Approximate Principal Components Analysis of Large Data Sets

Daniel J. McDonald

Department of Statistics
Indiana University

mypage.iu.edu/∼dajmcdon

April 27, 2016

Approximation-Regularization for Analysis of Large Data Sets

Daniel J. McDonald

Department of Statistics
Indiana University
mypage.iu.edu/∼dajmcdon

April 27, 2016

Lesson of the talk

Many statistical methods use (perhaps implicitly) a singular value decomposition (SVD).

The SVD is computationally expensive.

We want to understand the statistical properties of some approximations which speed up computation and save storage.

Spoiler: sometimes approximations actually improve the statistical properties

Core techniques

Suppose we have a matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ and vector $Y \in \mathbb{R}^n$

Least squares:

Finding

$$\widehat{\beta} = \underset{\beta}{\operatorname{argmin}} \|\mathbf{X}\beta - Y\|_2^2$$

Principal Components Analysis (PCA):

(Or graph Laplacian or diffusion map or..)

Finding $\mathbf{U}$, $\mathbf{V}$ orthogonal and Λ diagonal such that

$$\mathbf{X} = \widetilde{\mathbf{X}} - \overline{\mathbf{X}} = \mathbf{U}\Lambda\mathbf{V}^\top$$

where

$$\overline{\mathbf{X}} = \mathbf{1}\mathbf{1}^\top\widetilde{\mathbf{X}}$$

Core techniques

If **X** fits into RAM, there exist excellent algorithms in LAPACK that are

- Double precision
- Very stable
- cubic complexity, $O(\min\{n,p\}^3)$, with small constants
- require extensive random access to matrix

There is a lot of interest in finding and analyzing techniques that extend these approaches to large(r) problems

Out-of-core techniques

Many techniques focus on randomized compression

(This is sometimes known as sketching)

Least squares:

1. Rokhlin, Tygert, “A fast randomized algorithm for overdetermined linear least-squares regression” (2008).
2. Drineas, Mahoney, et al., “Faster least squares approximation” (2011).
3. Woodruff “Sketching as a Tool for Numerical Linear Algebra” (2013).
4. Homrighausen, McDonald, “Preconditioned least squares” (2016).

Spectral Decomposition:

1. Halko, et al., “Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions” (2011).
2. Gittens, Mahoney, “Revisiting the Nystrom Method for Improved Large-Scale Machine Learning” (2013).
3. Pourkamali, “Memory and Computation Efficient PCA via Very Sparse Random Projections” (2014).
4. Homrighausen, McDonald, “On the Nyström and Column-Sampling Methods for the Approximate PCA of Large Data Sets” (2015).

Out-of-core techniques

Many techniques focus on randomized compression

(This is sometimes known as sketching)

Least squares:

1. Rokhlin, Tygert, “A fast randomized algorithm for overdetermined linear least-squares regression” (2008).
2. Drineas, Mahoney, et al., “Faster least squares approximation” (2011).
3. Woodruff “Sketching as a Tool for Numerical Linear Algebra” (2013).
4. Homrighausen, McDonald, “Preconditioned least squares” (2016).

Spectral Decomposition:

1. Halko, et al., “Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions” (2011).
2. Gittens, Mahoney, “Revisiting the Nystrom Method for Improved Large-Scale Machine Learning” (2013).
3. Pourkamali, “Memory and Computation Efficient PCA via Very Sparse Random Projections” (2014).
4. Homrighausen, McDonald, “On the Nyström and Column-Sampling Methods for the Approximate PCA of Large Data Sets” (2015).

Compression

Basic Idea:

- Choose some matrix $Q \in \mathbb{R}^{q \times n}$ or $\mathbb{R}^{p \times q}$.
- Use $Q\mathbf{X}$ or $\mathbf{X}Q$ and QY instead

Finding $Q\mathbf{X}$ for arbitrary Q and $\mathbf{X}$ takes $O(qnp)$ computations

This can be expensive,

To get this approach to work, we need some structure on Q

The Q matrix

- Gaussian

  (Well behaved distribution and eas(ier) theory. Dense matrix)

- Fast Johnson-Lindenstrauss Methods
- Randomized Hadamard (or Fourier) transformation

  (Allows for $O(np\log(p))$ computations.)

