Efficient Globally Convergent Stochastic Optimization for Canonical Correlation Analysis

Authored by:

Weiran Wang
Dan Garber
Dan Garber
Nati Srebro
Jialei Wang

Abstract

We study the stochastic optimization of canonical correlation analysis (CCA), whose objective is nonconvex and does not decouple over training samples. Although several stochastic gradient based optimization algorithms have been recently proposed to solve this problem, no global convergence guarantee was provided by any of them. Inspired by the alternating least squares/power iterations formulation of CCA, and the shift-and-invert preconditioning method for PCA, we propose two globally convergent meta-algorithms for CCA, both of which transform the original problem into sequences of least squares problems that need only be solved approximately. We instantiate the meta-algorithms with state-of-the-art SGD methods and obtain time complexities that significantly improve upon that of previous work. Experimental results demonstrate their superior performance.

1 Paper Body

Canonical correlation analysis (CCA, [1]) and its extensions are ubiquitous techniques in scienti?c research areas for revealing the common sources of variability in multiple views of the same phenomenon. In CCA, the training set consists of paired observations from two views, denoted (x1 , y1), . . . , (xN , yN), where N is the training set size, xi ? Rdx and yi ? Rdy for i = 1, . . . , N . We also denote the data matrices for each view2 by X = [x1 , . . . , xN] ? Rdx ?N and Y = [y1 , . . . , yN] ? Rdy ?N , and d := dx + dy . The objective of CCA is to ?nd linear projections of each view such that the correlation between the projections is maximized: max u,v

u? ?xy v

s.t.

u? ?xx u = v? ?yy v = 1

where ?xy = N1 XY? is the cross-covariance matrix, ?xx = N1 XX? + ?x I and ?yy = ?y I are the auto-covariance matrices, and (?x , ?y) ? 0 are regularization parameters [2].

(1) 1 ? N YY +

We denote by (u? , v?) the global optimum of (1), which can be computed in closed-form. De?ne ?1

?1

T := ?xx2 ?xy ?yy2 ? Rdx ?dy ,

(2)

and let (?, ?) be the (unit-length) left and right singular vector pair associated with T?s largest singular value ?1 . Then the optimal objective value, i.e., the canonical correlation between the ?1 ?1 views, is ?1 , achieved by (u? , v?) = (?xx2 ?, ?yy2 ?). Note that

?1

?1 ?1 = kTk ? ?xx2 X

?yy2 Y ? 1. Furthermore, we are guaranteed to have ?1 ¡ 1 if (?x , ?y) ¿ 0. The ?rst two authors contributed equally. We assume that X and Y are centered at the origin for notational simplicity; if they are not, we can center them as a pre-processing operation. ?

2

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

Table 1: Time complexities of different algorithms for achieving ?-suboptimal solution (u, v) to

CCA, i.e., min (u? ?xx u?)2 , (v? ?yy v?)2 ? 1 ? ?. GD=gradient descent, AGD=accelerated GD, SVRG=stochastic variance reduced gradient, ASVRG=accelerated SVRG. Note ASVRG provides speedup over SVRG only when ? ? ¿ N , and we show the dominant term in its complexity. Algorithm Least squares solver Time complexity

?21 1 ? dN ? O ? ?2 ?? AppGrad [3] GD (local) 2 ? log ?

? 1 22 ? 1 1 ? dN ? CCALin [6] AGD ? ?2 ?? O 2 ? log ? 2 1

2

? 2 ? 2 1 1 ? O dN ? This work: AGD ? ?2 ??2 ? log ? 2 1

Alternating least

2 2 ?1 2 1 ? squares (ALS) SVRG O d(N + ? ?) ?2 ??2 ? log ? 2 1

2

? 2 ? 2 1 1 ? O d N? ASVRG ? ?2 ??2 ? log ?

? q 1 2 2 1 1 ? dN ? ? log O This work: AGD ? ?1 ?? 2

? Shift-and-invert 1 2 ? O d N + (? ? ?1 ??2) ? log2 ?1 SVRG preconditioning (SI)

