Efficient Algorithms for Large-scale Generalized Eigenvector Computation and Canonical Correlation Analysis

Rong Ge RONGGE@CS.DUKE.EDU
Duke University

Chi Jin CHIJIN@BERKELEY.EDU
University of California Berkeley

Sham Kakade SHAM@CS.WASHINGTON.EDU
University of Washington

Praneeth Netrapalli PRANEETHN@GMAIL.COM
Aaron Sidford ASID@MICROSOFT.COM
Microsoft Research New England

Abstract

This paper considers the problem of canonical-correlation analysis, and more broadly, the generalized eigenvector problem for a pair of symmetric matrices. We consider the setting of finding top-k canonical/eigen subspace, and solve these problems through a general framework that simply requires black box access to an approximate linear system solver. Instantiating this framework with accelerated gradient descent we obtain a running time of $O\left(\frac{zk\sqrt{\kappa}}{\rho}\log(1/\epsilon)\log\left(k\kappa/\rho\right)\right)$ where z is the total number of nonzero entries, κ is the condition number and ρ is the relative eigenvalue gap of the appropriate matrices. Our algorithm is linear in the input size and the number of components k up to a $\log(k)$ factor, which is essential for handling large-scale matrices that appear in practice. To the best of our knowledge this is the first such algorithm with global linear convergence. We hope that our results prompt further research improving the practical running time for performing these important data analysis procedures on large-scale data sets.

1. Introduction

Canonical-correlation analysis (CCA) and the generalized eigenvector problem are fundamental problems in scientific computing, data analysis, and statistics (Barnett & Preisendorfer, 1987; Friman et al., 2001).

Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48. Copyright 2016 by the author(s).

These problems arise naturally in statistical settings. Let $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{n\times d}$ denote two large sets of data points, with empirical covariance matrices $\mathbf{S}_x = \frac{1}{n}\mathbf{X}^\top\mathbf{X}$, $\mathbf{S}_y = \frac{1}{n}\mathbf{Y}^\top\mathbf{Y}$, and $\mathbf{S}_{xy} = \frac{1}{n}\mathbf{X}^\top\mathbf{Y}$ and suppose we wish to find features $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ that best encapsulate the similarity or dissimilarity of the data sets. CCA is the problem of maximizing the empirical correlation

$$\max_{\mathbf{x}^\top\mathbf{S}_{xx}\mathbf{x}=1 \text{ and } \mathbf{y}^\top\mathbf{S}_{yy}\mathbf{y}=1} \mathbf{x}^\top\mathbf{S}_{xy}\mathbf{y} \tag{1}$$

and thereby extracts common features of the data sets. On the other hand the generalized eigenvalue problems

$$\max_{\mathbf{x}\neq 0} \frac{\mathbf{x}^\top\mathbf{S}_{xx}\mathbf{x}}{\mathbf{x}^\top\mathbf{S}_{yy}\mathbf{x}} \quad \text{and} \quad \max_{\mathbf{y}\neq 0} \frac{\mathbf{y}^\top\mathbf{S}_{yy}\mathbf{y}}{\mathbf{y}^\top\mathbf{S}_{xx}\mathbf{y}}$$

compute features that maximizes discrepancies between the data sets. Both these problems are easily extended to the k-feature case (See Section 3). Algorithms for solving them are commonly used to extract features to compare and contrast large data sets and are used commonly in regression (Kakade & Foster, 2007), clustering (Chaudhuri et al., 2009), classification (Karampatziakis & Mineiro, 2013), word embeddings (Dhillon et al., 2011) and more.

Despite the prevalence of these problems and the breadth of research on solving them in practice ((Barnett & Preisendorfer, 1987; Barnston & Ropelewski, 1992; Sherry & Henson, 2005; Karampatziakis & Mineiro, 2013) to name a few), there are relatively few results on obtaining provably efficient algorithms. Both problems can be reduced to performing principle component analysis (PCA), albeit on complicated matrices e.g $\mathbf{S}_{yy}^{-1/2}\mathbf{S}_{xy}^{\top}\mathbf{S}_{xx}^{-1}\mathbf{S}_{xy}\mathbf{S}_{yy}^{-1/2}$ for CCA and $\mathbf{S}_{yy}^{-1/2}\mathbf{S}_{xx}\mathbf{S}_{yy}^{-1/2}$ for generalized eigenvector. However applying PCA to these matrices traditionally involves the formation of $\mathbf{S}_{xx}^{-1/2}$ and $\mathbf{S}_{yy}^{-1/2}$ which is prohibitive for sufficiently large datasets if we only want to estimate top-k eigenspace.

A natural open question in this area is to what degree can the formation of $\mathbf{S}_{xx}^{-1/2}$ and $\mathbf{S}_{yy}^{-1/2}$ can be bypassed to obtain efficient scalable algorithms in the case where the number of features k is much smaller than the dimensions of the problem n and d. Can we develop simple iterative practical methods that solve this problem in close to linear time when k is small and the condition number and eigenvalue gaps are bounded? While there has been recent work on solving these problems using iterative methods (Avron et al., 2014; Paul, 2015; Lu & Foster, 2014; Ma et al., 2015) we are unaware of previous provable global convergence results and more strongly, linearly convergent scalable algorithms.

The central goal of this paper is to answer this question in the affirmative. We present simple globally linearly convergent iterative methods that solve these problems. The running time of these problems scale well as the number of features and conditioning of the problem stay fixed and the size of the datasets grow. Moreover, we implement the method and perform experiments demonstrating that the techniques may be effective for large scale problems.

Specializing our results to the single feature case we show how to solve the problems all in time $O(\frac{z\sqrt{\kappa}}{\rho}\log\frac{1}{\rho}\log\frac{1}{\epsilon})$, where κ is the maximum of condition numbers of $\mathbf{S}_{xx}$ and $\mathbf{S}_{yy}$ and ρ is the eigengap of appropriate matrices and mentioned above, and z is the number of nonzero entries in $\mathbf{X}$ and $\mathbf{Y}$. To the best of our knowledge this is the first such globally linear convergent algorithm for solving these problems.

We achieve our results through a general and versatile framework that allows us to utilize fast linear system solvers in various regimes. We hope that by initiating this theoretical and practical analysis of CCA and the generalized eigenvector problem we can promote further research on the problem and ultimately advance the state-of-the-art for efficient data analysis.

1.1. Our Approach

To solve the problems motivated in the previous section we first directly reduce CCA to a generalized eigenvector problem (See Section 5). Consequently, for the majority of the paper we focus on the following:

Definition 1 (Top-k Generalized Eigenvector[1])**.** *Given symmetric matrices* $\mathbf{A}, \mathbf{B}$ *where* $\mathbf{B}$ *is positive definite compute* $\mathbf{w}_1, \cdots, \mathbf{w}_k$ *defined for all* $i \in [k]$ *by*

$$\mathbf{w}_i \in \underset{\mathbf{w}}{\operatorname{argmax}} \left|\mathbf{w}^\top \mathbf{A}\mathbf{w}\right| \ \textit{s.t.} \quad \begin{array}{c} \mathbf{w}^\top \mathbf{B}\mathbf{w} = 1 \textit{ and} \\ \mathbf{w}^\top \mathbf{B}\mathbf{w}_j = 0 \ \forall\, j \in [i-1]. \end{array}$$

The generalized eigenvector is equivalent to the problem of computing the PCA of $\mathbf{A}$ in the $\mathbf{B}$ norm. Consequently, it is the same as computing the top k eigenvectors of largest absolute value of the symmetric matrix $\mathbf{M} = \mathbf{B}^{-1/2}\mathbf{A}\mathbf{B}^{-1/2}$ and then multiplying by $\mathbf{B}^{-1/2}$.

