\frac{l}{n} C^T C \right]^{1/2} \right)^{-1} C^T$$

Nystrom Vs Column-Sampling

Spectral reconstruction: $\tilde{G} = \tilde{U}_G \tilde{\Sigma}_G \tilde{U}_G^T$

$$\tilde{G}_{nys} = CW^{-1}C^T$$

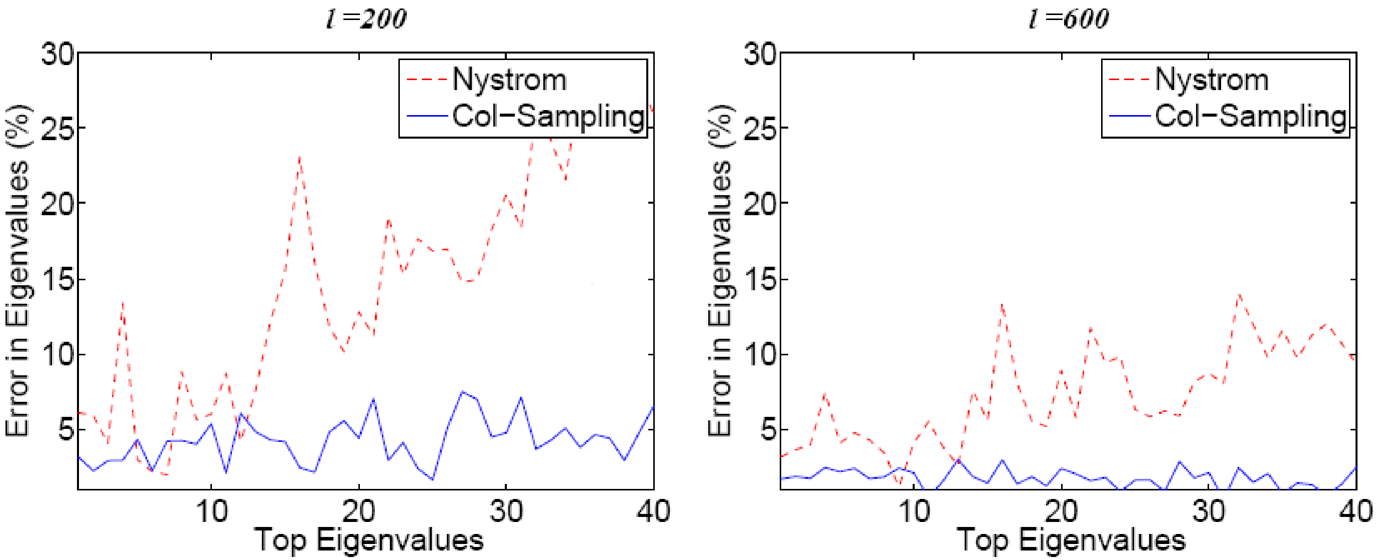
$$\tilde{G}_{col} = C\left(\left[\frac{l}{n}C^T C\right]^{1/2}\right)^{-1} C^T$$

Experimental Comparison

- PIE-7K: 7K face images under different pose/illumination
- Linear kernel: $k(x, y) = x^T y$
- G is a dense 7K x 7K symmetric positive semi-definite matrix
- Eigenvalues, eigenvectors, and low-rank approximations (spectral-reconstruction)

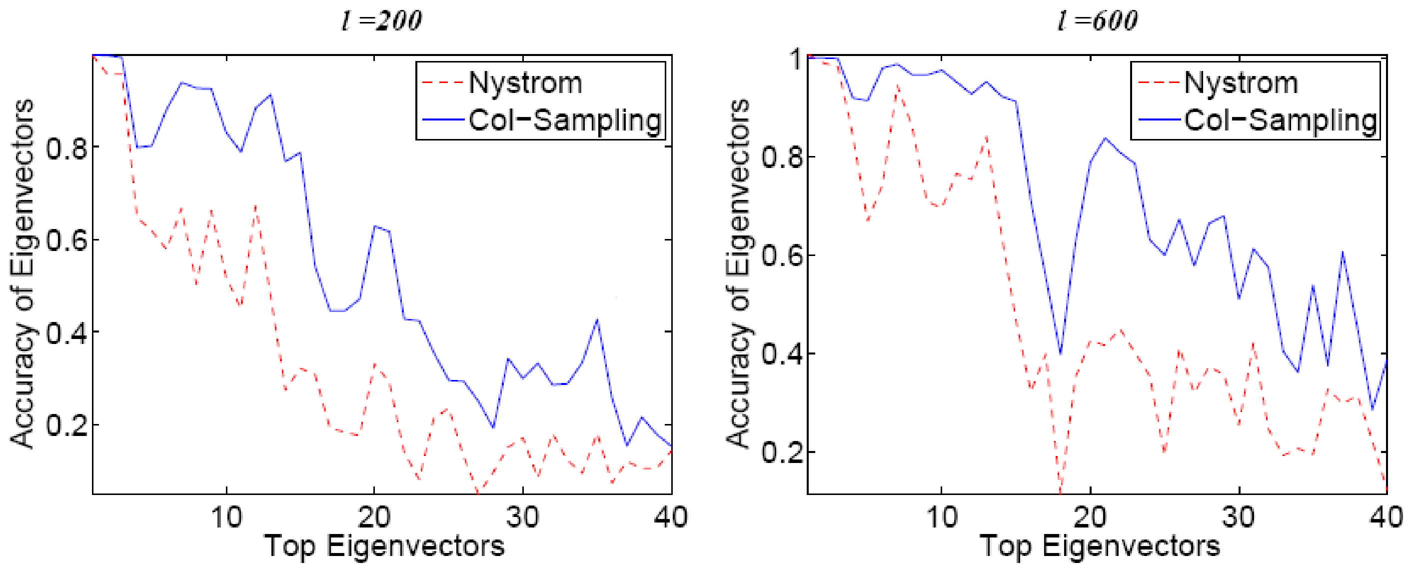
Eigenvalues Comparison

% deviation from exact



Eigenvectors Comparison

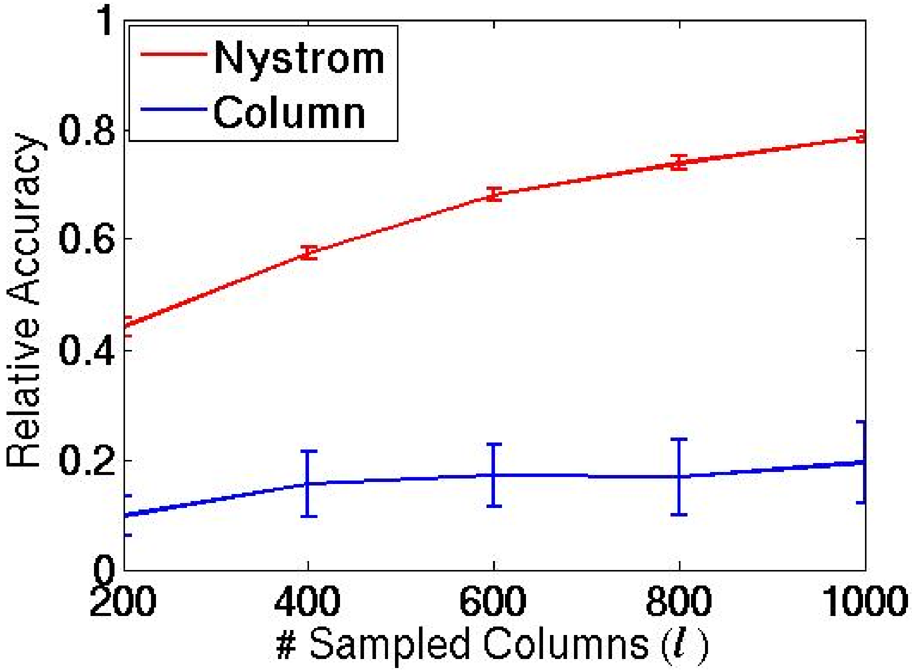
Principal angle with exact



Low-Rank Approximations

Spectral reconstruction: $\tilde{G}_k = \tilde{U}_k \tilde{\Sigma}_k \tilde{U}_k^T$