q ? 2 1 1 ? dN 43 ? ? log O ASVRG ? ?1 ?? ? 2 For large and high dimensional datasets, it is time and memory consuming to ?rst explicitly form the matrix T (which requires eigen-decomposition of the covariance matrices) and then compute its singular value decomposition (SVD). For such datasets, it is desirable to develop stochastic algorithms that have ef?cient updates, converges fast, and takes advantage of the input sparsity. There have been recent attempts to solve (1) based on stochastic gradient descent (SGD) methods [3, 4, 5], but none of these work provides rigorous convergence analysis for their stochastic CCA algorithms. The main contribution of this paper is the proposal of two globally convergent meta-algorithms for solving (1), namely, alternating least squares (ALS, Algorithm 2) and shift-and-invert preconditioning (SI, Algorithm 3), both of which transform the original problem (1) into sequences of least squares problems that need only be solved approximately. We instantiate the meta algorithms with state-of-the-art SGD methods and obtain ef?cient stochastic optimization algorithms for CCA. In order to measure the alignments between an approximate solution (u, v) and the optimum (u? , v?), we assume that T has a positive singular value gap ? := ?1 ? ?2 ? (0, 1] so its top left and right singular vector pair is unique (up to a change of sign). Table 1 summarizes the time complexities of several algorithms for achieving ?-suboptimal alignmax max(kxi k2 , kyi k2) i ments, where ? ? = min(? is the upper bound of condition numbers of least squares min (?xx), ?min (?yy)) 3 ? to hide poly-logarithmic dependencies (see problems solved in all cases. We use the notation O(?) Sec. 3.1.1 and Sec. 3.2.3 for the hidden factors). Each time complexity may be preferrable in certain regime depending on the parameters of the problem. Notations We use ?i (A) to denote the i-th largest singular value of a matrix A, and use ?max (A) and ?min (A) to denote the largest and smallest singular values of A respectively.

2

Motivation: Alternating least squares

Our solution to (1) is inspired by the alternating least squares (ALS) formulation of CCA [7, Algorithm 5.2], as shown in Algorithm 1. Let the nonzero singular values of T be 1 ? ?1 ? ?2 ? ? ? ? ? ?r ¿ 0, where r = rank(T) ? min(dx , dy), and the corresponding (unit-length) left and right singular vector pairs be (a1 , b1), . . . , (ar , br), with a1 =? and b1 = ?. De?ne

0 T C= ? Rd?d . (3) T? 0 3

For the ALS meta-algorithm, its enough to consider a per-view conditioning. And when using AGD as the least squares solver, the time complexities dependends on ?max (?xx) instead, which is less than maxi ?xi ?2 .

2

Algorithm 1 Alternating least squares for CCA. Input: Data matrices X ? Rdx ?N , Y ? Rdy ?N , regularization parameters (?x , ?y). ? 0 ? Rdy . ? 0 ? Rdx , v Initialize u q p ? ?0 ? ?? ? 0? ?yy v ? 0/ u u , v ? v / ? v u0 ? u 0 0 0 xx 0 for t = 1, 2, . . . , T do ? t ? ??1 u xx ?xy vt?1

? ? t ? ??1 v yy ?xy ut?1 q p ? ?t ? ?? ? t? ?yy v ? t/ u u , v ? v / v ? ut ? u t t t xx t end for Output: (uT , vT) ? (u? , v?) as T ? ?.

o ? , ? ? ? 0 0

o n

?

? ? ? ?0 ? ?0 / ?0 , ? 0 ? ? 0 / ? 0 n

n

1
1
? ?2 ? ? ?xx ?xy ?yy2 ? t?1 ? t
o
o ? 21 ? ? 12 ? ? ?yy ? ? ? ? xx t t?1 xy
o n
?
? ? ? ?t ? ?t / ?t , ? t ? ? t / ? t n
{(?T , ? T) ? (?, ?)}

It is straightforward to check that the nonzero eigenvalues of C are: ?1 ? ? ? ? ? ?r ? ??r ? ? ? ? ? ??1 ,

a1 ar ar 1 1 1 ? ? ? with corresponding eigenvectors 2 , ..., 2 , 2 , ..., b1 br ?br

?1 2

a1 ?b1

.

The key observation is that Algorithm 1 effectively runs a variant of power iterations on C to extract its top eigenvector. To see this, make the following change of variables 1

1

1

1

2 2 2 2 ? = ?yy ? = ?xx ? t, ?t. u v ? (4) ut , ? t = ?yy vt , ? ?t = ?xx t t Then we can equivalently rewrite the steps of Algorithm 1 in the new variables as in {} of each line.

Observe that the iterates are updated as follows from step t ? 1 to step t:

? /——? ? —— ? 0 T ? ?t ?t?1 ? t t t (5) ? , ? ? —— . ? /——? ? ?t ? t?1 T? 0 ? ? t t t Except for the special normalization steps which rescale the two sets of variables separately, Algorithm 1 is very similar to the power iterations [8].

We show the convergence rate of ALS below (see its proof in Appendix A). The ?rst measure of ? 2 2 ? progress is the alignment of ?t to ? and the alignment of ? t to ?, i.e., (?? t ?) = (ut ?xx u) ? and (? t ?)2 = (vt? ?yy v?)2 . The maximum value for such alignments is 1, achieved when the iterates completely align with the optimal solution. The second natural measure of progress is the objective of (1), i.e., u? t ?xy vt , with the maximum value being ?1 .