- $Q = \pi\tau$ for π a permutation of I and $\tau = [I_q\ 0]$.

  ($\mathbf{X}Q$ means "only read q columns")

- Sparse Bernoulli

$$Q_{ij} \overset{i.i.d.}{\sim} \begin{cases} 1 & \text{with probability } 1/(2s) \\ 0 & \text{with probability } 1 - 1/s \\ -1 & \text{with probability } 1/(2s) \end{cases}$$

This means $Q\mathbf{X}$ takes $O\left(\frac{qnp}{s}\right)$ "computations" on average

The Q matrix

- Gaussian

  (Well behaved distribution and eas(ier) theory. Dense matrix)

- Fast Johnson-Lindenstrauss Methods
- Randomized Hadamard (or Fourier) transformation

  (Allows for $O(np\log(p))$ computations.)

- $Q = \pi\tau$ for π a permutation of I and $\tau = [I_q\ 0]$.

  ($\mathbf{X}Q$ means “only read q columns”)

- Sparse Bernoulli

$$Q_{ij} \overset{i.i.d.}{\sim} \begin{cases} 1 & \text{with probability } 1/(2s) \\ 0 & \text{with probability } 1 - 1/s \\ -1 & \text{with probability } 1/(2s) \end{cases}$$

This means $Q\mathbf{X}$ takes $O\left(\frac{qnp}{s}\right)$ “computations” on average

Typical results

The general philosophy: Find an approximation that is as close as possible to the solution of the original problem

PCA:

A typical result would be to find an approximate $\tilde{V}$ such that

$$\text{angle}(V, \tilde{V}) \leq \sqrt{\frac{p}{n}} \left(\frac{1}{\text{spectral gap}} \right)$$

(This is the same order of convergence as PCA [Homrighausen, McDonald (2015)])

Typical results

The general philosophy: Find an approximation that is as close as possible to the solution of the original problem

Least Squares:

A typical result would be to find an $\tilde{\beta}$ such that

$$\|\mathbf{X}\tilde{\beta} - Y\|_2^2 \leq (1 + \epsilon) \left(\min_{\beta} \|\mathbf{X}\beta - Y\|_2^2 \right)$$

Here, $\tilde{\beta}$ should be 'easier' to compute than $\widehat{\beta}$

Collaborators & Grant Support

Collaborator:

Darren Homrighausen, Colorado State

Assistant Professor, Department of Statistics

Grant Support:

- NSF Grant DMS–14-07543
- INET Grant INO—14-00020

Typical result: Principal components analysis

Notation

- Write (again) the SVD of **X** as

$$\mathbf{X} = \mathbf{U}\Lambda\mathbf{V}^\top,$$

- For general rank *r* matrices **A** we write

$$\mathbf{A} = U(\mathbf{A})\Lambda(\mathbf{A})V(\mathbf{A})^\top$$

  where

$$\Lambda(\mathbf{A}) = \mathrm{diag}(\lambda_1(\mathbf{A}), \ldots, \lambda_r(\mathbf{A}))$$

- For some matrix **A**, we use $\mathbf{A}_d$ to be the first *d* columns of **A**.

Approximations

Can't use **X**

What about approximating it?