Unfortunately, as we have discussed, explicitly computing $\mathbf{B}^{-1/2}$ is prohibitively expensive when n is large and therefore we wish to avoid forming $\mathbf{M}$ explicitly. One natural approach is to develop an iterative methods to approximately apply $\mathbf{B}^{-1/2}$ to a vector and then use that method as a subroutine to perform the power method on $\mathbf{M}$. Even if we could perform the error analysis to make this work, such an approach would likely require at least a suboptimal $\Omega(\log^2(1/\epsilon))$ iterations to achieve error ϵ.

To bypass these difficulties, we take a closer look at the power method. For some initial vector $\mathbf{x}$, let $\mathbf{y} = \mathbf{B}^{-1/2}\mathbf{M}^i\mathbf{x}$ be the result of i iterations of power method on $\mathbf{M}$ followed by multiplying $\mathbf{B}^{-1/2}$. Clearly $\mathbf{y} = (\mathbf{B}^{-1}\mathbf{A})^i\mathbf{B}^{-1/2}\mathbf{x}$. Furthermore, since we typically initialize the power method by a random vector and since $\mathbf{B}$ is positive definite, if we instead we computed $\mathbf{y} = (\mathbf{B}^{-1}\mathbf{A})^i\mathbf{x}$ for random $\mathbf{x}$ we would likely converge at the same rate as the power method at the cost of just a slightly worse initialization quality.

Consequently, we can compute our desired eigenvectors by simply alternating between applying $\mathbf{A}$ and $\mathbf{B}^{-1}$ to a random initial vector. Unfortunately, computing $\mathbf{B}^{-1}$ exactly is again outside our computational budget. At best we should only attempt to apply $\mathbf{B}^{-1}$ approximately by linear system solvers.

One of our main technical contributions is to argue about the effect of inexact solvers in this method. Whereas solving every linear system to target accuracy ϵ would again require $O(\log(1/\epsilon))$ time per linear system, which leads to a sub-optimal $O(\log^2(1/\epsilon))$ overall running time, i.e. sublinear convergence, we instead show how to warm start the linear system solvers and obtain a faster rate. We exploit the fact that as we perform many iterations of power methods, points at time t converge to eigenvectors and therefore we can initialize our linear system solver at time t carefully using our points at time $t-1$. Ultimately we show that we only need to make fixed multiplicative progress in solving

[1]We use the term *generalized eigenvector* to refer to a nonzero vector $\mathbf{v}$ such that $\mathbf{A}\mathbf{v} = \lambda\mathbf{B}\mathbf{v}$ for symmetric $\mathbf{A}$ and $\mathbf{B}$, not the general notion of eigenvectors for asymmetric matrices.

the linear system in every iteration of the power method, thus the runtime for solving each linear system is independent of ϵ.

Putting these pieces together with careful error analysis yields our main result. Our algorithm only requires the ability to apply $\mathbf{A}$ to a vector and an approximate linear system solver for $\mathbf{B}$, which in turn can be obtained by just applying $\mathbf{B}$ to vectors. Consequently, our framework is versatile, scalable, and easily adaptable to take advantage of faster linear system solvers.

1.2. Previous Work

While there has been limited previous work on provably solving CCA and generalized eigenvectors, we note that there is an impressive body of literature on performing PCA(Rokhlin et al., 2009; Halko et al., 2011; Musco & Musco, 2015; Garber & Hazan, 2015; Jin et al., 2015) and solving positive semidefinite linear systems(Hestenes & Stiefel, 1952; Nesterov, 1983; Spielman & Teng, 2004). Our analysis in this paper draws on this work extensively and our results should be viewed as the principled application of them to the generalized eigenvector problem.

There has been much recent interest in designing scalable algorithms for CCA(Ma et al., 2015; Wang et al., 2015; Wang & Livescu, 2015; Michaeli et al., 2015). To our knowledge, there are no provable guarantees for approximate methods for this problem. Heuristic-based approachs (Witten et al., 2009; Lu & Foster, 2014) compute efficiently, but only give suboptimal result due to coarse approximation. The work in (Ma et al., 2015) provides one natural iterative procedure, where the per iterate computational complexity is low. This work only provides local convergence guarantees and does not provide guarantees of global convergence.

Also of note is that many recent algorithms (Ma et al., 2015; Wang et al., 2015) have mini-batch variations, but there's no guarantees for mini-batch style algorithm for CCA yet. Our algorithm can also be easily extends to a mini-batch version. While we do not explicitly analyze this variation, and we believe our analysis and techniques are helpful for extensions to this setting. We also view this as an important direction for future work.

We hope that by establishing the generalized eigenvector problem and providing provable guarantees under moderate regularity assumptions that our results may be further improved and ultimately this may advance the state-of-the-art in practical algorithms for performing data analysis.

1.3. Our Results

Our main result in this paper is a linearly convergent algorithm for computing the top generalized eigenvectors (see Definition 1). In order to be able to state our results we introduce some notation. Let $\lambda_1, \cdots, \lambda_d$ be the eigenvalues of $\mathbf{B}^{-1}\mathbf{A}$ (their existence is guaranteed by Lemma 9 in the appendix). The eigengap $\rho \overset{\text{def}}{=} 1 - \frac{|\lambda_{k+1}|}{|\lambda_k|}$ and $\gamma \overset{\text{def}}{=} \frac{|\lambda_1|}{|\lambda_k|}$. Let z denote the number of nonzero entries in $\mathbf{A}$ and $\mathbf{B}$.

Theorem 2 (Informal version of Theorem 6)**.** *Given two matrices $\mathbf{A}$ and $\mathbf{B} \in \mathbb{R}^{d\times d}$, there is an algorithm that computes the top-k generalized eigenvectors up to an error ϵ in time $\widetilde{O}(\frac{zk\sqrt{\kappa(\mathbf{B})}}{\rho}\log\frac{1}{\epsilon})$, where $\kappa(\mathbf{B})$ is the condition number of $\mathbf{B}$ and $\widetilde{O}(\cdot)$ hides logarithmic terms in d, γ, $\kappa(\mathbf{B})$ and ρ, and nothing else.*

Here is a comparison of our result with previous work.