$$\frac{\left\| G - \hat{G}_k \right\|_F}{\left\| G - \tilde{G}_k \right\|_F}$$



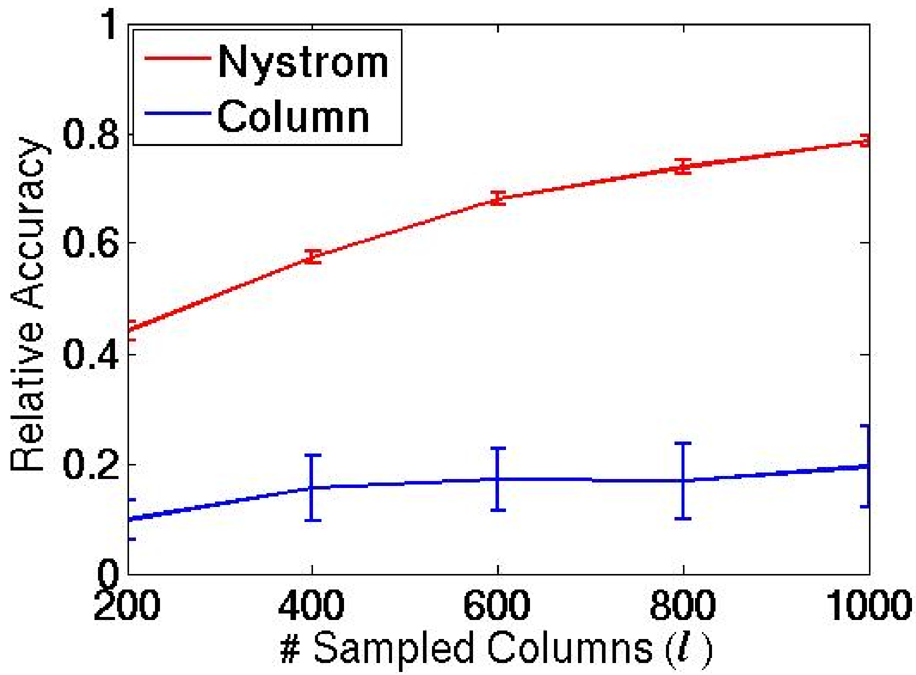
$k = 100$

Nystrom gives better reconstruction than Col-Sampling !

Low-Rank Approximations

Spectral reconstruction: $\tilde{G}_k = \tilde{U}_k \tilde{\Sigma}_k \tilde{U}_k^T$

$$\frac{\|G - \hat{G}_k\|_F}{\|G - \tilde{G}_k\|_F}$$



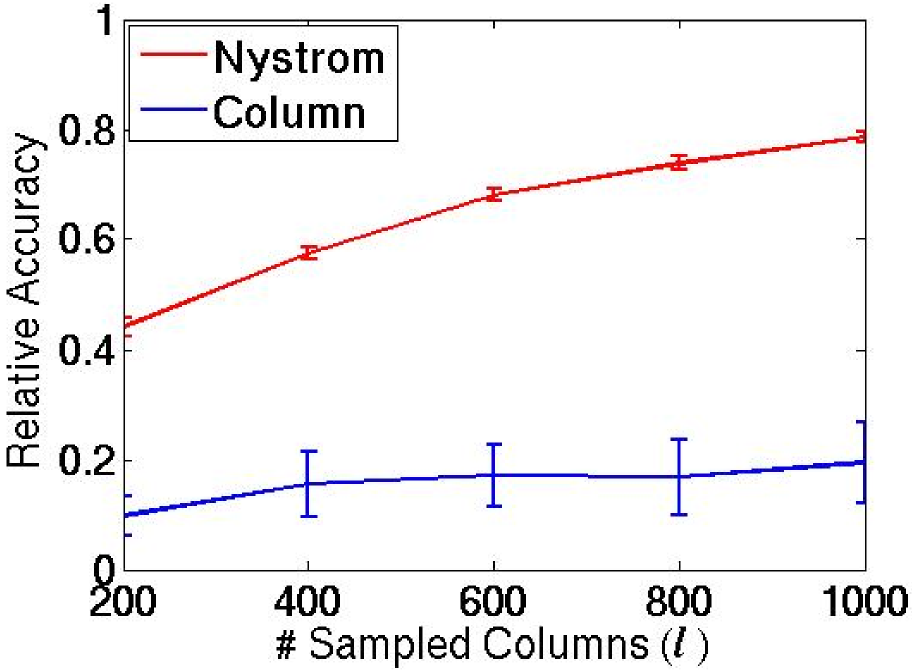
$k = 100$

$$\left.\begin{aligned} \tilde{U}_{col} &= C V_c \Sigma_c^{-1} \\ \tilde{\Sigma}_{col} &= \sqrt{\frac{n}{l}} \Sigma_c \end{aligned}\right\} \quad \tilde{G}_{col} = C\left(\left[\frac{l}{n} C^T C\right]^{1/2}\right)^{-1} C^T$$

Low-Rank Approximations

Spectral reconstruction: $\tilde{G}_k = \tilde{U}_k \tilde{\Sigma}_k \tilde{U}_k^T$

$$\frac{\left\| G - \hat{G}_k \right\|_F}{\left\| G - \tilde{G}_k \right\|_F}$$



$k = 100$

$$\left.\begin{aligned} \tilde{U}_{col} &= C V_c \Sigma_c^{-1} \\ \tilde{\Sigma}_{col} &= \sqrt{\frac{n}{l}} \Sigma_c \end{aligned}\right\} \quad \tilde{G}_{col} = C\left(\left[\frac{l}{n} C^T C\right]^{1/2}\right)^{-1} C^T$$

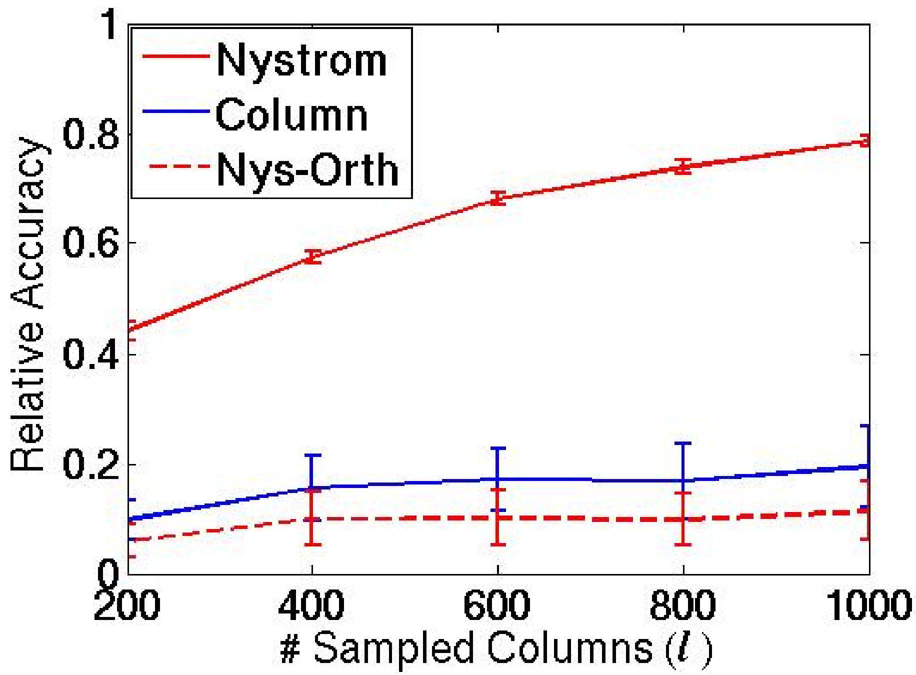
$$\left.\begin{aligned} \tilde{U}_{nys} &= \sqrt{\frac{l}{n}} C U_W \Sigma_W^{-1} \\ \tilde{\Sigma}_{nys} &= \frac{n}{l} \Sigma_W \end{aligned}\right\} \quad \tilde{G}_{nys} = C W^{-1} C^T$$

How about orthonormalized Nystrom eigenvectors?