? 2 ? ? 2 Theorem 1 (Convergence ofAlgorithm 1). Let ? := min (u? ¿ 0.4 0 ?xx u) , (v0 ?yy v)

2

?1 1 ? ? 2 ? ? 2 Then for t ? ? ?2 ?? ? 2 log ?? ?, we have in Algorithm 1 that min (ut ?xx u) , (vt ?yy v) 1

2

1 ? ?, and u? t ?xy vt ? ?1 (1 ? 2?).

Remarks We have assumed a nonzero singular value gap in Theorem 1 to obtain linear convergence in both the alignments and the objective. When there exists no singular value gap, the top singular vector pair is not unique and it is no longer meaningful to measure the alignments. Nonetheless, it is possible to extend our proof to obtain sublinear convergence for the objective in this case. Observe that, besides the steps of normalization to unit length, the basic operation in each iteration 1 ? ?1 1 ? ? t ? ??1 of Algorithm 1 is of the form u xx ?xy vt?1 = (N XX + ?x I) N XY vt?1 , which is equivalent to solving the following regularized least squares (ridge regression) problem N

2 ?x 2 ? x 1 1 X 1 ? 2 2 ? ?

u? X ? vt?1 min u xi ? vt?1 Y + yi + kuk ? min kuk . (6) u u 2N 2 N i=1 2 2 In the next section, we show that, to maintain the convergence of ALS, it is unnecessary to solve the least squares problems exactly. This enables us to use state-of-the-art SGD methods for solving (6) to suf?cient accuracy, and to obtain a globally convergent stochastic algorithm for CCA. 4

One can show that ? is bounded away from 0 with high probability using random initialization (u0 , v0).

3

Algorithm 2 The alternating least squares (ALS) meta-algorithm for CCA. Input: Data matrices X ? Rdx ?N , Y ? Rdy ?N , regularization parameters (?x , ?y). ? 0 ? Rdy . ? 0 ? Rdx , v Initialize u q p ? ? ? 0, ?0 ? ?0, ?? ? 0? ?yy v ?0 ? u ? 0/ u , v ? v / u0 ? u v0 ? v u ? v u 0 0 0 xx 0 for t = 1, 2, . . . , T do

2 ?x 1 2 ?

u? X ? vt?1 ? t?1 , and output Solve min ft (u) := kuk with initialization u Y + u 2N 2 ? t satisfying ft (? ut) ? minu ft (u) + ?. approximate solution u

2 ?y 1 2

v ? Y ? u ?

\+ ? t?1 , and output kvk with initialization v Solve min gt (v) := t?1 X v 2N 2 ? t satisfying gt (? vq) ? min g (v) + ?. approximate solution v t v t p ?? v ? ? ?? ? ? t/ u u , v ? v / v ? ut ? u t t xx t yy ? t t t end for Output: (uT , vT) is the approximate solution to CCA.

3 3.1

Our algorithms Algorithm I: Alternating least squares (ALS) with variance reduction

Our ?rst algorithm consists of two nested loops. The outer loop runs inexact power iterations while the inner loop uses advanced stochastic optimization methods, e.g., stochastic variance reduced gradient (SVRG, [9]) to obtain approximate matrix-vector multiplications. A sketch of our algorithm is provided in Algorithm 2. We make the following observations from this algorithm. Connection to previous work At step t, if we optimize ft (u) and gt (v) crudely by a single batch ? t?1), we obtain the following update rule: gradient descent step from the initialization (? ut?1 , v q ? t/ u ? t?1 ? Y? vt?1)/N, ? t?1 ? 2? X(X? u ?t ? u ?t ?? ut ? u u t ?xx u q ?t ? t? ?yy v ? t?1 ? X? ut?1)/N, ?t/ v ?t ? v ? t?1 ? 2? Y(Y? v vt ? v v where ? ¿ 0 is the stepsize (assuming ?x = ?y = 0). This coincides with the AppGrad algorithm of [3, Algorithm 3], for which only local convergence is shown. Since the objectives ft (u) and gt (v) decouple over training samples, it is convenient to apply SGD methods to them. This observation motivated the stochastic CCA algorithms of [3, 4]. We note however, no global convergence guarantee was shown for these stochastic CCA algorithms, and the key to our convergent algorithm is to solve the least squares problems to suf?cient accuracy. Warm-start Observe that for different t, the least squares problems ft (u) only differ in their targets as vt changes over time. Since vt?1 is close to vt (especially when near convergence), we may use ? t as initialization for minimizing ft+1 (u) with an iterative algorithm. u Normalization p At the end of each outer loop, Algorithm 2 implements exact normalization of the ? ? t/ u ? t = N1 (? ? t to ensure the constraints, where u ?? ?? form ut ? u u? u? t ?xx u t X)(? t X) + t ?xx u 2 ? ? t X. However, this does not inut k requires computing the projection of the training set u ?x k? troduce extra computation because we also compute this projection for the batch gradient used by SVRG (at the beginning of time step t + 1). In contrast, the stochastic algorithms of [3, 4] (possibly adaptively) estimate the covariance matrix from a minibatch of training samples and use the estimated covariance for normalization. This is because their algorithms perform normalizations after each update and thus need to avoid computing the projection of the entire training set frequently. But as a result, their inexact normalization steps introduce noise to the algorithms. Input sparsity For high dimensional sparse data (such as those used in natural language processing [10]), an advantage of gradient based methods over the closed-form solution is that the former takes into account the input sparsity. For sparse inputs, the time complexity of our algorithm depends on nnz(X, Y), i.e., the total number of nonzeros in the inputs instead of dN . Canonical ridge When (?x , ?y) ¿ 0, ft (u) and gt (v) are guaranteed to be strongly convex due to the ?2 regularizations, in which case SVRG converges linearly. It is therefore bene?cial to use 4