Table 1. Runtime Comparison - Generalized Eigenvectors

GENELINK(THIS PAPER)	$\widetilde{O}(\frac{d^2k\sqrt{\kappa(\mathbf{B})}}{\rho}\log\frac{1}{\epsilon})$
FAST MATRIX INVERSION	$O(d^{2.373\dots})$

Turning to the problem of CCA, cf. (1), the relevant parameters are $\kappa \overset{\text{def}}{=} \max(\kappa(\mathbf{S}_{xx}), \kappa(\mathbf{S}_{yy}))$ i.e., the maximum of the condition numbers of $\mathbf{S}_{xx}$ and $\mathbf{S}_{yy}$, $\gamma \overset{\text{def}}{=} \frac{|\lambda_1|}{|\lambda_k|}$ where $\lambda_1, \cdots, \lambda_k$ are the eigenvalues of $\mathbf{S}_{yy}^{-1}\mathbf{S}_{yx}\mathbf{S}_{xx}^{-1}\mathbf{S}_{xy}$ in decreasing absolute value. Let z denote the number of nonzeros in $\mathbf{X}$ and $\mathbf{Y}$. Our main results are a reduction from CCA to the generalized eigenvector problem.

Theorem 3 (Informal version of Theorem 7)**.** *Given data matrices $\mathbf{X} \in \mathbb{R}^{d_1\times n}$ and $\mathbf{Y} \in \mathbb{R}^{d_2\times n}$, there is an algorithm that performs top-k CCA up to error ϵ in time $\widetilde{O}(\frac{zk\sqrt{\kappa}}{\rho}\log\frac{1}{\epsilon})$, where $d = d_1 + d_2$ and $\widetilde{O}(\cdot)$ hides logarithmic terms in d, γ, κ and ρ, and nothing else.*

Table 2 compares our result with existing results. [2]

Table 2. Runtime Comparison - CCA

THIS PAPER	$\widetilde{O}(\frac{ndk\sqrt{\kappa}}{\rho}\log\frac{1}{\epsilon})$
S-APPGRAD (MA ET AL., 2015)	$\widetilde{O}(\frac{ndk\kappa}{\rho^2}\log\frac{1}{\epsilon})$
FAST MATRIX INVERSION	$O(nd^{1.373\dots})$

We should note that the actual bounds we obtain are somewhat stronger than the above informal bounds. Some of the terms in logarithm also appear only as additive terms. Finally we also give natural stochastic extensions of our algorithms where the cost of each iteration may be much smaller than the input size. The key idea behind our approach is to use an approximate linear system solver as a

[2](Ma et al., 2015) only shows local convergence for S-AppGrad. Starting within this radius of convergence requires us to already solve the problem to a high accuracy.

black box inside power method on an appropriate matrix. We show that this dependence on a linear system solver is in some sense essential. In Section A we show that the generalized eigenvector problem is strictly more general than the problem of solving positive semidefinite linear systems and consequently our dependence on the condition number of $\mathbf{B}$ is in some cases optimal.

Finally, we also run experiments to demonstrate the practical effectiveness of our algorithm on both small and large scale datasets.

1.4. Paper Overview

In Section 2, we present our notation. In Section 3, we formally define the problems we solve and their relevant parameters. In Section 4, we present our results for the generalized eigenvector problem. In Section 5, we present our results for the CCA problem. In Section A we argue that generalized eigenvector computation is as hard as linear system solving and that our dependence on $\kappa(\mathbf{B})$ is near optimal. In Section 6, we present experimental results of our algorithms on some real world data sets. Due to space limitations, proofs are deferred to the appendix.

2. Notation

We use bold capital letters $(\mathbf{A}, \mathbf{B}, \cdots)$ to denote matrices and bold lowercase letters $(\mathbf{u}, \mathbf{v}, \cdots)$ for vectors. For symmetric positive semidefinite (PSD) matrix $\mathbf{B}$, we let $\|\mathbf{u}\|_{\mathbf{B}} \stackrel{\text{def}}{=} \sqrt{\mathbf{u}^\top \mathbf{B} \mathbf{u}}$ denote the $\mathbf{B}$-norm of u and we let $\langle \mathbf{u}, \mathbf{v} \rangle_{\mathbf{B}} \stackrel{\text{def}}{=} \mathbf{u}^\top \mathbf{B} \mathbf{v}$ denotes the inner product of $\mathbf{u}$ and $\mathbf{v}$ in the $\mathbf{B}$-norm. We say that a matrix $\mathbf{W}$ is $\mathbf{B}$-orthonormal if $\mathbf{W}^\top \mathbf{B} \mathbf{W} = \mathbf{I}$. We let $\sigma_i(\mathbf{A})$ denotes the i^{th} largest singular value of $\mathbf{A}$, $\sigma_{\min}(\mathbf{A})$ and $\sigma_{\max}(\mathbf{A})$ denote the smallest and largest singular values of $\mathbf{A}$ respectively. Similarly we let $\lambda_i(\mathbf{A})$ refers to the i^{th} largest eigenvalue of $\mathbf{A}$ in magnitude. We let $\text{nnz}(\mathbf{A})$ denotes the number of nonzeros in $\mathbf{A}$. We also let $\kappa(\mathbf{B})$ denote the condition number of $\mathbf{B}$ (i.e., the ratio of the largest to smallest eigenvalue).

3. Problem Statement

In this section, we recall the generalized eigenvalue problem, define our error metric, and introduce all relevant parameters. Recall that the generalized eigenvalue problem is to find k vectors $\mathbf{w}_i$, $i \in [k]$ such that

$$\mathbf{w}_i \in \operatorname*{argmax}_{\mathbf{w}} \left|\mathbf{w}^\top \mathbf{A} \mathbf{w}\right| \quad \text{s.t.} \quad \begin{array}{c} \mathbf{w}^\top \mathbf{B} \mathbf{w} = 1 \text{ and} \\ \mathbf{w}^\top \mathbf{B} \mathbf{w}_j = 0 \, \forall \, j \in [i-1]. \end{array}$$

Using stationarity conditions, it can be shown that the vectors $\mathbf{w}_i$ are given by $\mathbf{w}_i = \mathbf{v}_i$, where v_i is an eigenvector of $\mathbf{B}^{-1}\mathbf{A}$ with eigenvalue λ_i such that $|\lambda_1| \geq \cdots \geq \lambda_n$. Our goal is to recover the top-k eigen space i.e., $\text{span}\{\mathbf{v}_1, \cdots, \mathbf{v}_k\}$. In order to quantify the error in estimating the eigenspace, we use largest principal angle, which is a standard notion of distance between subspaces (Golub & Van Loan, 2012).

Definition 4 (Largest principal angle)**.** *Let $\mathcal{W}$ and $\mathcal{V}$ be two k dimensional subspaces, $\mathbf{W}$ and $\mathbf{V}$ their $\mathbf{B}$-orthonormal basis respectively. The largest principal angle $\theta(\mathcal{W}, \mathcal{V})$ in the $\mathbf{B}$-norm is defined to be*

$$\theta(\mathcal{W}, \mathcal{V}) \stackrel{\text{def}}{=} \arccos\left(\sigma_{min}\left(\mathbf{V}^\top \mathbf{B} \mathbf{W}\right)\right),$$

Intuitively, the largest principal angle corresponds to the largest angle between any vector in the span of $\mathbf{W}$ and its projection onto the span of $\mathbf{V}$. In the special case where $k = 1$, the above definition reduces to our choice in the top-1 setting. Given two matrices $\mathbf{W}$ and $\mathbf{V}$, we use $\theta(\mathbf{W}, \mathbf{V})$ to denote the largest principle angle between the subspaces spanned by the columns of $\mathbf{W}$ and $\mathbf{V}$. We say that $\mathbf{W}$ achieves an error of ϵ if $\mathbf{W}^\top \mathbf{B} \mathbf{W} = \mathbf{I}$ and $\sin\theta(\mathbf{W}, \mathbf{V}) \leq \epsilon$, where $\mathbf{V}$ is the $d \times k$ matrix whose columns are $\mathbf{v}_1, \cdots, \mathbf{v}_k$. The relevant parameters for us are the eigengap, i.e. the relative difference between k^{th} and $(k+1)^{\text{th}}$ eigenvalues, $\rho \stackrel{\text{def}}{=} 1 - \frac{|\lambda_{k+1}|}{|\lambda_k|}$, and $\kappa(\mathbf{B})$, the condition number of $\mathbf{B}$.

