Efficient Algorithms and Error Analysis for the Modified Nyström Method

Shusen Wang[1] Zhihua Zhang[2]

[1]Zhejiang University, China

[2]Shanghai Jiao Tong University, China

AISTATS 2014

Outline

Outline

Kernel methods

- **K**: $n \times n$ kernel matrix.
- Matrix inverse $\mathbf{b} = (\mathbf{K} + \alpha \mathbf{I}_n)^{-1}\mathbf{y}$
  - time complexity: $\mathcal{O}(n^3)$
  - performed by Gaussian process regression, least square SVM, kernel ridge regression
- Partial eigenvalue decomposition of **K**
  - time complexity: $\mathcal{O}(n^2 k)$
  - performed by kernel PCA and some manifold learning methods

Kernel methods

- **K**: $n \times n$ kernel matrix.
- Matrix inverse $\mathbf{b} = (\mathbf{K} + \alpha \mathbf{I}_n)^{-1}\mathbf{y}$
  - time complexity: $\mathcal{O}(n^3)$
  - performed by Gaussian process regression, least square SVM, kernel ridge regression
- Partial eigenvalue decomposition of **K**
  - time complexity: $\mathcal{O}(n^2 k)$
  - performed by kernel PCA and some manifold learning methods

Kernel methods

- $\mathbf{K}$: $n \times n$ kernel matrix.
- Matrix inverse $\mathbf{b} = (\mathbf{K} + \alpha \mathbf{I}_n)^{-1}\mathbf{y}$
  - time complexity: $\mathcal{O}(n^3)$
  - performed by Gaussian process regression, least square SVM, kernel ridge regression
- Partial eigenvalue decomposition of $\mathbf{K}$
  - time complexity: $\mathcal{O}(n^2 k)$
  - performed by kernel PCA and some manifold learning methods

Computational Challenges

- High time complexities: $\mathcal{O}(n^3)$ or $\mathcal{O}(n^2k)$
- High space complexity: $\mathcal{O}(n^2)$
  - the iterative algorithms go many passes through the data
  - you had better put the entire kernel matrix in RAM
  - if the data does not fit in the RAM
    $\Rightarrow$ one swap between RAM and disk in each pass
    $\Rightarrow$ very slow!

Computational Challenges

- High time complexities: $\mathcal{O}(n^3)$ or $\mathcal{O}(n^2k)$
- High space complexity: $\mathcal{O}(n^2)$
  - the iterative algorithms go many passes through the data
  - you had better put the entire kernel matrix in RAM
  - if the data does not fit in the RAM
    $\Rightarrow$ one swap between RAM and disk in each pass
    $\Rightarrow$ very slow!

How to Speedup

- If we can find a fast low-rank factorization

$$\underbrace{\mathbf{K}}_{n\times n} \approx \underbrace{\mathbf{D}}_{n\times d}\underbrace{\mathbf{D}^T}_{d\times n},$$

then $(\mathbf{K}+\alpha\mathbf{I}_n)^{-1}$ and the partial eigenvalue decomposition of $\mathbf{K}$ can be approximated solved highly efficiently.

How to Speedup: Example 1

- Suppose we have a low-rank factorization

$$\underbrace{\mathbf{K}}_{n\times n} \approx \underbrace{\mathbf{D}}_{n\times d}\underbrace{\mathbf{D}^T}_{d\times n}.$$

- Approximately compute the matrix inverse $(\mathbf{K}+\alpha\mathbf{I}_n)^{-1}$ as follows.
- Expand $(\mathbf{D}\mathbf{D}^T+\alpha\mathbf{I}_n)^{-1}$ using the Sherman-Morrison-Woodbury formula and obtain

$$\left(\mathbf{D}\mathbf{D}^T+\alpha\mathbf{I}_n\right)^{-1} = \alpha^{-1}\mathbf{I}_n - \alpha^{-1}\underbrace{\mathbf{D}}_{n\times d}\left(\underbrace{\alpha\mathbf{I}_d+\mathbf{D}^T\mathbf{D}}_{d\times d}\right)^{-1}\underbrace{\mathbf{D}^T}_{d\times n}.$$

It costs only $\mathcal{O}(nd^2)$ time and $\mathcal{O}(nd)$ space to compute

$$\mathbf{b} = \left(\mathbf{D}\mathbf{D}^T+\alpha\mathbf{I}_n\right)^{-1}\mathbf{y}.$$

How to Speedup: Example 2

- Suppose we have a low-rank factorization

$$\underbrace{\mathbf{K}}_{n\times n} \approx \underbrace{\mathbf{D}}_{n\times d}\underbrace{\mathbf{D}^T}_{d\times n},$$

- Compute the eigenvalue decomposition of $\mathbf{K}$ as follows.

- Compute the eigenvalue decomposition of the $d \times d$ small matrix $\mathbf{S} = \mathbf{D}^T\mathbf{D} \in \mathbb{R}^{d\times d}$:

$$\mathbf{S} = \mathbf{U}_\mathbf{S}\mathbf{\Lambda}_\mathbf{S}\mathbf{U}_\mathbf{S}^T.$$

The partial eigenvalue decomposition of $\mathbf{D}\mathbf{D}^T$ is

$$\mathbf{K} \approx \mathbf{D}\mathbf{D}^T = \left(\mathbf{D}\mathbf{U}_\mathbf{S}\mathbf{\Lambda}_\mathbf{S}^{-1/2}\right)\mathbf{\Lambda}_\mathbf{S}\left(\mathbf{D}\mathbf{U}_\mathbf{S}\mathbf{\Lambda}_\mathbf{S}^{-1/2}\right)^T$$

It costs only $\mathcal{O}(nd^2)$ time and $\mathcal{O}(nd)$ space.