Focus on two methods of "approximate SVD"

1. Nyström extension
2. Column sampling

Intuition (not the algorithm)

- Essentially, let

$$\mathbf{S} = \mathbf{X}^\top \mathbf{X} \in \mathbb{R}^{n\times n} \qquad \text{and} \qquad \mathbf{R} = \mathbf{X}\mathbf{X}^\top \in \mathbb{R}^{p\times p}$$

- Both of these are symmetric, positive semi-definite
- Randomly choose q entries in $\{1,\ldots,n\}$ or $\{1,\ldots,p\}$
- Then partition the matrix so the selected portion is $\mathbf{S}_{11}$ or $\mathbf{R}_{11}$

$$\mathbf{S} = \mathbf{V}\Lambda^2\mathbf{V}^\top = \begin{bmatrix}\mathbf{S}_{11} & \mathbf{S}_{12}\\ \mathbf{S}_{21} & \mathbf{S}_{22}\end{bmatrix} \quad \mathbf{R} = \mathbf{U}\Lambda^2\mathbf{U}^\top = \begin{bmatrix}\mathbf{R}_{11} & \mathbf{R}_{12}\\ \mathbf{R}_{21} & \mathbf{R}_{22}\end{bmatrix}$$

Approximating things we don't care about

If we want to approximate $\mathbf{S}$ (or $\mathbf{R}$), we have for example

Nyström

$$\mathbf{S} \approx \begin{bmatrix} \mathbf{S}_{11} \\ \mathbf{S}_{21} \end{bmatrix} \mathbf{S}_{11}^{\dagger} \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} \end{bmatrix}$$

Column sampling

$$\mathbf{S} \approx U\left(\begin{bmatrix} \mathbf{S}_{11} \\ \mathbf{S}_{21} \end{bmatrix}\right) \Lambda\left(\begin{bmatrix} \mathbf{S}_{11} \\ \mathbf{S}_{21} \end{bmatrix}\right) U\left(\begin{bmatrix} \mathbf{S}_{11} \\ \mathbf{S}_{21} \end{bmatrix}\right)^{\top}$$

Previous theoretical results have focused on the accuracy of these approximations (and variants) for a fixed computational budget.

Approximating things we don't care about

If we want to approximate **S** (or **R**), we have for example

Nyström

$$\mathbf{S} \approx \begin{bmatrix} \mathbf{S}_{11} \\ \mathbf{S}_{21} \end{bmatrix} \mathbf{S}_{11}^{\dagger} \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} \end{bmatrix}$$

Column sampling

$$\mathbf{S} \approx U\left(\begin{bmatrix} \mathbf{S}_{11} \\ \mathbf{S}_{21} \end{bmatrix}\right) \Lambda\left(\begin{bmatrix} \mathbf{S}_{11} \\ \mathbf{S}_{21} \end{bmatrix}\right) U\left(\begin{bmatrix} \mathbf{S}_{11} \\ \mathbf{S}_{21} \end{bmatrix}\right)^{\top}$$

We don't care about these approximations. We don't want these matrices.

What we actually want...

We really want $\mathbf{U}$, $\mathbf{V}$, and Λ.

Or even better, the population analogues

Then we can get the principal components, principal coordinates, and the amount of variance explained.

It turns out that there are quite a few ways to use these two methods to get the things we want.

Let

$$L(\mathbf{S}) = \begin{bmatrix} \mathbf{S}_{11} \\ \mathbf{S}_{21} \end{bmatrix} \qquad L(\mathbf{R}) = \begin{bmatrix} \mathbf{R}_{11} \\ \mathbf{R}_{21} \end{bmatrix}$$

Lots of approximations

After some reasonable algebra…

Quantity of interest	Label	Approximations
$\mathbf{V}$	$\mathbf{V}_{nys}$ $\mathbf{V}_{cs}$	$L(\mathbf{S})\mathbf{V}(\mathbf{S}_{11})\Lambda(\mathbf{S}_{11})^{\dagger}$ $\mathbf{U}(L(\mathbf{S}))$
$\mathbf{U}$	$\mathbf{U}_{nys}$ $\mathbf{U}_{cs}$ $\widehat{\mathbf{U}}_{nys}$ $\widehat{\mathbf{U}}_{cs}$ $\widehat{\mathbf{U}}$	$L(\mathbf{R})\mathbf{V}(\mathbf{R}_{11})\Lambda(\mathbf{R}_{11})^{\dagger}$ $\mathbf{U}(L(\mathbf{R}))$ $\mathbf{X}\mathbf{V}_{nys}\Lambda_{nys}^{\dagger/2}$ $\mathbf{X}\mathbf{V}_{cs}\Lambda_{cs}^{\dagger/2}$ $\mathbf{U}(\mathbf{x}_1)$