Orthogonalized Nystrom

Spectral reconstruction: $\tilde{G}_k = \tilde{U}_k \tilde{\Sigma}_k \tilde{U}_k^T$

$$\frac{\|G - \hat{G}_k\|_F}{\|G - \tilde{G}_k\|_F}$$



$k = 100$

Nystrom-orthogonal gives worse reconstruction than Nystrom !

Low-Rank Approx: Matrix Projection

$$G_k = U_k \Sigma_k U_k^T = U_k U_k^T G = G U_k U_k^T$$

$$\tilde{G}_k = \boxed{\tilde{U}_k \tilde{U}_k^T G} \neq \tilde{U}_k \Sigma_k \tilde{U}_k^T$$

Low-Rank Approx: Matrix Projection

$$G_k = U_k \Sigma_k U_k^T = U_k U_k^T G = G U_k U_k^T$$

$$\tilde{G}_k = \tilde{U}_k \tilde{U}_k^T G \neq \tilde{U}_k \Sigma_k \tilde{U}_k^T$$

$$\tilde{G}_{col} = C\left(C^T C\right)^{-1} C^T G$$

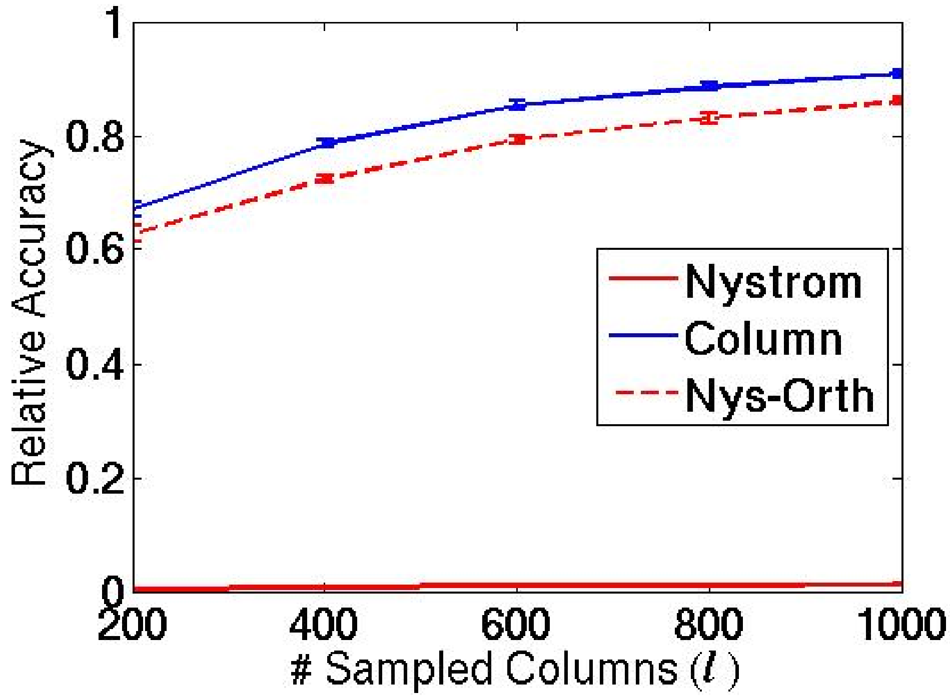
Reconstructs C exactly!

$$\tilde{G}_{nys} = C\left(\frac{l}{n} W^{-2}\right) C^T G$$

Low-Rank Approx: Matrix Projection

$$\tilde{G}_k = \tilde{U}_k \tilde{U}_k^T G$$

$$\frac{\|G - \hat{G}_k\|_F}{\|G - \tilde{G}_k\|_F}$$



$k = 100$

Col-Sampling gives better Reconstruction than Nystrom !

If $k = l$, Col-Sampling and Nystrom-orthogonal give the same answer !

Low-Rank Approx: Matrix Projection

Why does Col-sampling perform better than Nystrom?

Theorem: The matrix projection reconstruction for both Nystrom and Col-sampling is of the form $\tilde{G}_k = U_c R U_c^T G$, where R is SPSD. Col-sampling gives the lowest reconstruction error (in Frobenius norm) among all such approximations when $k = l$.

Low-Rank Approx: Matrix Projection

Why does Col-sampling perform better than Nystrom?

Theorem: The matrix projection reconstruction for both Nystrom and Col-sampling is of the form $\tilde{G}_k = U_c R U_c^T G$, where R is SPSD. Col-sampling gives the lowest reconstruction error (in Frobenius norm) among all such approximations when $k = l$.

Partial proof: Let's look at the difference between any generic approx of the above form vs col-sampling approximation

$$E - E_{col} = \left\| G - U_c R U_c^T G \right\|_F^2 - \left\| G - U_c U_c^T G \right\|_F^2$$

For col-sampling, $R = I$

$$= Tr[G^T (U_c R^2 U_c^T - 2 U_c R U_c^T + U_c U_c^T) G]$$

$\|A\|_F^2 = Tr[A^T A]$

Low-Rank Approx: Matrix Projection

Why does Col-sampling perform better than Nystrom?

Theorem: The matrix projection reconstruction for both Nystrom and Col-sampling is of the form $\tilde{G}_k = U_c R U_c^T G$, where R is SPSD. Col-sampling gives the lowest reconstruction error (in Frobenius norm) among all such approximations when $k = l$.

Partial proof: Let's look at the difference between any generic approx of the above form vs col-sampling approximation

$$E - E_{col} = \left\| G - U_c R U_c^T G \right\|_F^2 - \left\| G - U_c U_c^T G \right\|_F^2$$

For col-sampling, $R = I$

$$= Tr[G^T (U_c R^2 U_c^T - 2U_c R U_c^T + U_c U_c^T) G]$$

$\left\| A \right\|_F^2 = Tr[A^T A]$

$$= Tr[(R - I) U_c^T G)^T (R - I) U_c^T G)]$$

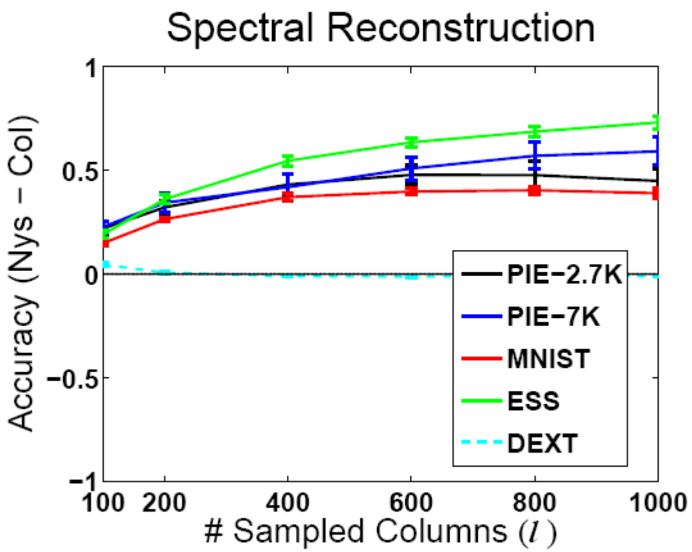
$$\geq 0$$

$Tr[A^T A] \geq 0$

Low-Rank Approx: Spectral Reconstruction

Why does Nystrom perform better than Col-sampling?