Outline

The Nyström Method

- **Random Selection**:
  selects c ($\ll n$) columns of $\mathbf{K}$ to construct $\mathbf{C}$ using some randomized algorithms. After permutation we have

$$\mathbf{K} = \begin{bmatrix} \mathbf{W} & \mathbf{K}_{21}^T \\ \mathbf{K}_{21} & \mathbf{K}_{22} \end{bmatrix}, \qquad \mathbf{C} = \begin{bmatrix} \mathbf{W} \\ \mathbf{K}_{21} \end{bmatrix}.$$

- **The Nyström Approximation**: $\tilde{\mathbf{K}}_c^{\text{nys}} \approx \mathbf{K}$

$$\underbrace{\tilde{\mathbf{K}}_c^{\text{nys}}}_{n\times n} = \underbrace{\mathbf{C}}_{n\times c}\underbrace{\mathbf{W}^{\dagger}}_{c\times c}\underbrace{\mathbf{C}^T}_{c\times n}.$$

The Nyström Method

- **Random Selection**:
  selects c ($\ll n$) columns of $\mathbf{K}$ to construct $\mathbf{C}$ using some randomized algorithms. After permutation we have

$$\mathbf{K} = \begin{bmatrix} \mathbf{W} & \mathbf{K}_{21}^T \\ \mathbf{K}_{21} & \mathbf{K}_{22} \end{bmatrix}, \qquad \mathbf{C} = \begin{bmatrix} \mathbf{W} \\ \mathbf{K}_{21} \end{bmatrix}.$$

- **The Nyström Approximation**: $\tilde{\mathbf{K}}_c^{\text{nys}} \approx \mathbf{K}$

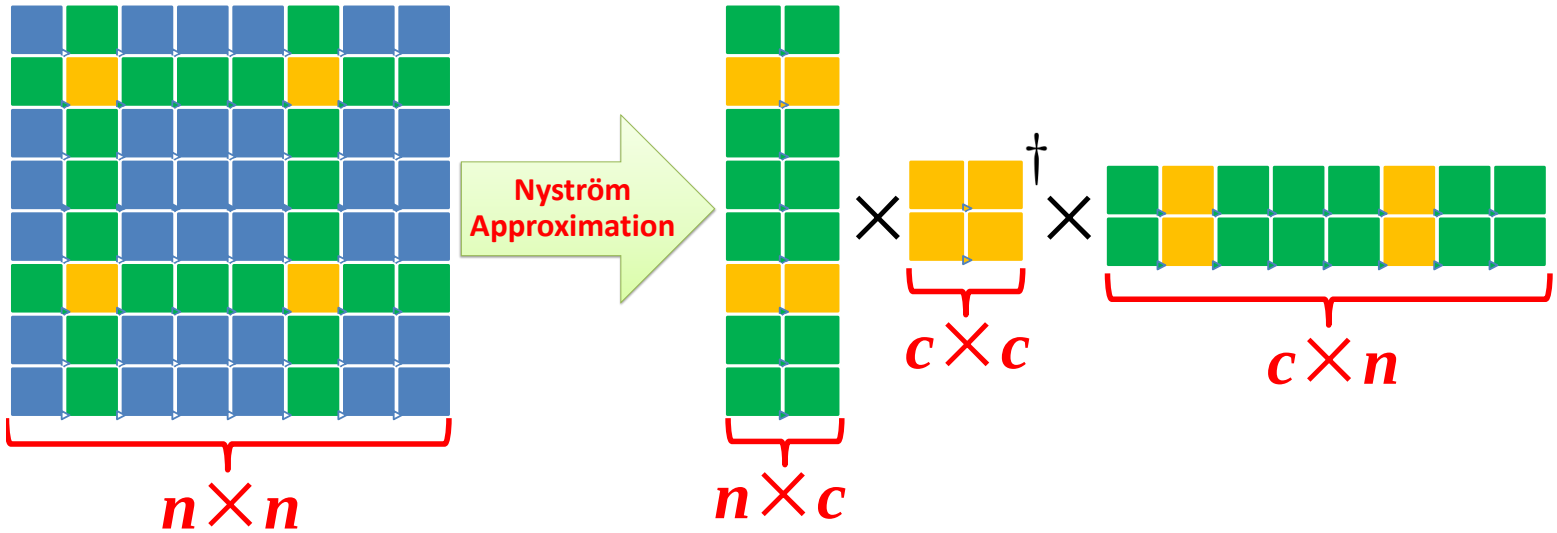
$$\underbrace{\tilde{\mathbf{K}}_c^{\text{nys}}}_{n\times n} = \underbrace{\mathbf{C}}_{n\times c}\,\underbrace{\mathbf{W}^{\dagger}}_{c\times c}\,\underbrace{\mathbf{C}^T}_{c\times n}.$$

The Nyström Approximation

- **The Nyström Approximation**:

$$\mathbf{K} \approx \tilde{\mathbf{K}}_c^{\text{nys}} = \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T$$

(A low-rank factorization).



Outline

Problem Formulation

Problem:

- How to select informative columns of $\mathbf{K} \in \mathbb{R}^{n \times n}$ to construct $\mathbf{C} \in \mathbb{R}^{n \times c}$?
- The approximation error $\left\|\mathbf{K} - \mathbf{C}\mathbf{U}\mathbf{C}^T\right\|_F$ or $\left\|\mathbf{K} - \mathbf{C}\mathbf{U}\mathbf{C}^T\right\|_2$ should be as small as possible.

Hardness:

- Totally $\binom{n}{c}$ choices.

Problem Formulation

Problem:

- How to select informative columns of $\mathbf{K} \in \mathbb{R}^{n \times n}$ to construct $\mathbf{C} \in \mathbb{R}^{n \times c}$?
- The approximation error $\left\|\mathbf{K} - \mathbf{C}\mathbf{U}\mathbf{C}^T\right\|_F$ or $\left\|\mathbf{K} - \mathbf{C}\mathbf{U}\mathbf{C}^T\right\|_2$ should be as small as possible.

Hardness:

- Totally $\binom{n}{c}$ choices.