small nonzero regularization for improved computational ef?ciency, especially for high dimensional datasets where inputs X and Y are approximately low-rank. Convergence By the analysis of inexact power iterations where the least squares problems are solved (or the matrix-vector multiplications are computed) only up to necessary accuracy, we provide the following theorem for the convergence of Algorithm 2 (see its proof in Appendix B). The key to our analysis is to bound the distances between the iterates of Algorithm 2 and that of Algorithm 1 at all time steps, and when the errors of the least squares problems are suf?ciently small (at the level of ? 2), the iterates of the two algorithms have the same2quality. ?1 2 Theorem 2 (Convergence of Algorithm 2). Fix T ? ? ?2 ?? 2 log ?? ?, and set ?(T) ? 2 1

2 ? 2 ?2r (2?1 /?r)?1 in Algorithm 2. Then we have u? = vT? ?yy vT = 1, T ?xx uT 128 ? (2?1 /?r)T ?1

? 2 ? ? 2 ? 1 ? ?, and u? min (u? T ?xx u) , (vT ?yy v) T ?xy vT ? ?1 (1 ? 2?). 3.1.1 Stochastic optimization of regularized least squares We now discuss the inner loop of Algorithm 2, which approximately solves problems of the form (6). Owing to the ?nite-sum structure of (6), several stochastic optimization methods such as SAG [11], SDCA [12] and SVRG [9], provide linear convergence rates. All these algorithms can be readily applied to (6); we choose SVRG since it is memory ef?cient and easy to implement. We also apply the recently developed accelerations techniques for ?rst order optimization methods [13, 14] to obtain an accelerated SVRG (ASVRG) algorithm. We give the sketch of SVRG for (6) in Appendix C.

2 PN 2 Note that f (u) = N1 i=1 f i (u) where each component f i (u) = 21 u? xi ? v? yi + ?2x kuk 2 is kxi k -smooth, and f (u) is ?min (?xx)-strongly convex5 with ?min (?xx) ? ?x . We show in Appendix D that the initial suboptimality for minimizing ft (u) is upper-bounded by constant when using the warm-starts. We quote the convergence rates of SVRG [9] and ASVRG [14] below. ? satisfying6 E[f (? Lemma 3. The SVRG algorithm [9] ?nds a vector u u)] ? minu f (u) ? ? in time

2 maxi kxi k 1 O dx (N + ?x) log ? where ?x = ?min (?xx) . The ASVRG algorithm [14] ?nds a such solution

? in time O dx N ?x log 1? . Remarks As mentioned in [14], the acceleration version provides speedup over normal SVRG only when ?x ¿ N and we only show the dominant term in the above complexity. By combining the iteration complexity of the outer loop (Theorem 2) and the time complexity of the inner loop (Lemma 3), we obtain the total time complexity of

2 2

2 2 ? ? 2 1 1 ? d N ? 2?1 2 ? log2 1 ? d (N + ?) 2 2 ? log for ALS+SVRG and O for O ? ? ?1 ??2 ?1 ??2

2 maxi kyi k2 i kxi k ? hides poly-logarithmic depenALS+ASVRG, where ? := max max and O(?) ?min (?xx) , ?min (?yy) dences on ?1 and ?1r . Our algorithm does not require the initialization to be close to the optimum and converges globally. For comparison,thelocally convergent AppGrad has a time complexity

2 ? dN ?? 2?1 2 ? log 1 , where ?? := max ?max (?xx) , ?max (?yy) . Note, [3, Theorem 2.1] of O ? ?min (?xx) ?min (?yy) ? ?? 1