4. Our Results

In this section, we provide our algorithms and results for solving the generalized eigenvector problem. We present our results for the special case of computing the top generalized eigenvector (Section 4.1) followed by the general case of computing the top-k generalized eigenvectors (Section 4.2). However, first we formally define a linear system solver as follows:

Linear system solver: In each of our main results (Theorems 5 and 6) we assume black box access to an approximate linear system solver. Given a PSD matrix $\mathbf{B}$, a vector $\mathbf{b}$, an initial estimate $\mathbf{u}_0$, and an error parameter δ, we require to decrease the error by a multiplicative δ, i.e. output $\mathbf{u}_1$ with $\|\mathbf{u}_1 - \mathbf{B}^{-1}\mathbf{b}\|_{\mathbf{B}}^2 \leq \delta \|\mathbf{u}_0 - \mathbf{B}^{-1}\mathbf{b}\|_{\mathbf{B}}^2$. We let $\mathcal{T}(\delta)$ denote the time needed for this operation. Since the error metric $\|\mathbf{u}_1 - \mathbf{B}^{-1}\mathbf{b}\|_{\mathbf{B}}^2$ is equivalent to function error on minimizing the convex quadratic $f(\mathbf{u}) \stackrel{\text{def}}{=} \frac{1}{2}\mathbf{u}^\top \mathbf{B} \mathbf{u} - \mathbf{u}^\top \mathbf{b}$ up to constant scaling, an approximate linear system solver is equivalent to an optimization algorithm for $f(\mathbf{u})$. We also specialize our results using Nesterov's accelerated gradient descent to state our bounds. Stating our results using linear system solver as a blackbox allows the user to choose an efficient solver depending on the structure of $\mathbf{B}$ and helps pass any improvements in linear system solvers on to the problem of generalized eigenvectors.

4.1. Top-1 Setting

Our algorithm for computing the top generalized eigenvector, GenELin is given in Algorithm 1.

Algorithm 1 **Gen**eralized **E**igenvector via **Lin**ear System Solver (GenELin)

```
Input: $T$, symmetric matrix $\mathbf{A}$, PSD matrix $\mathbf{B}$.
Output: top generalized eigenvector $\mathbf{w}$.
  $\tilde{\mathbf{w}}_0 \leftarrow$ sample uniformly from unit sphere in $\mathbb{R}^d$
  $\mathbf{w}_0 \leftarrow \tilde{\mathbf{w}}_0 / \|\tilde{\mathbf{w}}_0\|_{\mathbf{B}}$
  for $t = 0, \cdots, T-1$ do
    $\beta_t \leftarrow \mathbf{w}_t^\top \mathbf{A} \mathbf{w}_t / \mathbf{w}_t^\top \mathbf{B} \mathbf{w}_t$
    $\tilde{\mathbf{w}}_{t+1} \leftarrow \mathrm{argmin}_{\mathbf{w} \in \mathbb{R}^d} [\frac{1}{2}\mathbf{w}^\top \mathbf{B} \mathbf{w} - \mathbf{w}^\top \mathbf{A} \mathbf{w}_t]$
          {Use an optimization subroutine
           with initialization $\beta_t \mathbf{w}_t$ }
    $\mathbf{w}_{t+1} \leftarrow \tilde{\mathbf{w}}_{t+1} / \|\tilde{\mathbf{w}}_{t+1}\|_{\mathbf{B}}$
  end for
  Return $\mathbf{w}_T$.
```

The algorithm implements an approximate power method where each iteration consists of approximately multiplying a vector by $\mathbf{B}^{-1}\mathbf{A}$. In order to do this, GenELin solves a linear system in $\mathbf{B}$ and then scales the resulting vector to have unit $\mathbf{B}$-norm. Our main result states that given an oracle for solving the linear systems,[3] the number of iterations taken by Algorithm 1 to compute the top eigenvector up to an accuracy of ϵ is at most $\frac{4}{\rho} \log \frac{1}{\epsilon \cos \theta_0}$ where $\theta_0 \overset{\text{def}}{=} \theta(\mathbf{w}_0, \mathbf{v}_1)$.

Theorem 5. *Recall that the linear system solver takes time $\mathcal{T}(\delta)$ to reduce the error by a factor δ. Given matrices $\mathbf{A}$ and $\mathbf{B}$, GenELin (Algorithm 1) computes a vector $\mathbf{w}_T$ achieving an error of ϵ in $T = \frac{2}{\rho} \log \frac{1}{\epsilon \cos \theta_0}$ iterations, where $\theta_0 \overset{\text{def}}{=} \theta(\mathbf{w}_0, \mathbf{v}_1)$. The running time of the algorithm is at most*

$$O\left(\frac{1}{\rho}\left(\log \frac{1}{\cos \theta_0} \cdot \mathcal{T}(\frac{\rho^2 \cos^2 \theta_0}{16}) + \log \frac{1}{\epsilon} \cdot \mathcal{T}(\frac{\rho^2}{16})\right) + \frac{1}{\rho}(\mathrm{nnz}(\mathbf{A}) + \mathrm{nnz}(\mathbf{B}) + d) \log \frac{1}{\epsilon \cos \theta_0}\right).$$

Furthermore, if we use Nesterov's accelerated gradient descent (Algorithm 4) to solve the linear systems in Algorithm 1, the time can be bounded as

$$O\left(\frac{\mathrm{nnz}(\mathbf{B})\sqrt{\kappa(\mathbf{B})}}{\rho}\left(\log \frac{1}{\cos \theta_0} \log \frac{1}{\rho \cos \theta_0} + \log \frac{1}{\epsilon} \log \frac{1}{\rho}\right) + \frac{1}{\rho} \mathrm{nnz}(\mathbf{A}) \log \frac{1}{\epsilon \cos \theta_0}\right).$$

Remarks:

- Since GenELin chooses $\mathbf{w}_0$ randomly, Lemma 13 tells us that $\cos \theta_0 \geq \frac{\zeta}{\sqrt{d\kappa(\mathbf{B})}}$ with probability greater than $1 - \zeta$.
- Note that GenELin exploits the sparsity of input matrices since we only need to apply them as operators.
- Depending on computational restrictions, we can also use a subset of samples in each iteration of GenELin. In some large scale learning applications using mini-batches of data in each iteration helps make the method scalable while still maintaining the quality of performance.

4.2. Top-k Setting

In this section, we give an extension of our algorithm and result for computing the top-k generalized eigenvectors. Our algorithm, GenELinK is formally given as Algorithm 2.