Criterion: Upper Error Bounds

- Using approximation algorithms to find *c good* columns (not necessarily the *best*)
- Hope that $\frac{\|\mathbf{K}-\mathbf{C}\mathbf{U}\mathbf{C}^T\|_F}{\|\mathbf{K}-\mathbf{K}_k\|_F}$ has upper bound, which is the smaller the better.

Uniform Sampling: The Simplest Algorithm

- Sample c columns of $\mathbf{K}$ uniformly at random to construct $\mathbf{C}$.
- The simplest, but the most widely used.

Adaptive Sampling

The adaptive sampling algorithm [Deshpande *et al.* , 2006]:

1. Sample c_1 columns of $\mathbf{K}$ to construct $\mathbf{C}_1$ using some algorithm;
2. Compute the residual $\mathbf{B} = \mathbf{K} - \mathbf{C}_1\mathbf{C}_1^{\dagger}\mathbf{K}$;
3. Compute sampling probabilities $p_i = \frac{\|\mathbf{b}_i\|_2^2}{\|\mathbf{B}\|_F^2}$, for $i = 1$ to n;
4. Sample further c_2 columns of $\mathbf{K}$ in c_2 i.i.d. trials, in each trial the i-th column is chosen with probability p_i; Denote the selected columns by $\mathbf{C}_2$;
5. Return $\mathbf{C} = [\mathbf{C}_1 , \mathbf{C}_2]$.

Adaptive Sampling

- The error term $\|\mathbf{K} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{K}\|_F$ is bounded theoretically, but $\|\mathbf{K} - \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T\|_F$ is not.
- Empirically, the adaptive sampling algorithm works very well.

Outline

How to Improve the Nyström Approximation?

- Devise better sampling algorithms to improve the upper error bounds.
- Use other types of low-rank approximation instead of the Nyström approximation $\mathbf{K} \approx \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}$.

How to Improve the Nyström Approximation?

- Devise better sampling algorithms to improve the upper error bounds.
- Use other types of low-rank approximation instead of the Nyström approximation $\mathbf{K} \approx \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}$.

Better Sampling Algorithms?

- We hope $\frac{\|\mathbf{K}-\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}\|_F}{\|\mathbf{K}-\mathbf{K}_k\|_F}$ will be very small if the column sampling algorithm is good enough.
- But it cannot be arbitrarily small.
- Lower Error Bound

Theorem (Wang & Zhang, JMLR 2013)

Whatever column sampling is used to select c columns, there exists a bad case $\mathbf{K}$ *such that*

$$\frac{\|\mathbf{K}-\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}\|_F^2}{\|\mathbf{K}-\mathbf{K}_k\|_F^2} \geq \Omega\left(1+\frac{nk}{c^2}\right).$$

Better Sampling Algorithms?

- We hope $\frac{\|\mathbf{K}-\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}\|_F}{\|\mathbf{K}-\mathbf{K}_k\|_F}$ will be very small if the column sampling algorithm is good enough.
- But it cannot be arbitrarily small.
- Lower Error Bound

Theorem (Wang & Zhang, JMLR 2013)

Whatever column sampling is used to select c columns, there exists a bad case $\mathbf{K}$ *such that*

$$\frac{\|\mathbf{K}-\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}\|_F^2}{\|\mathbf{K}-\mathbf{K}_k\|_F^2} \geq \Omega\left(1+\frac{nk}{c^2}\right).$$

Better Sampling Algorithms?

- We hope $\frac{\|\mathbf{K}-\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}\|_F}{\|\mathbf{K}-\mathbf{K}_k\|_F}$ will be very small if the column sampling algorithm is good enough.
- But it cannot be arbitrarily small.
- Lower Error Bound

Theorem (Wang & Zhang, JMLR 2013)

Whatever column sampling is used to select c columns, there exists a bad case $\mathbf{K}$ *such that*

$$\frac{\|\mathbf{K}-\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}\|_F^2}{\|\mathbf{K}-\mathbf{K}_k\|_F^2} \geq \Omega\left(1+\frac{nk}{c^2}\right).$$

Different Types of Low-Rank Approximation?

- The *Ensemble Nyström Method* [Kumar *et al.*, JMLR 2012]:

$$\mathbf{K} \approx \sum_{i=1}^{t} \frac{1}{t} \mathbf{C}^{(i)} \mathbf{W}^{(i)\dagger} \mathbf{C}^{(i)T}$$

- It does not improve the lower error bound.
- Lower Error Bound

Theorem (Wang & Zhang, JMLR 2013)

Whatever column sampling is used to select c columns, there exists a bad case $\mathbf{K}$ *such that*

$$\frac{\|\mathbf{K} - \sum_{i=1}^{t} \frac{1}{t} \mathbf{C}^{(i)} \mathbf{W}^{(i)\dagger} \mathbf{C}^{(i)T}\|_F^2}{\|\mathbf{K} - \mathbf{K}_k\|_F^2} \geq \Omega\left(1 + \frac{nk}{c^2}\right).$$

Different Types of Low-Rank Approximation?

- The *Ensemble Nyström Method* [Kumar *et al.*, JMLR 2012]:

$$\mathbf{K} \approx \sum_{i=1}^{t} \frac{1}{t} \mathbf{C}^{(i)} \mathbf{W}^{(i)\dagger} \mathbf{C}^{(i)T}$$

- It does not improve the lower error bound.
- Lower Error Bound

Theorem (Wang & Zhang, JMLR 2013)

Whatever column sampling is used to select c columns, there exists a bad case $\mathbf{K}$ *such that*

$$\frac{\|\mathbf{K} - \sum_{i=1}^{t} \frac{1}{t} \mathbf{C}^{(i)} \mathbf{W}^{(i)\dagger} \mathbf{C}^{(i)T}\|_F^2}{\|\mathbf{K} - \mathbf{K}_k\|_F^2} \geq \Omega\left(1 + \frac{nk}{c^2}\right).$$

Different Types of Low-Rank Approximation?