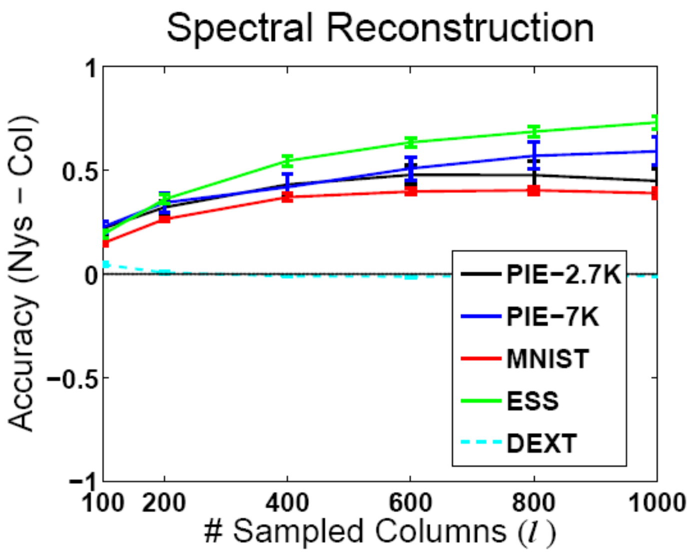
Unfortunately no clean theorem! Depends on the data spectrum!



Low-Rank Approx: Spectral Reconstruction

Why does Nystrom perform better than Col-sampling?

Unfortunately no clean theorem! Depends on the data spectrum!

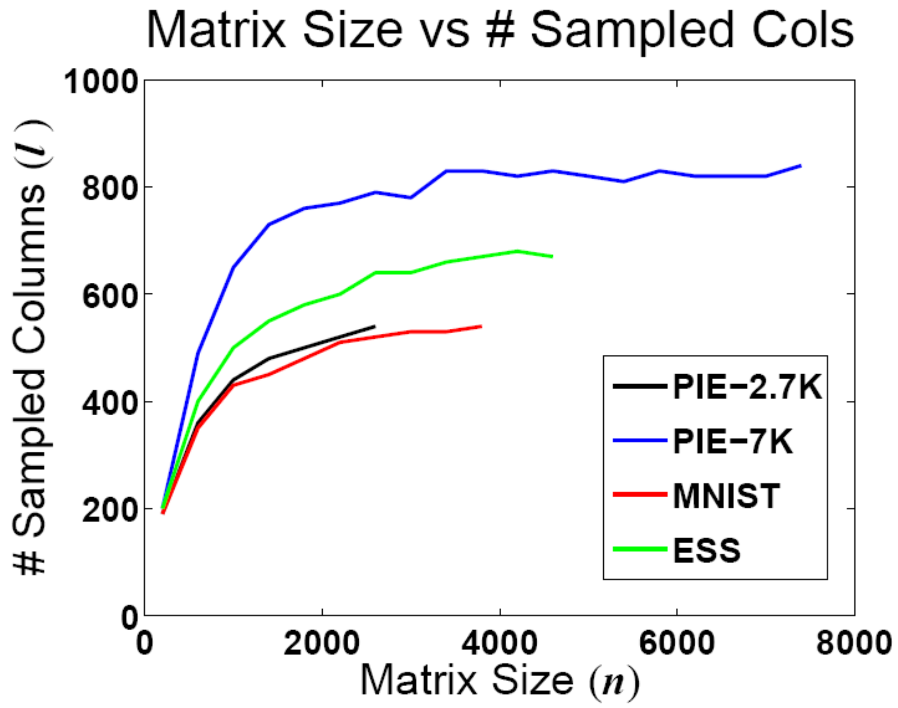


Theorem: Suppose, $r = \text{rank}(G) = \text{rank}(W),\ r \le k \le l$, then Nystrom approximation gives exact Spectral Reconstruction. In contrast, Col-sampling gives the same result iff it reduces to Nystrom form, i.e.,

$$W = ((l/n)C^T C)^{1/2}$$

How many columns are needed?

Columns needed to get 75% relative accuracy



Formal Statements

Formal procedures for Nystrom and Col-sampling methods

Bounds on Errors

Column-Sampling Method

Algorithm

Given $A, \; 1 \le k \le l \le n, \; \{p_i\}_{i=1}^{n}$ s.t. $\sum_i p_i = 1, \; p_i \ge 0$

Output $\tilde{U}_k, \; \tilde{\Sigma}_k$

Column-Sampling Method

Algorithm

Given $A,\ 1 \le k \le l \le n,\ \{p_i\}_{i=1}^{n}$ s.t. $\sum_i p_i = 1,\ \ p_i \ge 0$

Output $\tilde{U}_k,\ \tilde{\Sigma}_k$

For $t = 1,\ldots,l$

- Pick $i_t \in \{1,\ldots,n\}$ with $P(i_t = k) = p_k$ independently, with replacement

  fixed non-uniform distribution

- Set $C^{(t)} = A^{(i_t)} / \sqrt{l\, p_{i_t}}$

Compute $C^T C$ and decompose $C^T C = V_c^T \Sigma_c^2 V_c$

Return $\tilde{\Sigma}_k = \Sigma_{c,k}$ and $\tilde{U}_k = U_{c,k} = C V_{c,k} \Sigma_{c,k}^{-1}$

Column-Sampling Method

Bound on error

$$\left\| A - \tilde{U}_k \tilde{U}_k^T A \right\|_F^2 \le \left\| A - A_k \right\|_F^2 + 2\sqrt{k} \left\| AA^T - CC^T \right\|_F^2$$

best rank-k matrix: $A_k = U_k U_k^T A$

matrix-multiplication bound

Column-Sampling Method

Bound on error

$$\left\| A - \tilde{U}_k \tilde{U}_k^T A \right\|_F^2 \le \left\| A - A_k \right\|_F^2 + 2\sqrt{k} \left\| AA^T - CC^T \right\|_F^2$$

best rank-k matrix: $A_k = U_k U_k^T A$

matrix-multiplication bound

If $p_i \ge \beta \left| A^{(i)} \right|^2 / \left\| A \right\|_F^2$, $\beta \le 1$, $\eta = 1 + \sqrt{(8/\beta)\log(1/\delta)}$ and $l \ge 4k/\beta\varepsilon^2$

with probability at least (1-δ)

$$\boxed{\left\| A - \tilde{U}_k \tilde{U}_k^T A \right\|_F^2 \le \left\| A - A_k \right\|_F^2 + \varepsilon \left\| A \right\|_F^2}$$

Column-Sampling Method

Bound on error

$$\left\| A - \tilde{U}_k \tilde{U}_k^T A \right\|_F^2 \le \left\| A - A_k \right\|_F^2 + 2\sqrt{k} \left\| AA^T - CC^T \right\|_F^2$$

best rank-k matrix: $A_k = U_k U_k^T A$

matrix-multiplication bound

If $p_i \ge \beta \left| A^{(i)} \right|^2 / \left\| A \right\|_F^2$, $\beta \le 1$, $\eta = 1 + \sqrt{(8/\beta)\log(1/\delta)}$ and $l \ge 4k/\beta\varepsilon^2$

overestimate! In practice, much smaller

with probability at least (1-δ)

$$\boxed{\left\| A - \tilde{U}_k \tilde{U}_k^T A \right\|_F^2 \le \left\| A - A_k \right\|_F^2 + \varepsilon \left\| A \right\|_F^2}$$

Matrix-projection view

Nystrom Method

Originally developed as a tool for numerical integration. When applied to eigenfunction estimation problem with quadrature rule, it allows extrapolation on full domain.