Algorithm 2 **Gen**eralized **E**igenvectors via **Lin**ear System Solvers-**K** (GenELinK).

```
Input: $T$, $k$, symmetric matrix $\mathbf{A}$, PSD matrix $\mathbf{B}$.
  a subroutine $\mathrm{GS}_{\mathbf{B}}(\cdot)$ that performs Gram-Schmidt process, with inner product $\langle \cdot, \cdot \rangle_{\mathbf{B}}$.
Output: top $k$ eigen-space $\mathbf{W} \in \mathbb{R}^{d \times k}$.
  $\tilde{\mathbf{W}}_0 \leftarrow$ random $d \times k$ matrix with each entry i.i.d from $\mathcal{N}(0,1)$
  $\mathbf{W}_0 \leftarrow \mathrm{GS}_{\mathbf{B}}(\tilde{\mathbf{W}}_0)$.
  for $t = 0, \cdots, T-1$ do
    $\Gamma_t \leftarrow (\mathbf{W}_t^\top \mathbf{B} \mathbf{W}_t)^{-1} (\mathbf{W}_t^\top \mathbf{A} \mathbf{W}_t)$
    $\tilde{\mathbf{W}}_{t+1} \leftarrow \mathrm{argmin}_{\mathbf{W}} \mathrm{tr}(\frac{1}{2}\mathbf{W}^\top \mathbf{B} \mathbf{W} - \mathbf{W}^\top \mathbf{A} \mathbf{W}_t)$
          {Use an optimization subroutine
           with initialization $\mathbf{W}_t \Gamma_t$ }
    $\mathbf{W}_{t+1} \leftarrow \mathrm{GS}_{\mathbf{B}}(\tilde{\mathbf{W}}_{t+1})$
  end for
  Return $\mathbf{W}_T$.
```

GenELinK is a natural generalization of GenELin from the previous section. Given an initial set of vectors $\mathbf{W}_0$, the algorithm proceeds by doing approximate orthogonal iteration. Each iteration involves solving k independent linear systems[4] and orthonormalizing the iterates. The following theorem is the main result of our paper which gives runtime bounds for Algorithm 2. As before, we assume access to a blackbox linear system solver and also give a result instantiating the theorem with Nesterov's accelerated gradient descent algorithm.

Theorem 6. *Suppose the linear system solver takes time $\mathcal{T}(\delta)$ to reduce the error by a factor δ. Given input matrices*

[3]For example, we could use Nesterov's accelerated gradient descent, Algorithm 4

[4]Similarly, as before, we could use Nesterov's accelerated gradient descent, i.e. Algorithm 4.

$\mathbf{A}$ *and* $\mathbf{B}$*, GenELinK computes a* $d \times k$ *matrix* $\mathbf{W}_T$ *which is an estimate of the top generalized eigenvectors* $\mathbf{V}$ *with an error of* ϵ *i.e.,* $\mathbf{W}_T^\top \mathbf{B} \mathbf{W}_T = \mathbf{I}$ *and* $\sin\theta_T \leq \epsilon$*, where* $\theta_T \overset{\text{def}}{=} \theta(\mathbf{W}_T, \mathbf{V})$ *in* $T = \frac{2}{\rho}\log\frac{1}{\epsilon\cos\theta_0}$ *iterations where* $\theta_0 = \theta(\mathbf{W}_0, \mathbf{V})$*. The run time of this algorithm is at most*

$$O\left(\frac{1}{\rho}\left(\log\frac{1}{\cos\theta_0}\cdot\mathcal{T}(\frac{\rho^2\cos^4\theta_0}{64k\gamma^2}) + \mathcal{T}(\frac{\rho^2}{64k\gamma^2})\log\frac{1}{\epsilon}\right)\right.$$
$$\left.+\frac{1}{\rho}\left(\text{nnz}(\mathbf{A})k + \text{nnz}(\mathbf{B})k + dk^2\right)\log\frac{1}{\epsilon\cos\theta_0}\right),$$

where $\gamma \overset{\text{def}}{=} \frac{|\lambda_1|}{|\lambda_k|}$*,* $|\lambda_1| \geq \cdots \geq |\lambda_k|$ *being the top-k eigenvalues of* $\mathbf{B}^{-1}\mathbf{A}$*. Furthermore, if we use Nesterov's accelerated gradient descent (Algorithm 4) to solve the linear systems in Algorithm 2, the time above can be bounded as*

$$O\left(\frac{\text{nnz}(\mathbf{B})k\sqrt{\kappa(\mathbf{B})}}{\rho}\left(\log\frac{1}{\cos\theta_0}\log\frac{k\gamma}{\rho\cos\theta_0}\right.\right.$$
$$\left.\left.+\log\frac{1}{\epsilon}\log\frac{k\gamma}{\rho}\right) + \frac{\left(\text{nnz}(\mathbf{A})k + dk^2\right)}{\rho}\log\frac{1}{\epsilon\cos\theta_0}\right).$$

Remarks:

- Lemma 13 again tells us that since $\mathbf{W}_0$ is chosen to be normalized after choosing uniformly at random from the unit sphere, $\cos\theta_0 \geq \frac{\zeta}{\sqrt{dk\kappa(\mathbf{B})}}$ with probability greater than $1-\zeta$.
- This result recovers Theorem 5 as a special case, since when $k=1$, we also have $\gamma = \frac{|\lambda_1|}{|\lambda_1|} = 1$.

5. Application to CCA

We now outline how the CCA problem can be reduced to computing generalized eigenvectors. The CCA problem is as follows. Given two sets of data points $\mathbf{X} \in \mathbb{R}^{n\times d_1}$ and $\mathbf{Y} \in \mathbb{R}^{n\times d_2}$, let $\mathbf{S}_x \overset{\text{def}}{=} \mathbf{X}^\top\mathbf{X}/n$, $\mathbf{S}_y \overset{\text{def}}{=} \mathbf{Y}^\top\mathbf{Y}/n$, and $\mathbf{S}_{xy} \overset{\text{def}}{=} \mathbf{X}^\top\mathbf{Y}/n$. We wish to find vectors $\phi_1, \cdots, \phi_k$ and $\psi_1, \cdots, \psi_k$ which are defined recursively as

$$(\phi_i, \psi_i) \in \underset{\phi,\psi}{\text{argmax}}\, \phi^\top \mathbf{S}_{xy}\psi$$
$$\text{s.t.}\quad \begin{array}{l}\|\phi\|_{\mathbf{S}_x} = 1 \text{ and } \phi^\top\mathbf{S}_x\phi_j = 0\ \forall\, j \leq i-1 \\ \|\psi\|_{\mathbf{S}_y} = 1 \text{ and } \psi^\top\mathbf{S}_y\psi_j = 0\ \forall\, j \leq i-1.\end{array}$$

where the values of $\phi_i^\top\mathbf{S}_{xy}\psi_i$ are called canonical correlations between $\mathbf{X}$ and $\mathbf{Y}$.