- The *Ensemble Nyström Method* [Kumar *et al.*, JMLR 2012]:

$$\mathbf{K} \approx \sum_{i=1}^{t} \frac{1}{t} \mathbf{C}^{(i)} \mathbf{W}^{(i)\dagger} \mathbf{C}^{(i)T}$$

- It does not improve the lower error bound.
- Lower Error Bound

Theorem (Wang & Zhang, JMLR 2013)

Whatever column sampling is used to select c columns, there exists a bad case $\mathbf{K}$ *such that*

$$\frac{\left\|\mathbf{K} - \sum_{i=1}^{t} \frac{1}{t} \mathbf{C}^{(i)} \mathbf{W}^{(i)\dagger} \mathbf{C}^{(i)T}\right\|_F^2}{\|\mathbf{K} - \mathbf{K}_k\|_F^2} \geq \Omega\left(1 + \frac{nk}{c^2}\right).$$

Different Types of Low-Rank Approximation?

- The *Modified Nyström Method* [Wang & Zhang, JMLR 2013]:

$$\mathbf{K} \approx \mathbf{C}\Big(\underbrace{\mathbf{C}^{\dagger}\mathbf{K}(\mathbf{C}^{\dagger})^{T}}_{c\times c}\Big)\mathbf{C}^{T}.$$

Theorem (Wang & Zhang, JMLR 2013)

Using a column sampling algorithm, the error incurred by the modified Nyström method satisfies

$$\mathbb{E}\frac{\|\mathbf{K}-\mathbf{C}(\mathbf{C}^{\dagger}\mathbf{K}(\mathbf{C}^{\dagger})^{T})\mathbf{C}^{T}\|_F^2}{\|\mathbf{K}-\mathbf{K}_k\|_F^2} \leq 1+\sqrt{\frac{k}{c}}.$$

Different Types of Low-Rank Approximation?

- The *Modified Nyström Method* [Wang & Zhang, JMLR 2013]:

$$\mathbf{K} \approx \mathbf{C}\Big(\underbrace{\mathbf{C}^{\dagger}\mathbf{K}(\mathbf{C}^{\dagger})^{T}}_{c\times c}\Big)\mathbf{C}^{T}.$$

Theorem (Wang & Zhang, JMLR 2013)

Using a column sampling algorithm, the error incurred by the modified Nyström method satisfies

$$\mathbb{E}\frac{\left\|\mathbf{K}-\mathbf{C}\left(\mathbf{C}^{\dagger}\mathbf{K}(\mathbf{C}^{\dagger})^{T}\right)\mathbf{C}^{T}\right\|_{F}^{2}}{\|\mathbf{K}-\mathbf{K}_{k}\|_{F}^{2}} \leq 1+\sqrt{\frac{k}{c}}.$$

Outline

Notation

- Define $T_{\mathrm{SVD}}(n^3)$: time of the full SVD (or eigenvalue decomposition, matrix inverse, etc.) of an $n \times n$ matrix
- Define $T_{\mathrm{Multiply}}(n^3)$: time of multiplying two $n \times n$ matrices
- They are both $\mathcal{O}(n^3)$, but very different in practice.
- Large scale matrix multiplication is not a challenge in real-world applications.

Notation

- Define $T_{\mathrm{SVD}}(n^3)$: time of the full SVD (or eigenvalue decomposition, matrix inverse, etc.) of an $n \times n$ matrix
- Define $T_{\mathrm{Multiply}}(n^3)$: time of multiplying two $n \times n$ matrices
- They are both $\mathcal{O}(n^3)$, but very different in practice.
- Large scale matrix multiplication is not a challenge in real-world applications.

Comparisons between the Two Methods

- **The Standard Nyström Method: fast.**
  It costs only $T_{\mathrm{SVD}}(c^3)$ time to compute the intersection matrix $\mathbf{U}^{\mathrm{nys}} = \mathbf{W}^{\dagger}$.
- **The Modified Nyström Method: slow.**
  It costs $T_{\mathrm{SVD}}(nc^2) + T_{\mathrm{Multiply}}(n^2c)$ time to compute the intersection matrix $\mathbf{U}^{\mathrm{mod}} = \mathbf{C}^{\dagger}\mathbf{K}(\mathbf{C}^{\dagger})^T$ naively.

Comparisons between the Two Methods

- **The Standard Nyström Method: inaccurate.**
  It cannot attain $1+\epsilon$ Frobenius relative-error bound unless

$$c \geq \sqrt{nk/\epsilon}$$

  columns are selected, whatever column selection algorithm is used. (Due to its lower error bound.)

- **The Modified Nyström Method: accurate.**
  Some adaptive sampling based algorithms attain $1+\epsilon$ Frobenius relative-error bound when

$$c = \mathcal{O}(k/\epsilon^2).$$

  (c is the smaller the better.)

Comparisons between the Two Methods

- **The Standard Nyström Method: inaccurate.**
  It cannot attain $1+\epsilon$ Frobenius relative-error bound unless

  $$c \geq \sqrt{nk/\epsilon}$$

  columns are selected, whatever column selection algorithm is used. (Due to its lower error bound.)

- **The Modified Nyström Method: accurate.**
  Some adaptive sampling based algorithms attain $1+\epsilon$ Frobenius relative-error bound when

  $$c = \mathcal{O}(k/\epsilon^2).$$

  (c is the smaller the better.)

Fast Computation of the Intersection Matrix

- Naively computing the intersection matrix

$$\mathbf{U} = \mathbf{C}^{\dagger}\mathbf{K}(\mathbf{C}^{\dagger})^{T}$$

costs $T_{\mathrm{SVD}}(nc^2) + T_{\mathrm{Multiply}}(n^2c)$ time.

- How to speedup?

Fast Computation of the Intersection Matrix

- Naively computing the intersection matrix

$$\mathbf{U} = \mathbf{C}^{\dagger}\mathbf{K}(\mathbf{C}^{\dagger})^{T}$$

costs $T_{\mathrm{SVD}}(nc^2) + T_{\mathrm{Multiply}}(n^2c)$ time.