2

in this complexity, the dataset size N and the least squares condition number ?? are multiplied together because AppGrad essentially uses batch gradient descent as the least squares solver. Within our framework, we can use accelerated gradient descent(AGD, [15]) instead and obtain a globally

? ? 2 2 2 1 1 ? ? convergent algorithm with a total time complexity of O dN ? ?2 ??2 ? log ? . 1

3.2

2

Algorithm II: Shift-and-invert preconditioning (SI) with variance reduction

The second algorithm is inspired by the shift-and-invert preconditioning method for PCA [16, 17]. Instead of running power iterations on C as de?ned in (3), we will be running power iterations on ?1

?I ?T ?1 ? Rd?d , (7) M? = (?I ? C) = ?T? ?I 5

We omit the regularization in these constants, which are typically very small, to have concise expressions. The expectation is taken over random sampling of component functions. High probability error bounds can be obtained using the Markov?s inequality. 6

5

where ? ¿ ?1 . It is straightforward to check that M? is positive de?nite and its eigenvalues are: 1 1 1 1 ? ??? ? ? ??? ? ? ??? ? , ? ? ?1 ? ? ?r ? + ?r ? + ?1

a1 ar ar a1 , . . . , ?12 , . . . , ?12 , . . . , ?12 . with eigenvectors ?12 b1 br ?br ?b1

The main idea behind shift-and-invert power iterations is that when ? ? ?1 = c(?1 ? ?2) with c ? O(1), the relative eigenvalue gap of M? is large and so power iterations on M? converges quickly. Our shift-and-invert preconditioning (SI) meta-algorithm for CCA is sketched in Algorithm 3 (in Appendix E due to space limit) and it proceeds in two phases.

Phase I: shift-and-invert preconditioning for eigenvectors of M? ? and starting from an over-estimate of ?1 (1 + ? ? Using an estimate of the singular value gap ? suf?ces), the algorithm gradually shrinks ?(s) towards ?1 by crudely estimating the leading eigenvector/eigenvalues of each M?(s) along the way and shrinking the gap ?(s) ? ?1 , until we reach a ?(f) ? (?1 , ?1 + c(?1 ? ?2)) where c ? O(1). Afterwards, the algorithm ?xes ?(f) and runs inexact power iterations on M?(f) to obtain an accurate estimate of its leading eigenvector. Note " 1 # 2 ? u ? xx t in this phase, power iterations implicitly operate on the concatenated variables ?12 and 1 2 # " 1 ?t ?yy v 2 1 1 ut ?xx 2 2 ?1 and ?yy). in Rd (but without ever computing ?xx 1 2 2 ?yy vt Matrix-vector multiplication The matrix-vector multiplications in Phase I have the form

?1

??xx ??xy ?t u ?xx ut?1 ? , (8) ?t v ?yy vt?1 ??yy ??? xy where ? varies over time in order to locate ?(f) . This is equivalent to solving

1 ? ? ??xx ??xy ?t u u ? min ? u? ?xx ut?1 ? v? ?yy vt?1 . u v ?t v v ??? ??yy u,v 2 xy 3.2.1

And as in ALS, this least squares problem can be further written as ?nite-sum: N 1 X i min ht (u, v) = h (u, v) where (9) u,v N i=1 t

1 ? ? ? xi x? ?xi yi? u i + ?x I ? u? ?xx ut?1 ? v? ?yy vt?1 . u v hit (u, v) = v ?yi x? ? yi yi? + ?y I 2 i We could directly apply SGD methods to this problem as before. Normalization The normalization steps in Phase I have the form

q ? ?t ut u ?t, ? t? ?yy v ?t + v ?? ? 2 u t ?xx u ?t vt v and so the following remains true for the normalized iterates in Phase I: ? for t = 1, . . . , T. (10) u? t ?xx ut + vt ?yy vt = 2, Unlike the normalizations in ALS, the iterates ut and vt in Phase I do not satisfy the original CCA constraints, and this is taken care of in Phase II. We have the following convergence guarantee for Phase I (see its proof in Appendix F). ? := Theorem 4 (Convergence of Algorithm 3, Phase I). Let ? = ?1 ? ?2 ? (0, 1], and ?