For reduction, we know any stationary point of this optimization problem satisfies $\mathbf{S}_{xy}\psi_i = \lambda_i\mathbf{S}_x\phi_i$, and $\mathbf{S}_{yx}\phi_i = \mu_i\mathbf{S}_y\psi_i$, where λ_i and μ_i are two constants. Combined with the constraints, we also see that $\lambda_i = \mu_i$. This can be written in matrix form as $\begin{pmatrix}0 & \mathbf{S}_{xy}\\ \mathbf{S}_{yx} & 0\end{pmatrix}\begin{pmatrix}\phi_i\\ \psi_i\end{pmatrix} = \lambda_i\begin{pmatrix}\mathbf{S}_x & 0\\ 0 & \mathbf{S}_y\end{pmatrix}\begin{pmatrix}\phi_i\\ \psi_i\end{pmatrix}$. Suppose the generalized eigenvalues of the above matrices are $-\lambda_1 < -\lambda_2 < \cdots < \lambda_2 < \lambda_1$. The top $2k$-dimensional eigen-space of this generalized eigenvalue problem corresponds to the linear subspace spanned by the eigenvectors of λ_i and $-\lambda_i$, which are $\begin{pmatrix}\phi_i\\ \psi_i\end{pmatrix}$, $\begin{pmatrix}-\phi_i\\ \psi_i\end{pmatrix}$ $\forall\ i \in [k]$. Once we solve the top-$2k$ generalized eigenvector problem for the matrices $\begin{pmatrix}0 & \mathbf{S}_{xy}\\ \mathbf{S}_{yx} & 0\end{pmatrix}$ and $\begin{pmatrix}\mathbf{S}_{xx} & 0\\ 0 & \mathbf{S}_{yy}\end{pmatrix}$, we can pick any orthonormal basis that spans the output subspace and choose a random k-dimensional projection of those vectors. The formal algorithm is given in Algorithm 3. Combining this with our results for computing generalized eigenvectors, we obtain the following result.

Algorithm 3 CCA via **Lin**ear System Solvers (CCALin)
```
Input: $T$, $k$, data matrix $\mathbf{X} \in \mathbb{R}^{n\times d_1}$, $\mathbf{Y} \in \mathbb{R}^{n\times d_2}$
Output: top $k$ canonical subspace $\mathbf{W}_x \in \mathbb{R}^{d_1\times k}$, $\mathbf{W}_y \in \mathbb{R}^{d_2\times k}$.
  $\mathbf{S}_{xx} \leftarrow \mathbf{X}^\top\mathbf{X}/n,\ \mathbf{S}_{yy} \leftarrow \mathbf{Y}^\top\mathbf{Y}/n,\ \mathbf{S}_{xy} \leftarrow \mathbf{X}^\top\mathbf{Y}/n$.
  $\mathbf{A} \leftarrow \begin{pmatrix}0 & \mathbf{S}_{xy}\\ \mathbf{S}_{xy}^\top & 0\end{pmatrix}, \mathbf{B} \leftarrow \begin{pmatrix}\mathbf{S}_{xx} & 0\\ 0 & \mathbf{S}_{yy}\end{pmatrix}$
  $\begin{pmatrix}\bar{\mathbf{W}}_x \in \mathbb{R}^{d_1\times 2k}\\ \bar{\mathbf{W}}_y \in \mathbb{R}^{d_2\times 2k}\end{pmatrix} \leftarrow$ GenELinK$(\mathbf{A}, \mathbf{B})$.
  $\mathbf{U} \leftarrow 2k \times k$ random Gaussian matrix
  $\tilde{\mathbf{W}}_x \leftarrow \bar{\mathbf{W}}_x\mathbf{U}$.
  $\tilde{\mathbf{W}}_y \leftarrow \bar{\mathbf{W}}_y\mathbf{U}$.
  $\mathbf{W}_x = \text{GS}_{\mathbf{S}_{xx}}(\tilde{\mathbf{W}}_x), \mathbf{W}_x = \text{GS}_{\mathbf{S}_{yy}}(\tilde{\mathbf{W}}_y)$
  Return $\mathbf{W}_x, \mathbf{W}_y$.
```

Theorem 7. *Suppose the linear system solver takes time* $\mathcal{T}(\delta)$ *to reduce the error by a factor* δ*. Given inputs* $\mathbf{X}$ *and* $\mathbf{Y}$*, with probability greater than* $1-\zeta$*, then there is some universal constant c, so that Algorithm 3 outputs* $\mathbf{W}_x$ *and* $\mathbf{W}_y$ *such that* $\sin\theta(\text{span}(\phi_i; i\in[k]), \mathbf{W}_x) \leq \epsilon$*, and* $\sin\theta(\text{span}(\psi_i; i\in[k]), \mathbf{W}_y) \leq \epsilon$*, in time*

$$O\left(\frac{1}{\rho}\left(\log\frac{d\kappa}{\zeta}\cdot\mathcal{T}(\frac{c\zeta^6\rho^2}{d^2k^5\kappa^2\gamma^2}) + \mathcal{T}(\frac{c\zeta^2\rho^2}{k^3\gamma^2})\log\frac{1}{\epsilon}\right)\right.$$
$$\left.+\frac{1}{\rho}\left(\text{nnz}(\mathbf{X},\mathbf{Y})k + dk^2\right)\log\frac{d\kappa}{\zeta\epsilon}\right),$$

where $\text{nnz}(\mathbf{X},\mathbf{Y}) \overset{\text{def}}{=} \text{nnz}(\mathbf{X}) + \text{nnz}(\mathbf{Y})$ *and* $\kappa \overset{\text{def}}{=} \max(\kappa(\mathbf{S}_{xx}), \kappa(\mathbf{S}_{yy}))$ *and* $\gamma \overset{\text{def}}{=} \frac{\lambda_1}{\lambda_k}$*. If we use Nesterov's accelerated gradient descent (Algorithm 4) to solve the lin-*

ear systems in GenELink, then the total runtime is

$$O\left(\frac{\mathrm{nnz}\left(\mathbf{X},\mathbf{Y}\right)k\sqrt{\kappa}}{\rho}\left(\log\frac{d\kappa}{\zeta}\log\frac{d\kappa\gamma}{\zeta\rho}+\log\frac{1}{\epsilon}\log\frac{k\gamma}{\rho}\right)\right.$$
$$\left.+\frac{2dk^2}{\rho}\log\frac{d\kappa}{\zeta\epsilon}\right),$$

Remarks:

- Note that we depend on the maximum of the condition numbers of $\mathbf{S}_{xx}$ and $\mathbf{S}_{yy}$ since the linear systems that arise in GenELinK decompose into two separate linear systems, one in $\mathbf{S}_{xx}$ and the other in $\mathbf{S}_{yy}$.
- We can also exploit sparsity in the data matrices $\mathbf{X}$ and $\mathbf{Y}$ since we only need to apply $\mathbf{S}_{xx}, \mathbf{S}_{xy}$ or $\mathbf{S}_{yy}$ only as operators, which can be done by applying $\mathbf{X}$ and $\mathbf{Y}$ in appropriate order.

6. Simulations

In this section, we present our experiment results performing CCA on three benchmark datasets which are summarized in Table 3. We wish to demonstrate two things via these simulations: 1) the behavior of CCALin verifies our theoretical result on relatively small-scale dataset, and 2) scalability of CCALin comparing it with other existing algorithms on a large-scale dataset.