- How to speedup?

Fast Computation of the Intersection Matrix

- Moore-Penrose inverse of partitioned matrices can be expanded!
- Let $\mathbf{P}$ be a permutation matrix, and let

$$\mathbf{PC} = \begin{bmatrix} \mathbf{W} \\ \mathbf{K}_{21} \end{bmatrix}.$$

If $\mathbf{W}$ is nonsingular, let $\mathbf{S} = \mathbf{K}_{21}\mathbf{W}^{-1}$, the Moore-Penrose inverse of $\mathbf{C}$ can be written as

$$\mathbf{C}^{\dagger} = \mathbf{W}^{-1}\left(\mathbf{I}_c + \mathbf{S}^T\mathbf{S}\right)^{-1}\begin{bmatrix} \mathbf{I}_c & \mathbf{S}^T \end{bmatrix}\mathbf{P},$$

Fast Computation of the Intersection Matrix

Compute the intersection matrix by

$$\mathbf{U} = \mathbf{C}^{\dagger}\mathbf{K}(\mathbf{C}^{\dagger})^{T} = \mathbf{T}_1\left(\mathbf{W} + \mathbf{T}_2 + \mathbf{T}_2^{T} + \mathbf{T}_3\right)\mathbf{T}_1^{T},$$

where the intermediate matrices are computed by

$$\mathbf{T}_0 = \mathbf{K}_{21}^{T}\mathbf{K}_{21}, \quad \mathbf{T}_1 = \mathbf{W}^{-1}\left(\mathbf{I}_c + \mathbf{W}^{-1}\mathbf{T}_2\right)^{-1},$$
$$\mathbf{T}_2 = \mathbf{T}_0\mathbf{W}^{-1}, \quad \mathbf{T}_3 = \mathbf{W}^{-1}\left(\mathbf{K}_{21}^{T}\mathbf{K}_{22}\mathbf{K}_{21}\right)\mathbf{W}^{-1}.$$

The four intermediate matrices are all of size $c \times c$, and the matrix inverse operations are on $c \times c$ small matrices.

Fast Computation of the Intersection Matrix

- In this way, it costs only $T_{\mathrm{SVD}}(c^3) + T_{\mathrm{Multiply}}\left((n-c)^2 c\right)$ time.
- The naive approach cost $T_{\mathrm{SVD}}(nc^2) + T_{\mathrm{Multiply}}(n^2 c)$ time.
- Our method works only if **W** is nonsingular.
- If **K** is Gaussian RBF kernel matrix, and if the selected c data are distinct points, then **W** is nonsingular.
- If **K** is linear kernel matrix, **W** is usually singular.

Fast Computation of the Intersection Matrix

- In this way, it costs only $T_{\text{SVD}}(c^3) + T_{\text{Multiply}}\left((n-c)^2 c\right)$ time.
- The naive approach cost $T_{\text{SVD}}(nc^2) + T_{\text{Multiply}}(n^2 c)$ time.
- Our method works only if **W** is nonsingular.
- If **K** is Gaussian RBF kernel matrix, and if the selected c data are distinct points, then **W** is nonsingular.
- If **K** is linear kernel matrix, **W** is usually singular.

Fast Computation of the Intersection Matrix

- In this way, it costs only $T_{\text{SVD}}(c^3) + T_{\text{Multiply}}\left((n-c)^2 c\right)$ time.
- The naive approach cost $T_{\text{SVD}}(nc^2) + T_{\text{Multiply}}(n^2 c)$ time.
- Our method works only if **W** is nonsingular.
- If **K** is Gaussian RBF kernel matrix, and if the selected c data are distinct points, then **W** is nonsingular.
- If **K** is linear kernel matrix, **W** is usually singular.

Fast Computation of the Intersection Matrix

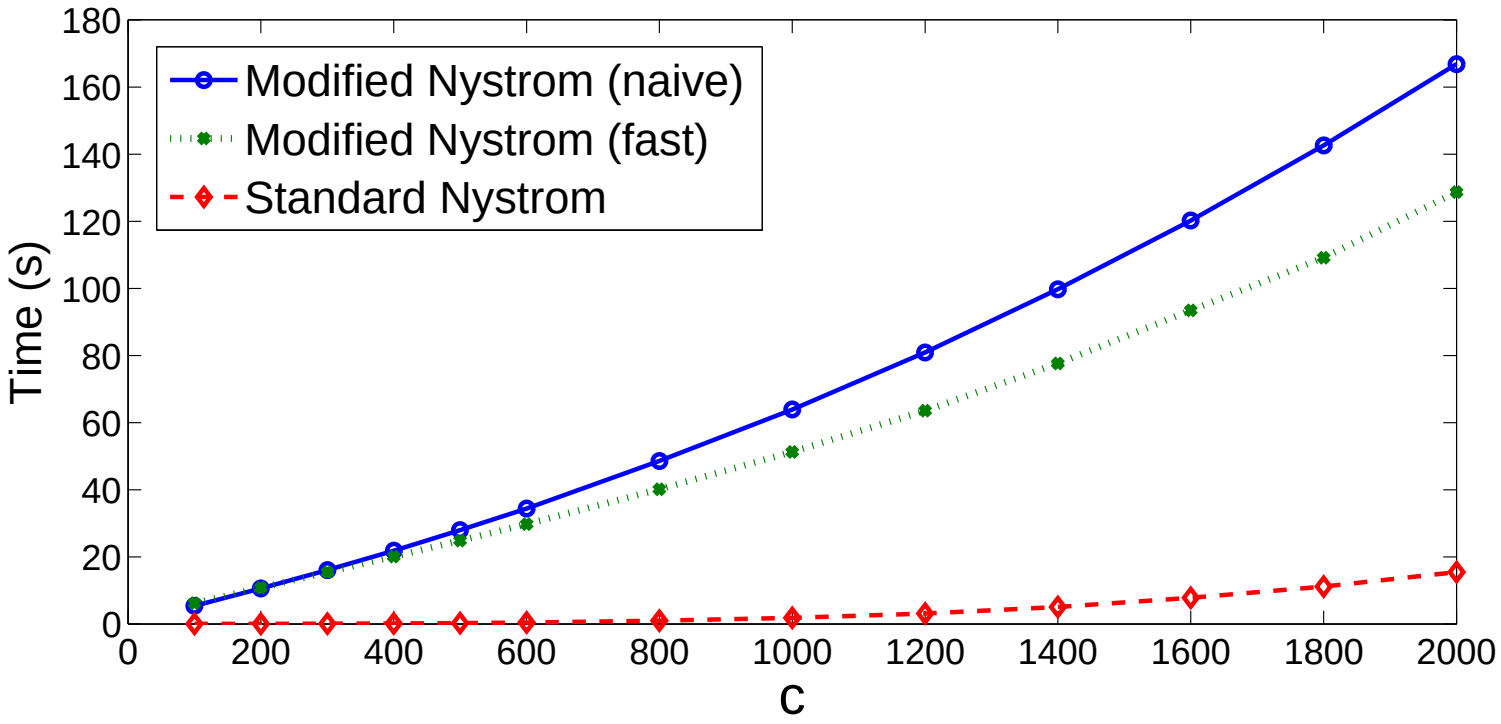
- In this way, it costs only $T_{\text{SVD}}(c^3) + T_{\text{Multiply}}\left((n-c)^2 c\right)$ time.
- The naive approach cost $T_{\text{SVD}}(nc^2) + T_{\text{Multiply}}(n^2 c)$ time.
- Our method works only if **W** is nonsingular.
- If **K** is Gaussian RBF kernel matrix, and if the selected c data are distinct points, then **W** is nonsingular.
- If **K** is linear kernel matrix, **W** is usually singular.