1 ? ? ? ? 2 ? u ? u + v ? v ¿ 0, and ? ? [c ?, c ?] where 0 ¡ c ? c ? 1. Set xx yy 1 2 1 2 0 0 4

m2 ?1

m1 ?1 4 ? ? 1 ? 5 128 ? m1 = ?8 log 16 ? ? min 3084 in , 4?10 18 ? ? ?, m2 = ? 4 log ? ? ? 2 ?, and ? 18

Algorithm 3. Then the (uT , vT) output by Phase I of Algorithm 3 satis?es (10) and ?2 1 ? (uT ?xx u? + vT? ?yy v?)2 ? 1 ? , (11) 4 64

1 and the number of calls to the least squares solver of ht (u, v) is O log ?1? log ? + log ??1?2 . 6

3.2.2 Phase II: ?nal normalization In order to satisfy the CCA constraints, we perform a last normalization q q ? ? ? u T / u? ? u , v ? v / vT? ?yy vT . u xx T T T

(12)

?) as our ?nal approximate solution to (1). We show that this step does not cause And we output (? u, v much loss in the alignments, as stated below (see it proof in Appendix G). Theorem 5 (Convergence of Algorithm 3, Phase II). Let Phase I of Algorithm 3 outputs (uT , vT) ?) to (1) such that that satisfy (11). Then after (12), we obtain an approximate u, v solution (? ? ? ?xy v ? ? ?1 (1?2?). ? = 1, min (? ?=v ? ? ?yy v ? ? ?xx u v? ?yy v?)2 ? 1??, and u u? ?xx u?)2 , (? u 3.2.3 Time complexity We have shown in Theorem 4 that Phase I only approximately solves a small number of instances of (9). The normalization steps (10) require computing the projections of the traning set which are reused for computing batch gradients of (9). The ?nal normalization (12) is done only once and costs O(dN). Therefore, the time complexity of our algorithm mainly comes from solving the least squares problems (9) using SGD methods in a blackbox fashion. And the time complexity for SGD methods depends on the condition number of (9). Denote # #

" 12 " 21 ??xx ??xy ?I ?T ?xx ?xx Q? = = . (13) 1 1 ?T? ?I ??yy ??? 2 2 xy ?yy ?yy It is clear that

?max (Q?) ? (? + ?1) ? max (?max (?xx), ?max (?yy)) , ?min (Q?) ? (? ? ?1) ? min (?min (?xx), ?min (?yy)) .

We have shown in the proof of Theorem 4 that

?+?1 ???1

?

9 ? ?

?

9 c1 ?

Lemma 10, Appendix F.2), and thus the condtion number for AGD is max(?max (?xx), ?max (?yy)) min(?min (?xx), ?min (?yy)) .

throughout Algorithm 3 (cf. ?max (Q?) ?min (Q?)

?

9/c1 ?? , ?1 ??2 ?

where

? ? := For SVRG/ASVRG, the relevant condition number depends on the gradient Lipschitz constant of individual components. We show in Appendix H (Lemma 12) that the maxi max(kxi k2 , kyi k2) 1 relevant condition number is at most ?9/c . An interesting ? ? , where ? ? := ?? min(? 1 2 min (?xx), ?min (?yy)) issue for SVRG/ASVRG is that, depending on the value of ?, the independent components hit (u, v) may be nonconvex. If ? ? 1, each component is still guaranteed to by convex; otherwise, some PN components might be non-convex, with the overall average N1 i=1 hit being convex. In the later case, we use the modi?ed analysis of SVRG [16, Appendix B] for its time complexity. We use warmstart in SI as in ALS, and the initial suboptimality for each subproblem can be bounded similarly. ?

The total time complexities of our SI meta-algorithm are given in Table 1. Note that ? ? (or ? ??) 1 are multiplied together, giving the effective condition number. When using SVRG as and ?1 ??

2 2 1 1 ? the least squares solver, we obtain the total ? ?1 ??2) ? log ? + ? time complexity of O d(N 2 1 1 2 ? d(N + (?)) ? log otherwise. When usif all components are convex, and O ? ?1 ?? ? 2

? ? q 1 ? d N ? ? ?1 ?? ? log2 ?1 if all components are convex, and ing ASVRG, we have O 2

q 3? 1 ? hides poly-logarithmic dependences on 1 ? dN 4 ? ? ?1 ?? otherwise. Here O(?) ? log2 ?1 O ? ? 2 1 . It is remarkable that the SI meta-algorithm is able to separate the dependence of dataset size and ? N from other parameters in the time complexities.

Parallel work In a parallel work [6], the authors independently proposed a similar ALS algorithm7 , ?

and they solve the least squares problems using AGD. The time complexity of their algorithm for ex2 ? dN ?? 2?1 2 ? log 1 , which has linear dependence tracting the ?rst canonical correlation is O ?

?1 ??2 ?21 1 (so their algorithm is linearly convergent, but our complexity for ALS+AGD has on ?2 ?? 2 log ? 1

2

quadratic dependence on this factor), but typically worse dependence on N and ?? (see remarks in Section 3.1.1). Moreover, our SI algorithm tends to signi?cantly outperform ALS theoretically and

1 empirically. It is future work to remove extra log ? dependence in our analysis. 7

Our arxiv preprint for the ALS meta-algorithm was posted before their paper got accepted by ICML 2016.