Table 3. Summary of Datasets

DATASET	d_1	d_2	n	SPARSITY [5]
MNIST	392	392	6×10^4	0.19
PENN TREE BANK	10^4	10^4	5×10^5	1×10^{-4}
URL REPUTATION	10^5	10^5	1×10^6	5.8×10^{-5}

Let us now specify the error metrics we use in our experiments. The first ones are the principal angles between the estimated subspaces and the true ones. Let $\mathbf{W}_x$ and $\mathbf{W}_y$ be the estimated subspaces and $\mathbf{V}_x$, $\mathbf{V}_y$ be the true canonical subspaces. We will use principle angles $\theta_x = \theta(\mathbf{W}_x, \mathbf{V}_x)$ under $\mathbf{S}_{xx}$-norm, $\theta_y = \theta(\mathbf{W}_x, \mathbf{V}_x)$ under $\mathbf{S}_{yy}$-norm and $\theta_{\mathbf{B}} = \theta\left(\begin{pmatrix}\mathbf{V}_x & 0\\ 0 & \mathbf{V}_y\end{pmatrix}, \begin{pmatrix}\bar{\mathbf{W}}_x\\ \bar{\mathbf{W}}_y\end{pmatrix}\right)$ [6], under the $\begin{pmatrix}\mathbf{S}_{xx} & 0\\ 0 & \mathbf{S}_{yy}\end{pmatrix}$ norm. Unfortunately, we cannot compute these error metrics for large-scale datasets since they require knowledge of the true canonical components. Instead we will use Total Correlations Captured (TCC), which is another metric widely used by practitioners, defined to be the sum of canonical correlation between

[5]Sparsity is given by $(\mathrm{nnz}\left(\mathbf{X}\right)+\mathrm{nnz}\left(\mathbf{Y}\right))/(nd_1+nd_2)$.
[6]See Algorithm 3 for definition of $\bar{\mathbf{W}}_x, \bar{\mathbf{W}}_y$

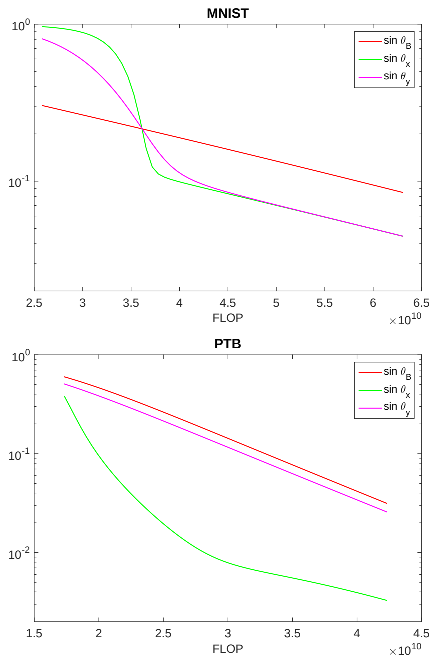

Figure 1. Linear convergence of principle angles

two matrices. Also, Proportion of Correlations Captured is given as

$$\mathrm{PCC} = \mathrm{TCC}(\mathbf{X}\mathbf{W}_x, \mathbf{Y}\mathbf{W}_y)/\mathrm{TCC}(\mathbf{X}\mathbf{V}_x, \mathbf{Y}\mathbf{V}_y)$$

For a fair comparison with other algorithms (which usually call highly optimized matrix inversion subroutines), we use number of FLOPs instead of wall clock time to measure the performance.

6.1. Small-scale Datasets

MNIST dataset(LeCun et al., 1998) consists of 60,000 handwritten digits from 0 to 9. Each digit is a image represented by 392×392 real values in [0,1]. Here CCA is performed between left half images and right half images. The data matrix is dense but the dimension is fairly small.

Penn Tree Bank (PTB) dataset comes from full Wall Street Journal Part of Penn Tree Bank which consists of 1.17 million tokens and a vocabulary size of 43k(Marcus et al., 1993), which has already been used to successfully learn the word embedding by CCA(Dhillon et al., 2011). Here, the task is to learn correlated components between two consecutive words. We only use the top 10,000 most frequent words. Each row of data matrix $\mathbf{X}$ is an indicator vector and hence it is very sparse and $\mathbf{X}^\top\mathbf{X}$ is diagonal.

Since the input matrices are very ill conditioned, we add some regularization and replace $\mathbf{S}_{xx}$ by $\mathbf{S}_{xx} + \lambda\mathbf{I}$ (and similarly with $\mathbf{S}_{yy}$). In CCALin, we run GenELinK with $k = 10$ and accelerated gradient descent (Algorithm 4 in

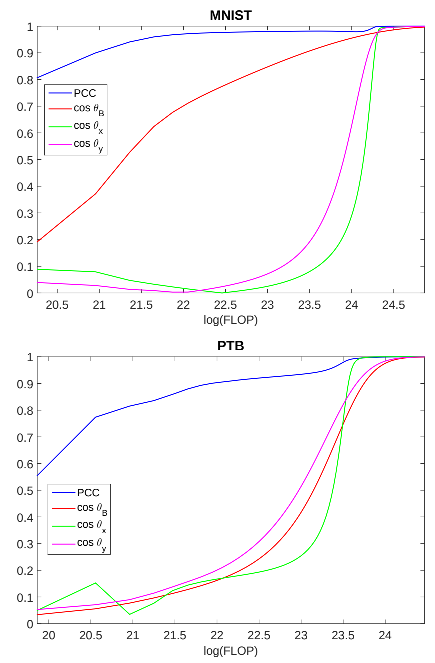

Figure 2. Global convergence of PCC and principle angles

the supplementary material) to solve the linear systems. The results are presented in Figure 2 and Figure 1.

Figure 2 shows a typical run of CCALin from random initialization on both MNIST and PTB dataset. We see although θ_x, θ_y may be even 90 degree at some point respectively, $\theta_{\mathbf{B}}$ is always monotonically decreasing (as $\cos\theta_{\mathbf{B}}$ monotonically increasing) as predicted by our theory. In the end, as $\theta_{\mathbf{B}}$ goes to zero, it will push both θ_x and θ_y go to zero, and PCC go to 1. This demonstrates that our algorithm indeed converges to the true canonical space.

Furthermore, by a more detailed examination of experimental data in Figure 2, we observe in Figure 1 that $\sin\theta_{\mathbf{B}}$ is indeed linearly convergent as we predicted in the theory. In the meantime, $\sin\theta_x$ and $\sin\theta_y$ may initially converge a bit slower than $\sin\theta_{\mathbf{B}}$, but in the end they will be upper bounded by $\sin\theta_{\mathbf{B}}$ times a constant factor, thus will eventually converge at a linear rate at least as fast as $\sin\theta_{\mathbf{B}}$.

6.2. Large-scale Dataset

URL Reputation dataset contains 2.4 million URLs and 3.2 million features including both host-based features and lexical based features. Each feature is either real valued or binary. For experiments in this section, we follow the setting of (Ma et al., 2015). We use the first 2 million samples, and run CCA between a subset of host based features and a subset of lexical based features to extract the top 20 components. Although the data matrix $\mathbf{X}$ is relatively sparse, unlike PTB, it has strong correlations among different coordinates, which makes $\mathbf{X}^\top\mathbf{X}$ much denser ($\mathrm{nnz}\left(\mathbf{X}^\top\mathbf{X}\right)/d_1^2 \approx 10^{-3}$).