Fast Computation of the Intersection Matrix

- In this way, it costs only $T_{\mathrm{SVD}}(c^3) + T_{\mathrm{Multiply}}\left((n-c)^2 c\right)$ time.
- The naive approach cost $T_{\mathrm{SVD}}(nc^2) + T_{\mathrm{Multiply}}(n^2 c)$ time.
- Our method works only if **W** is nonsingular.
- If **K** is Gaussian RBF kernel matrix, and if the selected c data are distinct points, then **W** is nonsingular.
- If **K** is linear kernel matrix, **W** is usually singular.

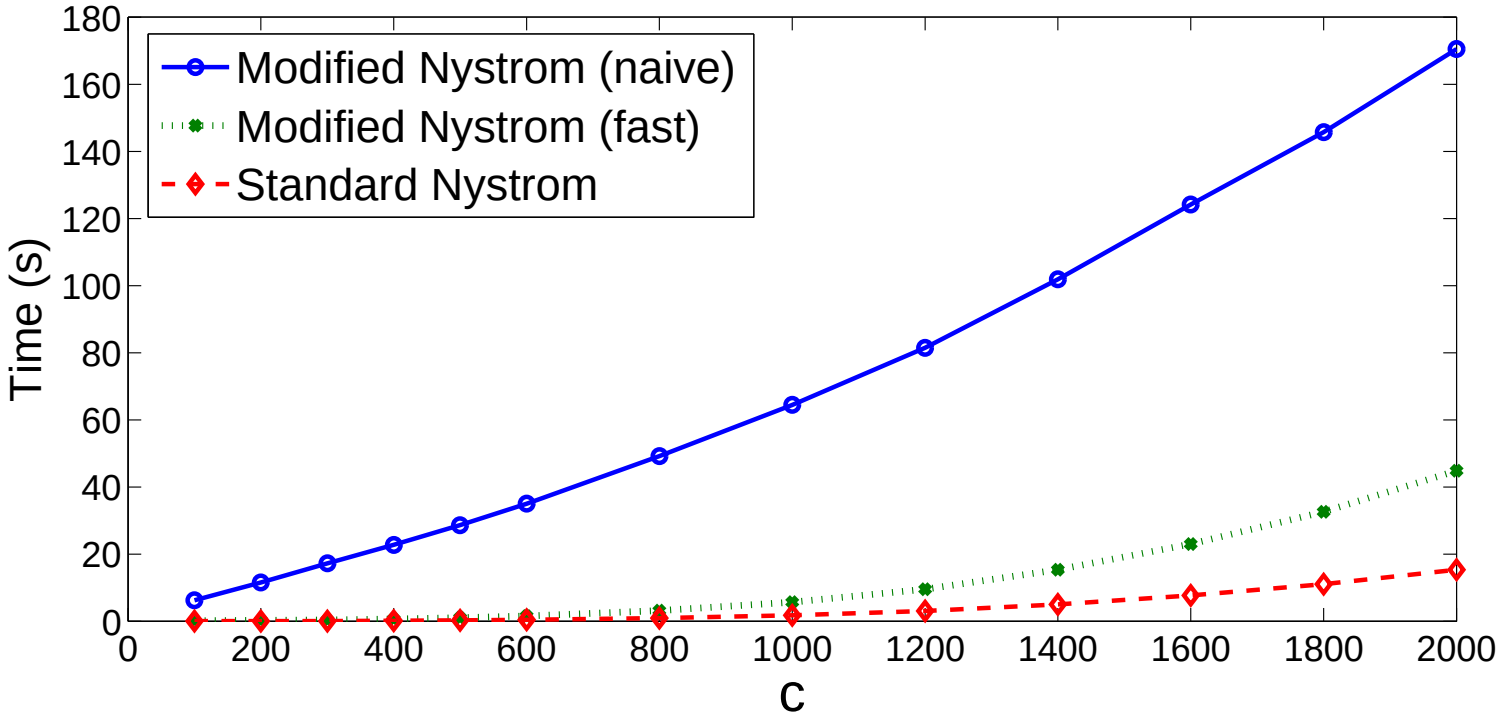
Fast Computation of the Intersection Matrix

Results on an $15,000 \times 15,000$ (dense) RBF kernel matrix.