Notes of interest

	Complexity:	
Method	Computational	Storage
Standard	$O(n^2p + pn^2)$	$O(np)$
Nyström	$O(nq^2 + q^3)$	$O(q^2)$ [$O(nq)$]
Column sampling	$O(qnp + qp^2)$	$O(pq)$

- Column sampling results in orthogonal **U** and **V**, Nyström doesn't

- $\widehat{\mathbf{U}} = U(\mathbf{x}_1)$ where $\mathbf{X} = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2\end{bmatrix}$ seems reasonable if we think that only the first l selected covariates matter (gets used frequently by new statistical methods)

So what do we use?

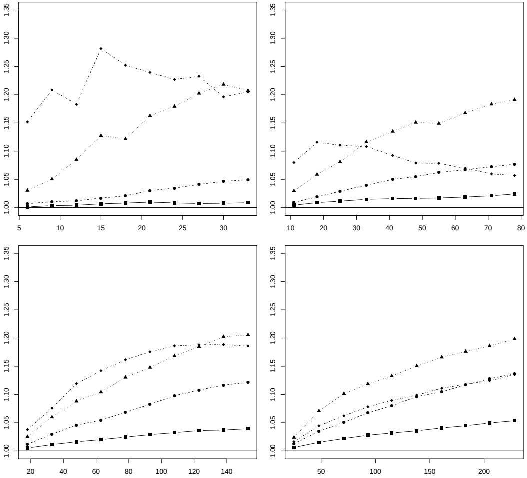
Well…

It depends. We did some theory which gives a way to calculate how far off your approximation might be. You could use these bounds if you like to use your data and make a choice.

Did some simulations too.

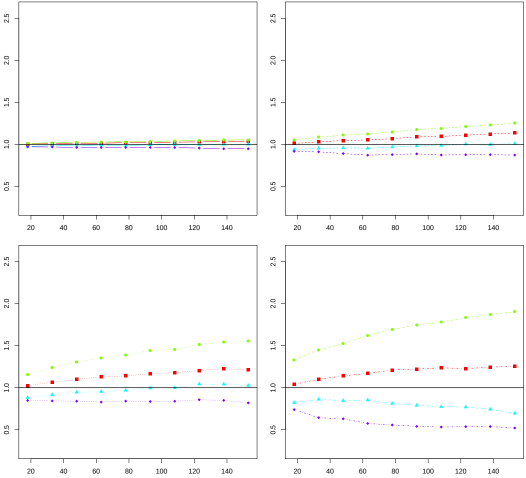
- Draw $\widetilde{X}_i \overset{iid}{\sim} \mathrm{N}_p(0, \Sigma)$ for $n = 5000,\ p = 3000$.
- 4 different conditions.
- For each method report: $\frac{\text{Projection error of estimator}}{\text{Projection error of baseline estimator}}$.

APPROXIMATIONS TO $\mathbf{V}$



- y-axis, relative performance
  Nyström to column sampling.
- $< 1 \rightarrow$ better; $> 1 \rightarrow$ worse.
- $d \in \{2,\ 5,\ 10,\ 15\}$.
- x-axis, approximation
  parameter $q \in [3d/2,\ 15d]$
- `Random`$_{0.001}$ (solid, square)
  `Random`$_{0.01}$ (dashed, circle)
  `Random`$_{0.1}$ (dotted, triangle)
  `Band` (dot-dash, diamond)
- d small, similar time
  d large, Nyström is 10–15%