Classical algorithms are impractical for this dataset on a typical computer, either running out of memory or requiring prohibitive amount of time. Since we cannot estimate the principal angles, we will evaluate TCC performance of CCALin.

We compare our algorithm to S-AppGrad (Ma et al., 2015) which is an iterative algorithm and PCA-CCA (Ma et al., 2015), NW-CCA (Witten et al., 2009) and DW-CCA (Lu & Foster, 2014) which are one-shot estimation procedures.

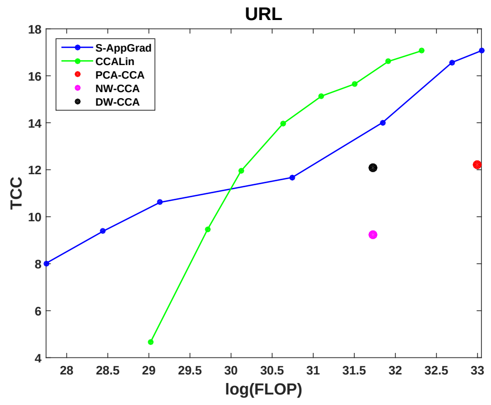

Figure 3. Comparison with existing algorithms

In CCALin, we employ GenELinK using stochastic accelerated gradient descent for solving linear systems using minibatches in each of the gradient steps and also leverage sparsity of the data to deal with the large data size. The result is shown in Figure 3. It is clear from the plot that our algorithm takes fewer computations than the other algorithms to achieve the same accuracy.

7. Conclusion

In summary, we have provided the first provable globally linearly convergent algorithms for solving canonical correlation analysis and the generalized eigenvector problems. We have shown that for recovering the top k components our algorithms are much faster than traditional methods based on fast matrix multiplication and singular value decomposition when $k \ll n$ and the condition numbers and eigenvalue gaps of the matrices involved are moderate. Moreover, we have provided empirical evidence that our algorithms may be useful in practice. We hope these results serve as the basis for further improvements in performing large scale data analysis both in theory and in practice.

References

Avron, Haim, Boutsidis, Christos, Toledo, Sivan, and Zouzias, Anastasios. Efficient dimensionality reduction for canonical correlation analysis. *SIAM Journal on Scientific Computing*, 36(5):S111–S131, 2014.

Barnett, TP and Preisendorfer, R. Origins and levels of monthly and seasonal forecast skill for united states surface air temperatures determined by canonical correlation analysis. *Monthly Weather Review*, 115(9):1825–1850, 1987.

Barnston, Anthony G and Ropelewski, Chester F. Prediction of enso episodes using canonical correlation analysis. *Journal of climate*, 5(11):1316–1345, 1992.

Chaudhuri, Kamalika, Kakade, Sham M, Livescu, Karen, and Sridharan, Karthik. Multi-view clustering via canonical correlation analysis. In *Proceedings of the 26th annual international conference on machine learning*, pp. 129–136. ACM, 2009.

Dhillon, Paramveer, Foster, Dean P, and Ungar, Lyle H. Multi-view learning of word embeddings via cca. In *Advances in Neural Information Processing Systems*, pp. 199–207, 2011.

Friman, Ola, Cedefamn, Jonny, Lundberg, Peter, Borga, Magnus, and Knutsson, Hans. Detection of neural activity in functional mri using canonical correlation analysis. *Magnetic Resonance in Medicine*, 45(2):323–330, 2001.

Garber, Dan and Hazan, Elad. Fast and simple pca via convex optimization. *arXiv preprint arXiv:1509.05647*, 2015.

Golub, Gene H and Van Loan, Charles F. *Matrix computations*, volume 3. JHU Press, 2012.

Halko, Nathan, Martinsson, Per-Gunnar, and Tropp, Joel A. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

Hestenes, Magnus Rudolph and Stiefel, Eduard. *Methods of conjugate gradients for solving linear systems*, volume 49. NBS, 1952.

Jin, Chi, Kakade, Sham M., Musco, Cameron, Netrapalli, Praneeth, and Sidford, Aaron. Robust shift-and-invert preconditioning: Faster and more sample efficient algorithms for eigenvector computation. *CoRR*, abs/1510.08896, 2015.

Kakade, Sham M and Foster, Dean P. Multi-view regression via canonical correlation analysis. In *Learning theory*, pp. 82–96. Springer, 2007.

Karampatziakis, Nikos and Mineiro, Paul. Discriminative features via generalized eigenvectors. *arXiv preprint arXiv:1310.1934*, 2013.

LeCun, Yann, Cortes, Corinna, and Burges, Christopher JC. The mnist database of handwritten digits, 1998.

Lu, Yichao and Foster, Dean P. Large scale canonical correlation analysis with iterative least squares. In *Advances in Neural Information Processing Systems*, pp. 91–99, 2014.

Ma, Zhuang, Lu, Yichao, and Foster, Dean P. Finding Linear Structure in Large Datasets with Scalable Canonical Correlation Analysis. In *Proceedings of the 32nd International Conference on Machine Learning*, JMLR Proceedings, 2015.

Marcus, Mitchell P, Marcinkiewicz, Mary Ann, and Santorini, Beatrice. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993.

Michaeli, Tomer, Wang, Weiran, and Livescu, Karen. Nonparametric Canonical Correlation Analysis. *CoRR*, abs/1511.04839, 2015.

Musco, Cameron and Musco, Christopher. Stronger approximate singular value decomposition via the block lanczos and power methods. *arXiv preprint arXiv:1504.05477*, 2015.

Nesterov, Yurii. A method of solving a convex programming problem with convergence rate o (1/k2). In *Soviet Mathematics Doklady*, volume 27, pp. 372–376, 1983.

Paul, Saurabh. Core-sets for canonical correlation analysis. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pp. 1887–1890. ACM, 2015.

Rokhlin, Vladimir, Szlam, Arthur, and Tygert, Mark. A randomized algorithm for principal component analysis. *SIAM Journal on Matrix Analysis and Applications*, 31 (3):1100–1124, 2009.

Rudelson, Mark and Vershynin, Roman. Non-asymptotic theory of random matrices: extreme singular values. *arXiv preprint arXiv:1003.2990*, 2010.

Sherry, Alissa and Henson, Robin K. Conducting and interpreting canonical correlation analysis in personality research: A user-friendly primer. *Journal of personality assessment*, 84(1):37–48, 2005.

Shewchuk, Jonathan R. An introduction to the conjugate gradient method without the agonizing pain. Technical report, Pittsburgh, PA, USA, 1994.

Spielman, Daniel A and Teng, Shang-Hua. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pp. 81–90. ACM, 2004.

Wang, Weiran and Livescu, Karen. Large-Scale Approximate Kernel Canonical Correlation Analysis. *CoRR*, abs/1511.04773, 2015.

Wang, Weiran, Arora, Raman, Livescu, Karen, and Srebro, Nathan. Stochastic optimization for deep CCA via nonlinear orthogonal iterations. volume abs/1510.02054, 2015.

Witten, Daniela M, Tibshirani, Robert, and Hastie, Trevor. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, pp. kxp008, 2009.