Fast Computation of the Intersection Matrix

Results on an $15,000 \times 15,000$ sparse RBF kernel matrix with 1% entries nonzero.



Efficient Column Sampling Algorithm

The uniform+adaptive^2 algorithm:

1. **Uniform Sampling.** Uniformly sample
$$c_1 = 8.7\mu k \log\left(\sqrt{5}k\right)$$
columns of $\mathbf{K}$ without replacement to construct $\mathbf{C}_1$;
2. **Adaptive Sampling.** Sample
$$c_2 = 10k\epsilon^{-1}$$
columns of $\mathbf{K}$ to construct $\mathbf{C}_2$ using adaptive sampling algorithm according to the residual $\mathbf{K} - \mathcal{P}_{\mathbf{C}_1}\mathbf{K}$;
3. **Adaptive Sampling.** Sample
$$c_3 = 2\epsilon^{-1}(c_1 + c_2)$$
columns of $\mathbf{K}$ to construct $\mathbf{C}_3$ using adaptive sampling algorithm according to the residual $\mathbf{K} - \mathcal{P}_{[\mathbf{C}_1,\, \mathbf{C}_2]}\mathbf{K}$;
4. **Return** $\mathbf{C} = [\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3]$.

Efficient Column Sampling Algorithm

The uniform+adaptive^2 algorithm:

1. **Uniform Sampling.** Uniformly sample

$$c_1 = 8.7\mu k \log\left(\sqrt{5}k\right)$$

columns of $\mathbf{K}$ without replacement to construct $\mathbf{C}_1$;

2. **Adaptive Sampling.** Sample

$$c_2 = 10k\epsilon^{-1}$$

columns of $\mathbf{K}$ to construct $\mathbf{C}_2$ using adaptive sampling algorithm according to the residual $\mathbf{K} - \mathcal{P}_{\mathbf{C}_1}\mathbf{K}$;

3. **Adaptive Sampling.** Sample

$$c_3 = 2\epsilon^{-1}(c_1 + c_2)$$

columns of $\mathbf{K}$ to construct $\mathbf{C}_3$ using adaptive sampling algorithm according to the residual $\mathbf{K} - \mathcal{P}_{[\mathbf{C}_1,\, \mathbf{C}_2]}\mathbf{K}$;

4. **Return** $\mathbf{C} = [\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3]$.

Efficient Column Sampling Algorithm

The uniform+adaptive^2 algorithm:

1. **Uniform Sampling.** Uniformly sample
$$c_1 = 8.7\mu k \log\left(\sqrt{5}k\right)$$
columns of $\mathbf{K}$ without replacement to construct $\mathbf{C}_1$;
2. **Adaptive Sampling.** Sample
$$c_2 = 10k\epsilon^{-1}$$
columns of $\mathbf{K}$ to construct $\mathbf{C}_2$ using adaptive sampling algorithm according to the residual $\mathbf{K} - \mathcal{P}_{\mathbf{C}_1}\mathbf{K}$;
3. **Adaptive Sampling.** Sample
$$c_3 = 2\epsilon^{-1}(c_1 + c_2)$$
columns of $\mathbf{K}$ to construct $\mathbf{C}_3$ using adaptive sampling algorithm according to the residual $\mathbf{K} - \mathcal{P}_{[\mathbf{C}_1,\, \mathbf{C}_2]}\mathbf{K}$;
4. **Return** $\mathbf{C} = [\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3]$.

Efficient Column Sampling Algorithm

Theorem

The uniform+adaptive^2 algorithm cost time

$$T_{\mathrm{SVD}}(nc^2\epsilon^2) + T_{\mathrm{Multiply}}(n^2c\epsilon).$$

Theorem

By sampling

$$c = \mathcal{O}(k\epsilon^{-2} + \mu_k\epsilon^{-1}k\log k)$$

columns using the uniform+adaptive^2 algorithm,

$$\left\|\mathbf{K} - \mathbf{C}(\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T)\mathbf{C}^T\right\|_F \leq (1+\epsilon)\left\|\mathbf{K} - \mathbf{K}_k\right\|_F$$

holds with high probability.

Efficient Column Sampling Algorithm

Theorem

The uniform+adaptive^2 algorithm cost time

$$T_{\mathrm{SVD}}(nc^2\epsilon^2) + T_{\mathrm{Multiply}}(n^2c\epsilon).$$

Theorem

By sampling

$$c = \mathcal{O}(k\epsilon^{-2} + \mu_k\epsilon^{-1}k\log k)$$

columns using the uniform+adaptive^2 algorithm,

$$\left\|\mathbf{K} - \mathbf{C}(\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T)\mathbf{C}^T\right\|_F \leq (1+\epsilon)\left\|\mathbf{K} - \mathbf{K}_k\right\|_F$$

holds with high probability.

Theoretical Justification

Theorem (Exact Recovery.)

For the symmetric matrix $\mathbf{K}$ *defined previously, the following three statements are equivalent:*

1. $\operatorname{rank}(\mathbf{W}) = \operatorname{rank}(\mathbf{K})$,
2. $\mathbf{K} = \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}$,
   (i.e., the standard Nyström method is exact)
3. $\mathbf{K} = \mathbf{C}\left(\mathbf{C}^{\dagger}\mathbf{K}(\mathbf{C}^{\dagger})^{T}\right)\mathbf{C}^{T}$,
   (i.e., the modified Nyström method is exact)

Theoretical Justification

Theorem (Exact Recovery.)