APPROXIMATIONS TO U



- *y*-axis, performance relative to $\mathbf{U}_{cs}$
- *x*-axis, approximation parameter *q*
- $\mathbf{U}_{nys}$, $\widehat{\mathbf{U}}_{nys}$, $\widehat{\mathbf{U}}_{cs}$, $\widehat{\mathbf{U}}$
- $\texttt{Random}_{0.001}$
  $\texttt{Random}_{0.01}$
  $\texttt{Random}_{0.1}$
  $\texttt{Band}$
- $d = 10$

Conclusions (so far)

Approximate PCA

- For computing **V**, CS beats Nyström in terms of accuracy, but is much slower for similar choices of the approximation parameter and d large.
- For computing **U**, the naïve methods are bad, better to approximate **V** and multiply, so see above
- $\widehat{\mathbf{U}}$ really stinks. But it's most obvious. This also gets used frequently
- For more info, see the paper. Contains boring theory, more extensive simulations, and application to Enron email dataset.

Even better Approximate PCA

- We really want the population versions: top d eigenvectors of $\Sigma \ldots$

Atypical (better) result: Compressed regression

Estimation risk

Form a loss function $\ell : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}^+$

(For this talk, let's just specify $\ell(\widehat{\theta}, \theta) = \|\widehat{\theta} - \theta\|_2^2$)

The quality of an estimator is given by its risk

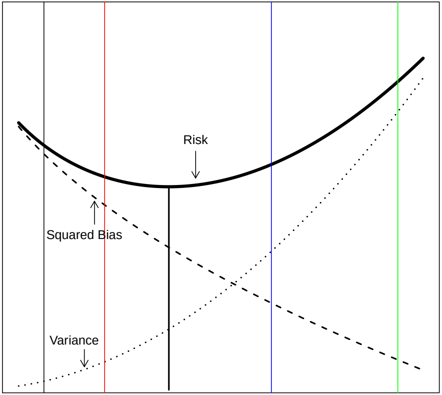
$$R(\widehat{\theta}) = \mathbb{E}\|\widehat{\theta} - \theta\|_2^2$$

This can be decomposed as:

$$\begin{aligned} R(\widehat{\theta}) &= \|\mathbb{E}[\widehat{\theta}] - \theta\|^2 + \mathbb{E}\|\widehat{\theta} - \mathbb{E}[\widehat{\theta}]\|^2 \\ &= \text{Bias}^2 + \text{Variance} \end{aligned}$$

There is a natural conservation between these quantities…

BIAS-VARIANCE TRADEOFF



Model Complexity →

Typical result examines the bias, atypical result also considers the variance

Fully compressed regression

Let $Q \in \mathbb{R}^{q \times n}$

Let's solve the fully compressed least squares problem

$$\widehat{\beta}_{FC} = \underset{\beta}{\operatorname{argmin}} \|Q\mathbf{X}\beta - QY\|_2^2$$

→ A common way to solve least squares problems that are:

- Very large or
- Poorly conditioned

The numerical/theoretical properties generally depend on Q, q

(Multiply quickly? Reduce dimension? Reduce Storage?)

Three approximations

Note:

$$\|Q(\mathbf{X}\beta - Y)\|_2^2 \propto \beta^\top \mathbf{X}^\top Q^\top Q \mathbf{X}\beta - 2\beta^\top \mathbf{X}^\top Q^\top Q Y$$

Full compression:

$$\widehat{\beta}_{FC} = (\mathbf{X}^\top Q^\top Q \mathbf{X})^{-1} \mathbf{X}^\top Q^\top Q Y$$

Partial compression:

$$\widehat{\beta}_{PC} = (\mathbf{X}^\top Q^\top Q \mathbf{X})^{-1} \mathbf{X}^\top Y$$

Convex combination compression:[1]

$$\widehat{\beta}_C = W\widehat{\alpha} \qquad W = [\, \widehat{\beta}_{FC},\ \widehat{\beta}_{PC} \,] \qquad \widehat{\alpha} = \underset{\alpha}{\operatorname{argmin}} \|W\alpha - Y\|_2^2$$

[1] see the work of Stephen Becker CU Boulder Applied Math

WHY THESE?