7

?x = ?y = 10?5 ?? = 53340, ? = 5.345

?x = ?y = 10?4 ?? = 5335, ? = 4.924

0

Suboptimality

Mediamill

10

CCALin SI-AVR

-5

10-5

10 SI-AVR

ALS-AVR

ALS-VR

10-4

AppGrad

-5

10 SI-AVR

SI-VR
ALS-VR
ALS-AVR
10-10
10-10
-10
10
ALS-VR
ALS-VR
100
SI-VR
10-15
100
200
300
400
500
600
0
100
200
300
SI-VR
10-15
400
500
0
100
200
-15
300
400
500
600
0
?? = 34070, ? = 10.58
100
200
300
400
500
600
?? = 3416, ? = 9.082 0
S-AppGrad 10
100CCALin
AppGrad

ALS-VR
AppGrad
AppGrad
10-1
ALS-AVR
10
0
600
?? = 332800, ? = 11.10 10
CCALin
CCALin
SI-VR
ALS-AVR
0
JW11
S-AppGrad
S-AppGrad
SI-AVR
-2
10
-6
Suboptimality
AppGrad CCALin
?? = 2699000, ? = 11.22
CCALin
S-AppGrad
10-2
S-AppGrad
ALS-VR
SI-AVR -2
ALS-AVR
10-5
10
S-AppGrad
SI-VR
10
SI-AVR
SI-AVR
ALS-VR
10-4 SI-VR
-3
ALS-AVR
-5
10
CCALin
AppGrad

ALS-AVR
SI-AVR
-10
10
ALS-VR ALS-AVR
10-4
10-6 0
100
200
300
400
500
600
100
SI-VR
10-10
0
?? = 2235000, ? = 12.82
100
200
300
400
500
600
100
100
200
300
400
500
?? = 22350, ? = 12.30
100
200
300
400
500
600
?? = 2236, ? = 9.874 0
10
ALS-VR
AppGrad
S-AppGrad
AppGrad
AppGrad
S-AppGrad
CCALin

S-AppGrad
ALS-AVR
0
600
100
CCALin
SI-VR -15
10 0
?? = 223500, ? = 12.75
AppGrad
Suboptimality
0
10 S-AppGrad
AppGrad
AppGrad
10
MNIST
?x = ?y = 10?2 ?? = 54.34, ? = 2.548
0 10 S-AppGrad
0
10CCALin
?x = ?y = 10?3 ?? = 534.4, ? = 4.256
CCALin ALS-VR
10-2
CCALin
S-AppGrad
ALS-AVR
ALS-VR
-5
-5
10
-5
10
ALS-AVR
10
ALS-AVR
SI-AVR
10-4 ALS-VR
-10
10-10
-10
10
10 SI-VR
SI-VR
10-6
100

200
300
400
Passes
500
600
10-15
SI-AVR
10-15 0
100
200
300
400
500
600
Passes
0
100
SI-AVR
SI-VR
SI-VR
SI-AVR
0
200
300
400
Passes
500
600
10-15 0
100
200
300
400
500
600
Passes

Figure 1: Comparison of suboptimality vs. # passes for For each dataset and different algorithms. ?21 (?xx) ?max (?yy) , regularization parameters (?x , ?y), we give ?? = max ??max ?min (?yy) and ? = ?2 ??2 . min (?xx) 1
2

Extension to multi-dimensional projections To extend our algorithms to L-dimensional projections, we can extract the dimensions sequentially and remove the explained correlation from ?xy each time we extract a new dimension [18]. For the ALS meta-algorithm, a cleaner approach is to extract the L dimensions simultaneously using (inexact) orthogonal iterations [8], in which case the subproblems become multi-dimensional regressions and our normalization steps are of the form ? t)? 12 (the same normalization is used by [3, 4]). Such normalization involves ? t (U ? ? ?xx U Ut ? U t the eigenvalue decomposition of a L ? L matrix and can be solved exactly as we typically look for low dimensional projections. Our analysis for L = 1 can be extended to this scenario and the convergence rate of ALS will depend on the gap between ?L and ?L+1 .