Algorithm

Given $G,\ 1 \le k \le l \le n,\ \{p_i\}_{i=1}^{n}$ s.t. $\sum_i p_i = 1,\ \ p_i \ge 0$

Output $\tilde{G}_k$

Nystrom Method

Originally developed as a tool for numerical integration. When applied to eigenfunction estimation problem with quadrature rule, it allows extrapolation on full domain.

Algorithm

Given $G,\ 1 \le k \le l \le n,\ \{p_i\}_{i=1}^{n}$ s.t. $\sum_i p_i = 1,\ \ p_i \ge 0$

Output $\tilde{G}_k$

- Pick $i \in I \subset \{1,\ldots,n\}$ with $P(i = k) = p_k$ independently, with replacement

  fixed non-uniform distribution

- Set $C = [G^{(i)} / \sqrt{l\, p_i}\,]$

- Select corresponding rows of C and form W such that each entry is

$$W_{ij} = G_{ij} / l\sqrt{p_i p_j} \qquad i, j \in I$$

Return $\tilde{G}_k = C W_k^{-1} C^T$

Nystrom Method

Bound on error

If $p_i = G_{ii}^2 / \sum_i G_{ii}^2$, $\eta = 1 + \sqrt{8 \log(1/\delta)}$ and $l \geq 64 k \eta^2 / \varepsilon^4$

with probability at least (1-δ)

$$\boxed{\left\| G - C W^{-1} C^T \right\|_F \leq \left\| G - G_k \right\|_F + \varepsilon \sum_{i=1}^{n} G_{ii}^2}$$

best rank-k matrix: $G_k = U_k \Sigma_k U_k^T$

matrix-multiplication bound

Overview

1. Approximate Matrix Multiplication
   – Sample columns of one matrix and rows from the other
2. Column-sampling methods for spectral decomposition
   – Methods that use decomposition of entire sampled columns
   – Methods that further sample the rows from the sampled columns
3. Low-rank approximation
   – Spectral reconstructions $A_k = U_k \Sigma_k V_k^T$
   – Matrix Projection $A_k = U_k U_k^T A$
4. Sampling Techniques
5. Ensemble Methods
   – How to combine multiple approximations to yield more accurate one

Sampling Techniques

Fixed-Distribution Sampling methods

- Pick the columns randomly with equal probability
- Pick the columns proportional to their L_2 norm
- Pick the columns proportional to their diagonal entries

Advantages

- Uniform sampling – very fast (constant time and space) and has been shown to work well in practice
- Data-dependent methods also provide fast sampling

Disadvantages

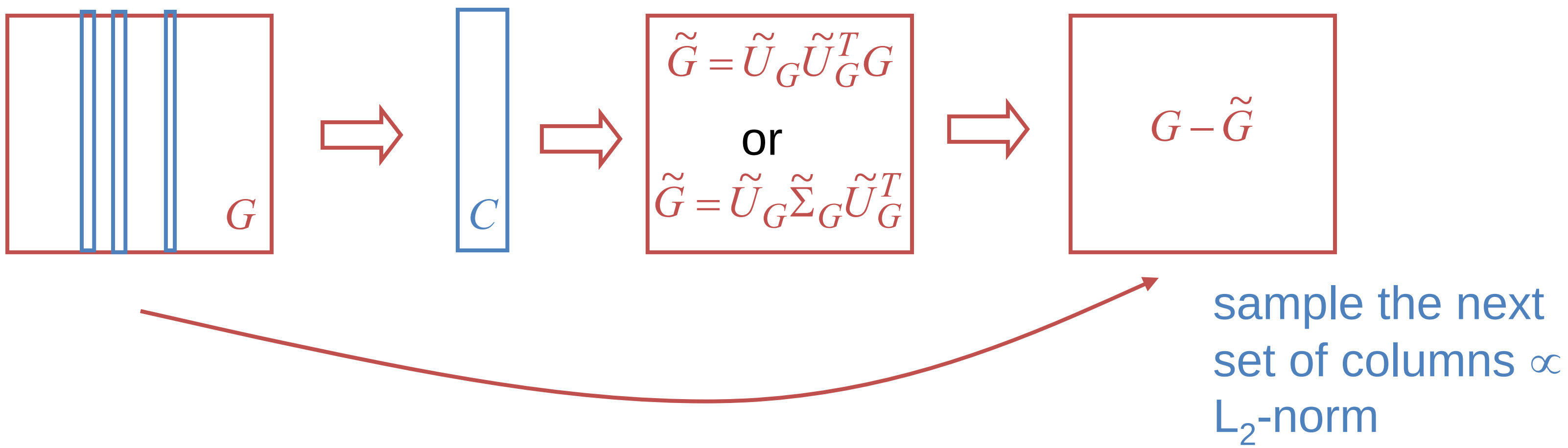
- L_2-norm based methods need one pass through the entire matrix
- Expensive for large scale applications since each entry of the matrix is to be reconstructed → $O(n^2)$

Adaptive Sampling Techniques

Distribution over columns changes each time a column subset is picked

Basic Idea

- Reconstruct the matrix given all the samples selected so far
- Find out reconstruction error for each column
- Pick the columns proportional to the reconstruction error



Adaptive Sampling Techniques

Distribution over columns changes each time a column subset is picked

Basic Idea

- Reconstruct the matrix given all the samples selected so far
- Find out reconstruction error for each column
- Pick the columns proportional to the reconstruction error

Issues

- Usually much better than the fixed-distribution sampling methods
- Quite expensive for large scale applications
- Each entry of the matrix is to be reconstructed many times iteratively → $O(ln^2)$

Tighter Error Bound
$l \geq kt / \varepsilon$

$$\boxed{\left\| A - \tilde{U}_k \tilde{U}_k^T A \right\|_F^2 \leq (1/1-\varepsilon) \left\| A - A_k \right\|_F^2 + \varepsilon^t \left\| A \right\|_F^2}$$

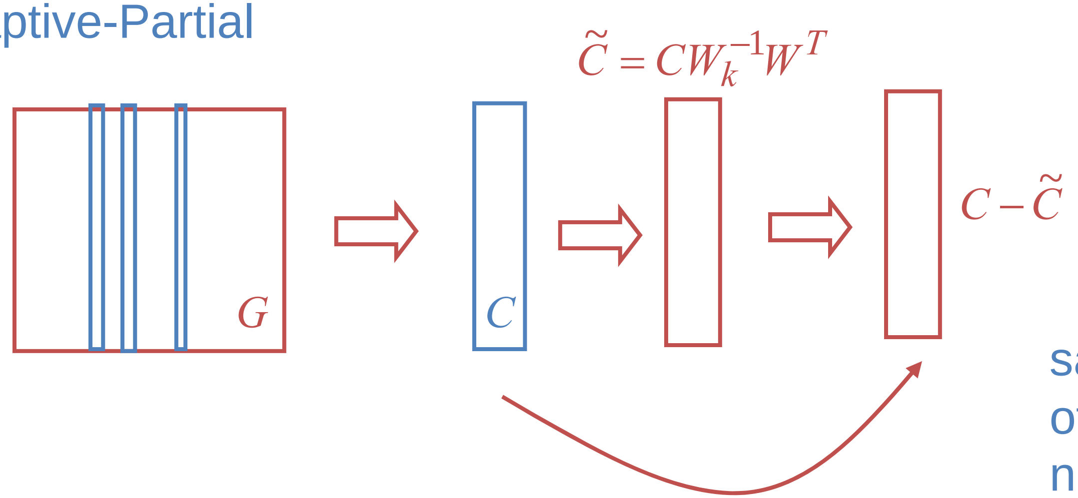
Adaptive Sampling Techniques

Distribution over columns changes each time a column subset is picked

Basic Idea

- Reconstruct the matrix given all the samples selected so far
- Find out reconstruction error for each column
- Pick the columns proportional to the reconstruction error