- Turns out that FC is unbiased, and therefore worse than OLS (has high variance)
- On the other hand, PC is biased and empirics demonstrate low variance
- Combination should give better statistical properties

Compressed regression

With this Q, $\widehat{\beta}_C$ "works" in practice:

- Computational savings: $O\left(\frac{qnp}{s} + qp^2\right)$
- Approximately the same estimation risk as OLS

This is good, but we had a realization:

If ridge regression is better than OLS, why not "point" the approximation at ridge?

Compressed ridge regression

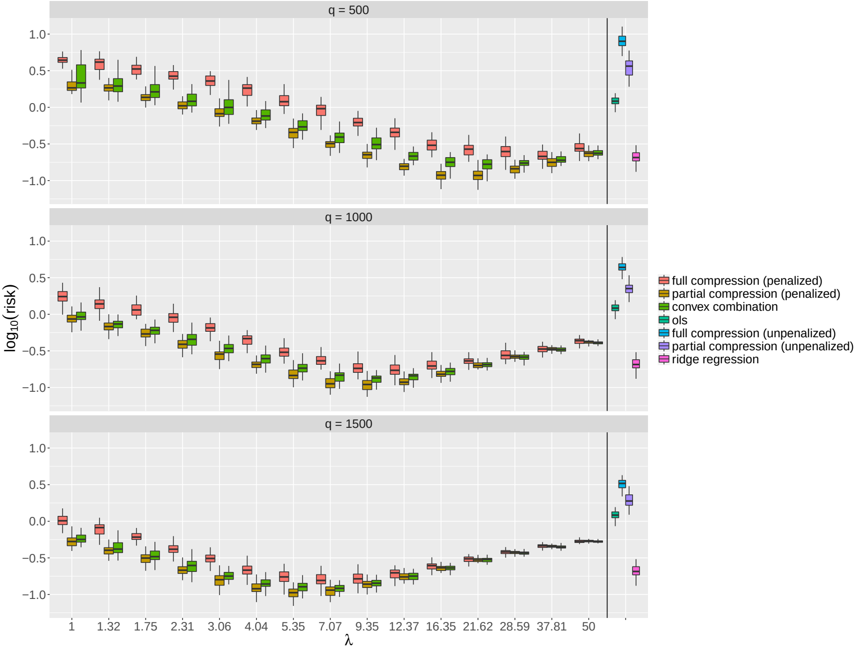
This means introducing a tuning parameter λ and defining:

$$\widehat{\beta}_{PC}(\lambda) = (\mathbf{X}^\top Q^\top Q\mathbf{X} + \lambda I)^{-1}\mathbf{X}^\top Y$$
$$\widehat{\beta}_{FC}(\lambda) = (\mathbf{X}^\top Q^\top Q\mathbf{X} + \lambda I)^{-1}\mathbf{X}^\top Q^\top QY$$

(Everything else about the procedure is the same)

This has the same computational complexity, but has much lower risk

Let’s look at an (a)typical result…


q = 500
q = 1000
q = 1500
1.0
0.5
0.0
−0.5
−1.0
log$_{10}$(risk)
1
1.32
1.75
2.31
3.06
4.04
5.35
7.07
9.35
12.37
16.35
21.62
28.59
37.81
50
λ
full compression (penalized)
partial compression (penalized)
convex combination
ols
full compression (unpenalized)
partial compression (unpenalized)
ridge regression

CONCLUSIONS

COMPRESSED REGRESSION

- In this case, approximation actually improves existing approaches
- We have ways of choosing the tuning parameter using the data
- Much more elaborate simulations and theory (current work)

GENERAL PHILOSOPHY

- Approximation is not necessarily a bad thing
- Don't just minimize it, that only considers the bias
- Examine statistical properties to calibrate
- In some cases, we can get the best of both worlds:
  - easier computations
  - better statistical properties