It holds in general that

$$\|\mathbf{K} - \mathbf{C}(\mathbf{C}^{\dagger}\mathbf{K}(\mathbf{C}^{\dagger})^T)\mathbf{C}^T\|_F \leq \|\mathbf{K} - \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T\|_F.$$

It is because

$$\mathbf{U} = \mathbf{C}^{\dagger}\mathbf{K}(\mathbf{C}^{\dagger})^T$$

is the solution to the problem

$$\min_{\mathbf{U}} \|\mathbf{K} - \mathbf{C}\mathbf{U}\mathbf{C}^T\|_F.$$

Lower Error Bounds

Lower error bound of the modified Nyström method

Theorem

Whatever column sampling is used to select c columns, there exists a bad case $\mathbf{K}$ *such that*

$$\frac{\left\|\mathbf{K}-\mathbf{C}\left(\mathbf{C}^{\dagger}\mathbf{K}(\mathbf{C}^{\dagger})^{T}\right)\mathbf{C}^{T}\right\|_{F}^{2}}{\|\mathbf{K}-\mathbf{K}_{k}\|_{F}^{2}} \geq \frac{n-c}{n-k}\left(1+\frac{2k}{c}\right).$$

Lower Error Bounds

- The modified Nyström method has a strong resemblance with the column selection problem.
- Lower error bound of the column selection problem

Theorem (Boutsidis *et al.*, FOCS 2011)

Whatever column sampling is used to select c columns, there exists a bad case $\mathbf{A} \in \mathbb{R}^{m \times n}$ such that

$$\frac{\left\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\right\|_F^2}{\left\|\mathbf{A} - \mathbf{A}_k\right\|_F^2} \geq \frac{n-c}{n-k}\left(1 + \frac{k}{c}\right).$$

- This lower bound is tight, because it is attained by a column selection algorithm of [Guruswami & Sinop, SODA 2012].

Lower Error Bounds

- The modified Nyström method has a strong resemblance with the column selection problem.
- Lower error bound of the column selection problem

Theorem (Boutsidis *et al.*, FOCS 2011)

Whatever column sampling is used to select c columns, there exists a bad case $\mathbf{A} \in \mathbb{R}^{m \times n}$ such that

$$\frac{\left\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\right\|_F^2}{\left\|\mathbf{A} - \mathbf{A}_k\right\|_F^2} \geq \frac{n-c}{n-k}\left(1 + \frac{k}{c}\right).$$

- This lower bound is tight, because it is attained by a column selection algorithm of [Guruswami & Sinop, SODA 2012].

Lower Error Bounds

- Their lower error bounds are very similar:

$$\frac{\left\|\mathbf{K}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{K}(\mathbf{C}^{\dagger})^{T}\mathbf{C}^{T}\right\|_F^2}{\|\mathbf{K}-\mathbf{K}_k\|_F^2} \geq \frac{n-c}{n-k}\left(1+\frac{2k}{c}\right)\|\mathbf{K}-\mathbf{K}_k\|_F^2,$$

$$\frac{\left\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\right\|_F^2}{\|\mathbf{A}-\mathbf{A}_k\|_F^2} \geq \frac{n-c}{n-k}\left(1+\frac{k}{c}\right)\|\mathbf{A}-\mathbf{A}_k\|_F^2.$$

- It is a reasonable conjecture that the lower bound of the modified Nyström method is also tight! (an open problem).

Lower Error Bounds

- Their lower error bounds are very similar:

$$\frac{\left\|\mathbf{K}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{K}(\mathbf{C}^{\dagger})^{T}\mathbf{C}^{T}\right\|_F^2}{\|\mathbf{K}-\mathbf{K}_k\|_F^2} \geq \frac{n-c}{n-k}\left(1+\frac{2k}{c}\right)\|\mathbf{K}-\mathbf{K}_k\|_F^2,$$

$$\frac{\left\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\right\|_F^2}{\|\mathbf{A}-\mathbf{A}_k\|_F^2} \geq \frac{n-c}{n-k}\left(1+\frac{k}{c}\right)\|\mathbf{A}-\mathbf{A}_k\|_F^2.$$

- It is a reasonable conjecture that the lower bound of the modified Nyström method is also tight! (an open problem).

Open Problem

- **Lower error bound:**
  at least $c \geq 2k/\epsilon$ to attain $1+\epsilon$ relative-error bound.
- **An upper error bound** [Wang & Zhang, JMLR 2013]:
  samples $c = \frac{k}{\epsilon^2}(1 + o(1))$ columns to attain $1+\epsilon$ relative-error bound.
- The gap implies better column sampling algorithms for the modified Nyström method.

Open Problem

- **Lower error bound:**
  at least $c \geq 2k/\epsilon$ to attain $1+\epsilon$ relative-error bound.
- **An upper error bound** [Wang & Zhang, JMLR 2013]:
  samples $c = \frac{k}{\epsilon^2}(1 + o(1))$ columns to attain $1+\epsilon$ relative-error bound.
- The gap implies better column sampling algorithms for the modified Nyström method.

Open Problem

- **Lower error bound:**
  at least $c \geq 2k/\epsilon$ to attain $1+\epsilon$ relative-error bound.
- **An upper error bound** [Wang & Zhang, JMLR 2013]:
  samples $c = \frac{k}{\epsilon^2}(1 + o(1))$ columns to attain $1+\epsilon$ relative-error bound.
- The gap implies better column sampling algorithms for the modified Nyström method.

Reference


A. Deshpande, L. Rademacher, S. Vempala, and G.Wang: *Matrix approximation and projective clustering via volume sampling.* Theory of Computing, 2006.


S. Kumar, M. Mohri, and A. Talwalkar: *Sampling methods for the Nyström method.* JMLR, 2012.


S. Wang and Z. Zhang: *Improving CUR matrix decomposition and the Nyström approximation via adaptive sampling.* JMLR, 2013.


C. Boutsidis, P. Drineas, and M. Magdon-Ismail: *Near optimal column-based matrix reconstruction*. In FOCS, 2011.


V. Guruswami and A. K. Sinop: *Optimal column based low-rank matrix reconstruction*. In SODA, 2012.