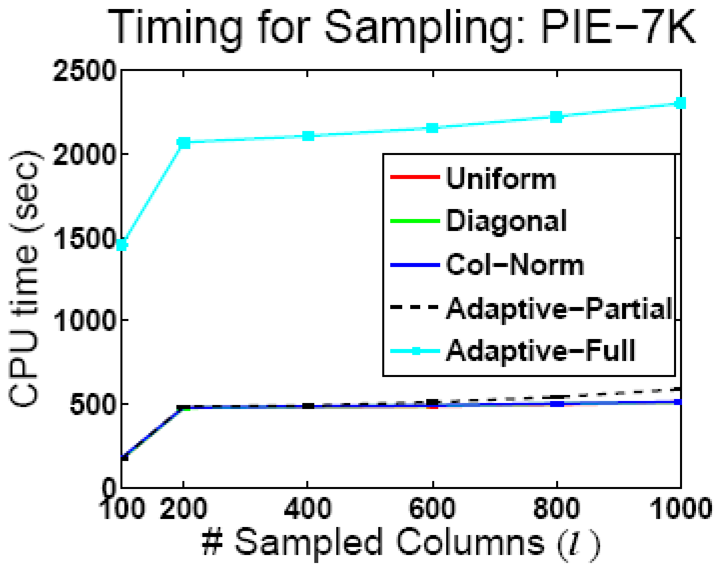
Adaptive-Partial



sample the next set of columns $\propto$ L_2-norm of rows in C

Experiments - Sampling Methods

l	Dataset	Uniform	Diagonal	Col-Norm	Adapt-Part	Adapt-Full
400	PIE-2.7K	67.2 (±1.1)	62.1 (±0.9)	59.7 (±1.0)	70.4 (±0.9)	**72.6 (±1.0)**
	PIE-7K	57.5 (±1.1)	50.8 (±1.9)	56.8 (±1.6)	62.8 (±0.9)	**64.3 (±0.7)**
	MNIST	67.4 (±0.7)	67.4 (±0.4)	65.3 (±0.5)	**69.3 (±0.7)**	69.2 (±0.7)
	ESS	61.0 (±1.7)	61.5 (±1.5)	57.5 (±1.9)	**65.0 (±1.0)**	63.9 (±0.9)
800	PIE-2.7K	84.1 (±0.5)	77.8 (±0.6)	73.9 (±1.0)	86.5 (±0.4)	**87.7 (±0.4)**
	PIE-7K	73.8 (±1.2)	64.9 (±1.8)	71.8 (±3.0)	**78.5 (±0.5)**	74.1 (±0.6)
	MNIST	83.3 (±0.3)	83.0 (±0.3)	80.4 (±0.4)	**84.2 (±0.4)**	80.7 (±0.5)
	ESS	78.1 (±1.0)	79.2 (±0.9)	75.4 (±1.2)	**80.6 (±1.1)**	74.8 (±0.8)



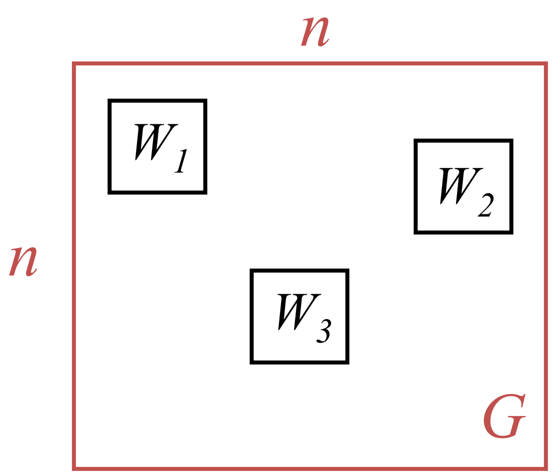
Overview

1. Approximate Matrix Multiplication
   – Sample columns of one matrix and rows from the other
2. Column-sampling methods for spectral decomposition
   – Methods that use decomposition of entire sampled columns
   – Methods that further sample the rows from the sampled columns
3. Low-rank approximation
   – Spectral reconstructions $A_k = U_k \Sigma_k V_k^T$
   – Matrix Projection $A_k = U_k U_k^T A$
4. Sampling Techniques
5. Ensemble Methods
   – How to combine multiple approximations to yield more accurate one

Ensemble Methods

So far…

- Nytrom Method picks a single square (usually non-contiguous) matrix from A



– Can we pick more such blocks and combine results to get better accuracy?
  - If yes, how to combine the results ?
  - Computational cost ?

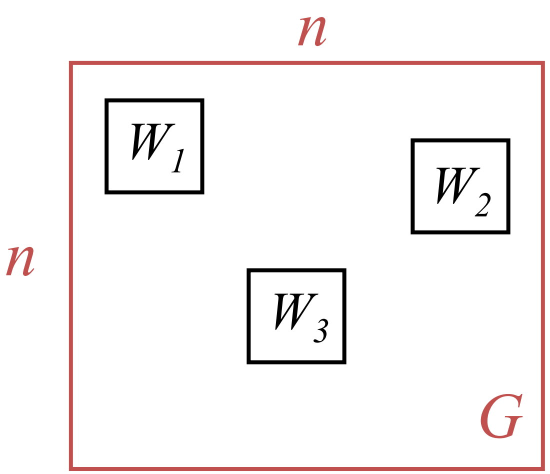
Ensemble Methods

Yes, it is possible…

Pick lp columns without replacement: Divide into p sets

$$\tilde{G}_r = C_r W_r^{+} C_r^{T} \qquad \text{for } r = 1, \ldots, p$$

Each C_r is non-overlapping



Ensemble Methods

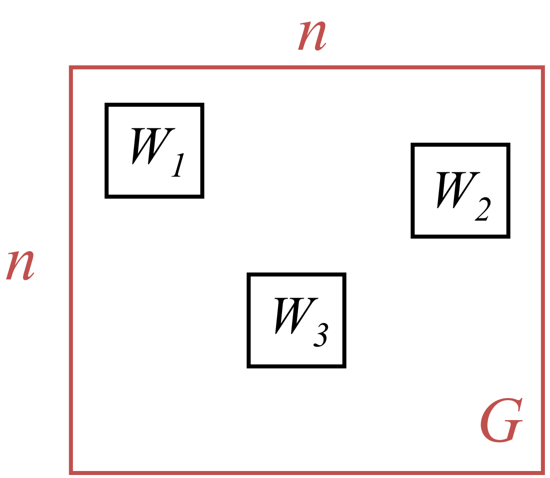
Yes, it is possible…

Pick lp columns without replacement: Divide into p sets

$$\tilde{G}_r = C_r W_r^{+} C_r^{T} \quad \text{for } r = 1, \ldots, p$$

Each C_r is non-overlapping

$$\tilde{G} = \sum_{r=1}^{p} \mu_r \tilde{G}_r$$



Ensemble Methods

Yes, it is possible…

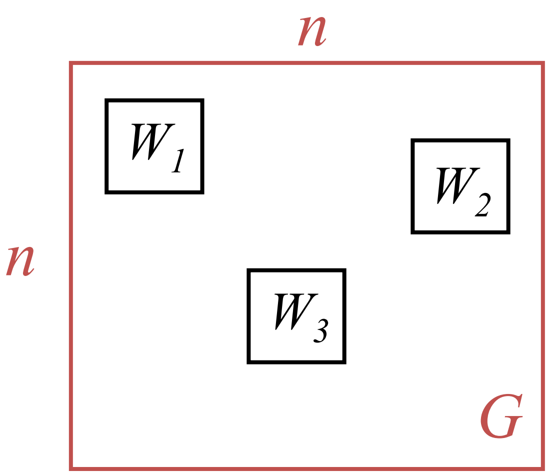
Pick lp columns without replacement: Divide into p sets