4

Experiments

We demonstrate the proposed algorithms, namely ALS-VR, ALS-AVR, SI-VR, and SI-AVR, abbreviated as ?meta-algorithm ? least squares solver? (VR for SVRG, and AVR for ASVRG) on three real-world datasets: Mediamill [19] (N = 3 ? 104), JW11 [20] (N = 3 ? 104), and MNIST [21] (N = 6 ? 104). We compare our algorithms with batch AppGrad and its stochastic version s-AppGrad [3], as well as the CCALin algorithm in parallel work [6]. For each algorithm, we compare the canonical correlation estimated by the iterates at different number of passes over the data with that of the exact solution by SVD. For each dataset, we vary the regularization parameters ?x = ?y over {10?5 , 10?4 , 10?3 , 10?2 } to vary the least squares condition numbers, and larger regularization leads to better conditioning. We plot the suboptimality in objective vs. # passes for each algorithm in Figure 1. Experimental details (e.g. SVRG parameters) are given in Appendix I. We make the following observations from the results. First, the proposed stochastic algorithms signi?cantly outperform batch gradient based methods AppGrad/CCALin. This is because the least squares condition numbers for these datasets are large, and SVRG enable us to decouple dependences on the dataset size N and the condition number ? in the time complexity. Second, SI-VR converges faster than ALS-VR as it further decouples the dependence on N and the singular value gap of T. Third, inexact normalizations keep the s-AppGrad algorithm from converging to an accurate solution. Finally, ASVRG improves over SVRG when the the condition number is large. Acknowledgments Research partially supported by NSF BIGDATA grant 1546500. 8

2 References

[1] H. Hotelling. Relations between two sets of variates. Biometrika, 28(3/4):321?377, 1936. [2] H. D. Vinod. Canonical ridge and econometrics of joint production. J. Econometrics, 1976. [3] Z. Ma, Y. Lu, and D. Foster. Finding linear structure in large datasets with scalable canonical correlation analysis. In ICML, 2015. [4] W. Wang, R. Arora, N. Srebro, and K. Livescu. Stochastic optimization for deep CCA via nonlinear orthogonal iterations. In ALLERTON, 2015. [5] B. Xie, Y. Liang, and L. Song. Scale up nonlinear component analysis with doubly stochastic gradients. In NIPS, 2015. [6] R. Ge, C. Jin, S. Kakade, P. Netrapalli, and A. Sidford. Ef?cient algorithms for large-scale generalized eigenvector computation and canonical correlation analysis. arXiv, April 13 2016. [7] G. Golub

and H. Zha. Linear Algebra for Signal Processing, chapter The Canonical Correlations of Matrix Pairs and their Numerical Computation, pages 27?49. 1995. [8] G. Golub and C. van Loan. Matrix Computations. third edition, 1996. [9] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In NIPS, 2013. [10] Y. Lu and D. Foster. Large scale canonical correlation analysis with iterative least squares. In NIPS, 2014. [11] M. Schmidt, N. Le Roux, and F. Bach. Minimizing ?nite sums with the stochastic average gradient. Technical Report HAL 00860051, ?cole Normale Sup?rieure, 2013. [12] S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. Journal of Machine Learning Research, 2013. [13] R. Frostig, R. Ge, S. Kakade, and A. Sidford. Un-regularizing: Approximate proximal point and faster stochastic algorithms for empirical risk minimization. In ICML, 2015. [14] H. Lin, J. Mairal, and Z. Harchaoui. A universal catalyst for ?rst-order optimization. In NIPS, 2015. [15] Y. Nesterov. Introductory Lectures on Convex Optimization. A Basic Course. Springer, 2004. [16] D. Garber and E. Hazan. Fast and simple PCA via convex optimization. arXiv, 2015. [17] C. Jin, S. Kakade, C. Musco, P. Netrapalli, and A. Sidford. Robust shift-and-invert preconditioning: Faster and more sample ef?cient algorithms for eigenvector computation. 2015. [18] D. Witten, R. Tibshirani, and T. Hastie. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. Biostatistics, 2009. [19] C. Snoek, M. Worring, J. van Gemert, J. Geusebroek, and A. Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. In MULTIMEDIA, 2006. [20] J. Westbury. X-Ray Microbeam Speech Production Database User?s Handbook, 1994. [21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proc. IEEE, 86(11):2278?2324, 1998. [22] M. Warmuth and D. Kuzmin. Randomized online PCA algorithms with regret bounds that are logarithmic in the dimension. Journal of Machine Learning Research, 2008. [23] R. Arora, A. Cotter, K. Livescu, and N. Srebro. Stochastic optimization for PCA and PLS. In ALLERTON, 2012. [24] A. Balsubramani, S. Dasgupta, and Y. Freund. The fast convergence of incremental PCA. In NIPS, 2013. [25] O. Shamir. A stochastic PCA and SVD algorithm with an exponential convergence rate. In ICML, 2015. [26] F. Yger, M. Berar, G. Gasso, and A. Rakotomamonjy. Adaptive canonical correlation analysis based on matrix manifolds. In ICML, 2012.

9