$$\tilde{G}_r = C_r W_r^{+} C_r^{T} \qquad \text{for } r = 1, \ldots, p$$

Each C_r is non-overlapping

$$\tilde{G} = \sum_{r=1}^{p} \mu_r \tilde{G}_r$$

mixture weights



– How to compute mixture weights?
  – simplest choice: $\mu_r = 1/p$

Ensemble Methods

Yes, it is possible…

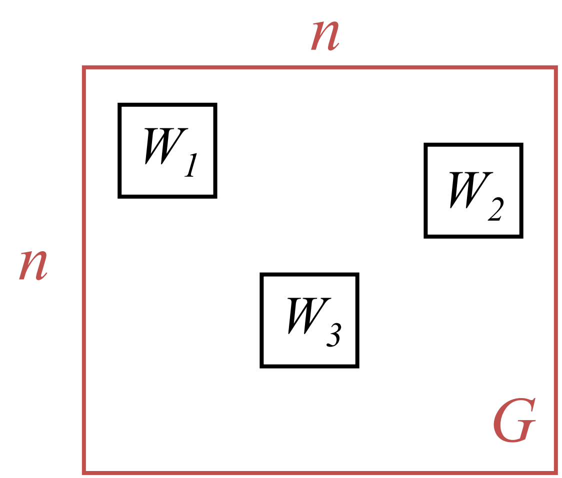
Pick lp columns without replacement: Divide into p sets

$$\tilde{G}_r = C_r W_r^{+} C_r^{T} \qquad \text{for } r = 1, \ldots, p$$

Each C_r is non-overlapping

$$\tilde{G} = \sum_{r=1}^{p} \mu_r \tilde{G}_r$$

mixture weights

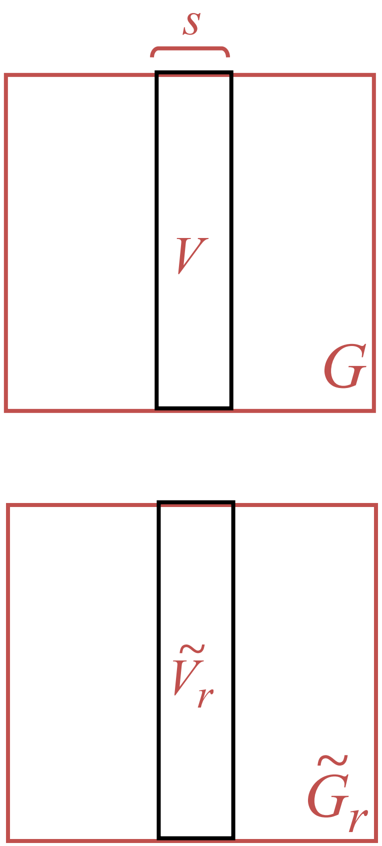


- How to compute mixture weights?
  - simplest choice: $\mu_r = 1/p$
  - Learn using "training data"
  - Sample s columns separate from previous lp columns, and measure error in reconstructing those by each "expert" in ensemble

Learning of Mixture Weights

Error in reconstruction for an expert

$$\varepsilon_r = \left\| V - \tilde{V}_r \right\|_F \quad \text{for } r = 1, \dots, p$$



Learning of Mixture Weights

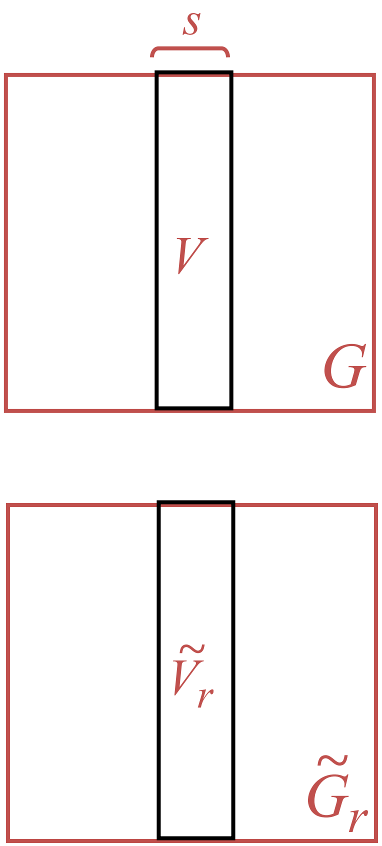
Error in reconstruction for an expert

$$\varepsilon_r = \left\| V - \tilde{V}_r \right\|_F \quad \text{for } r = 1, \ldots, p$$

Exponential weighting

$$\mu_r = \exp(-\eta \varepsilon_r) / Z \qquad \text{for } \eta > 0$$

Z is a normalizing constant such that $\sum_r \mu_r = 1$



Learning of Mixture Weights

Error in reconstruction for an expert

$$\varepsilon_r = \left\| V - \tilde{V}_r \right\|_F \quad \text{for } r = 1, \ldots, p$$

Exponential weighting

$$\mu_r = \exp(-\eta \varepsilon_r) / Z \qquad \text{for } \eta > 0$$

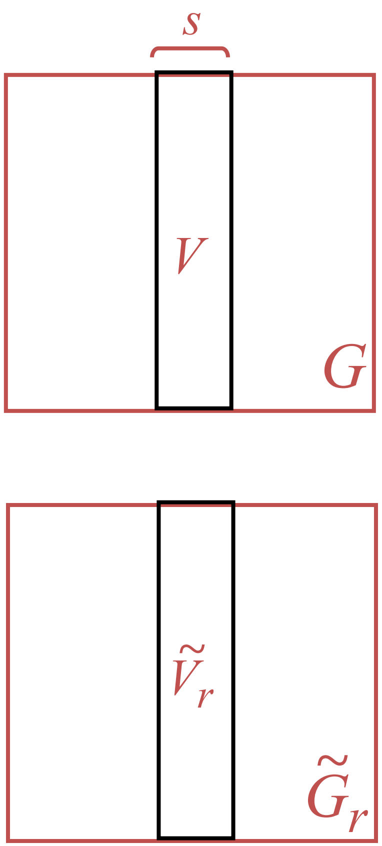
Z is a normalizing constant such that $\sum_r \mu_r = 1$

Linear (Ridge) Regression

– Try to find weights that best reconstruct V

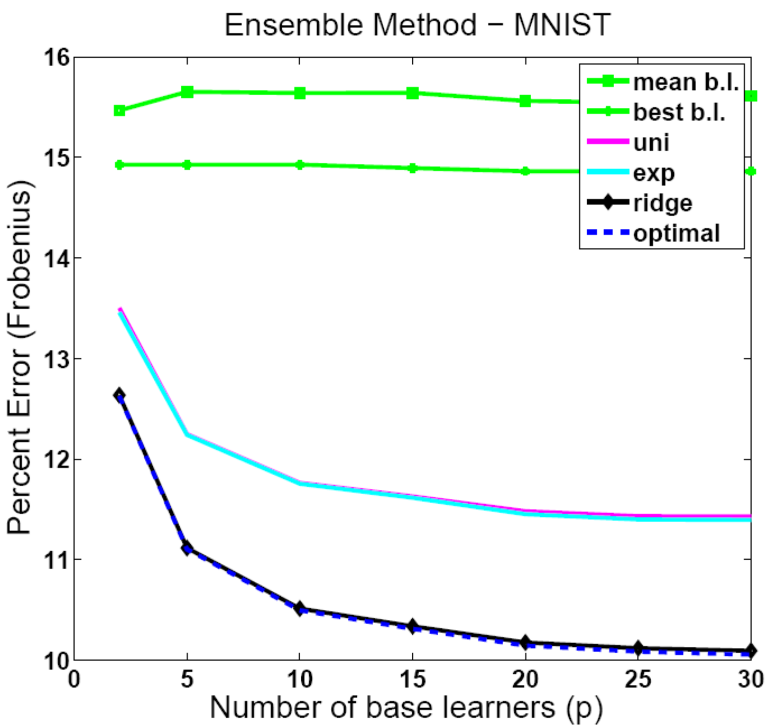
$$\mu = [\mu_1, \ldots, \mu_p]^T$$

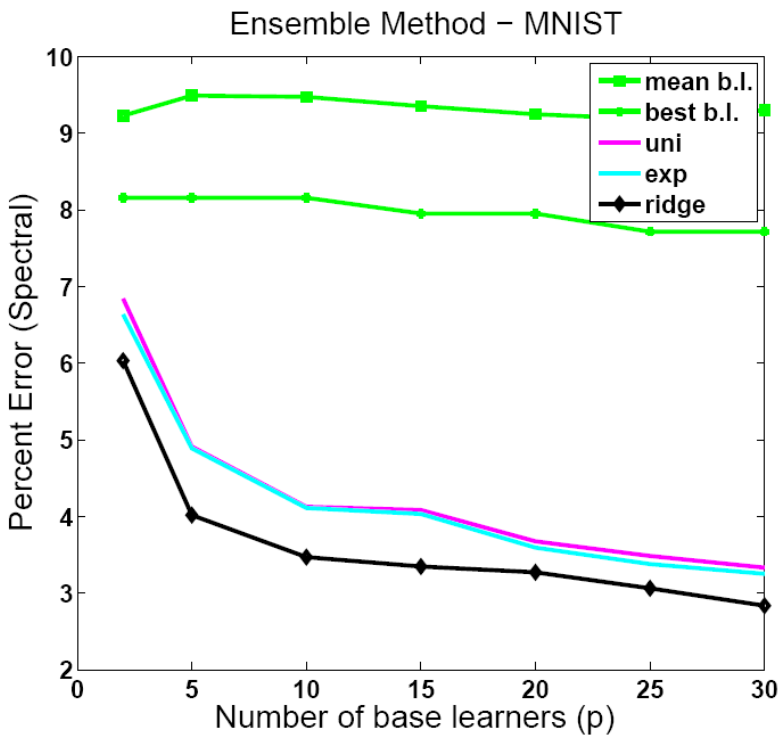
$$\hat{\mu} = \arg\min_{\mu} \left(\left\| \sum_r \mu_r \tilde{V}_r - V \right\|_F^2 + \lambda \left\| \mu \right\|_2^2 \right)$$



Examples

- MNIST dataset: n = 4000, s = 20, k = 50
- Optimal weights: linear regression with $s = n$

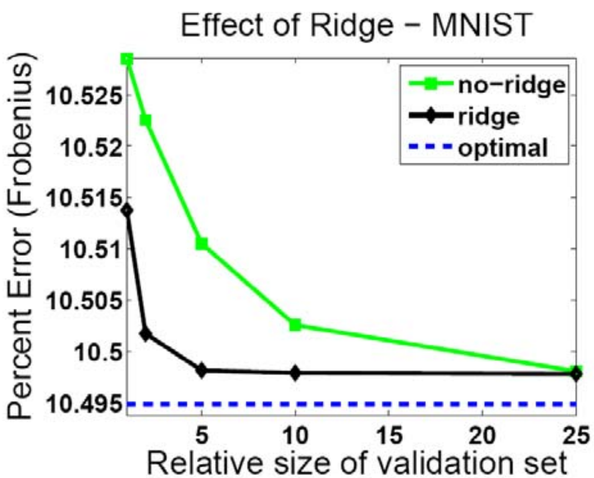




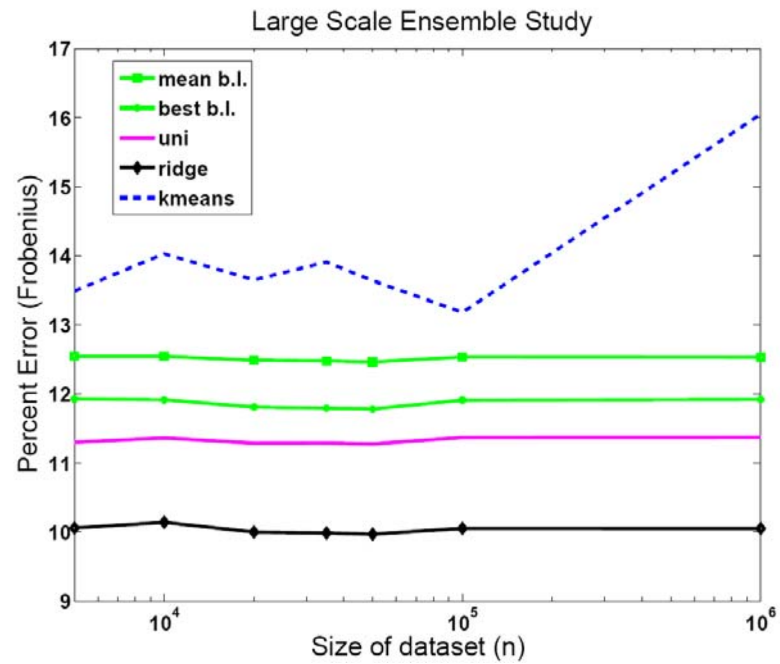
Examples

- How important is ridge penalty?
- Large-scale comparison

SIFT-1M dataset



$(s/n)\%$



Fixed-time experiment

$k = 50, p = 10, s = 2$

References

1. A. Frieze, R. Kannan and S. Vempala, Fast Monte-Carlo Algorithms for finding low-rank approximations, Proceedings of the Foundations of Computer Science, 1998.
2. P. Drineas and M. W. Mahoney, On the Nystrom method for approximating a Gram matrix for improved kernel-based learning, Journal of Machine Learning Research, 2005.
3. P. Drineas and R. Kannan, M. Mahoney, Fast Monte Carlo Algorithms for Matrices I: Approximating Matrix Multiplication, SIAM Journal on Computing, 2006.
4. P. Drineas, R. Kannan, and M. W. Mahoney, Fast monte carlo algorithms for matrices II: computing a low rank approximation to a matrix, SIAM Journal on Computing, 36(1), 2006.
5. Amit Deshpande, Santosh Vempala, Grant Wang, Matrix Approximation and Projective Clustering via Volume Sampling. Theory of Computing and SODA 2006**.**
6. A. Talwalkar, S. Kumar and H. A. Rowley, Large-Scale Manifold Learning, *IEEE Computer Vision and Pattern Recognition (CVPR), 2008.*
7. S. Kumar, M. Mohri and A. Talwalkar, Sampling Techniques for the Nystrom Method, *Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS), 2009.*
8. S. Kumar, M. Mohri and A. Talwalkar, On Sampling-based Approximate Spectral Decomposition, *International Conference on Machine Learning (ICML), 2009.*
9. S. Kumar, M. Mohri and A. Talwalkar, Ensemble Nystrom Method *Neural Information Processing Systems (NIPS), 2009.*

11. M. Li, J.T. Kwok, B. Lu, Making large-scale Nystrom approximation possible. Proceedings of the Twenty-Seventh International Conference on Machine Learning (*ICML*), June 2010.