Principal Component Projection Without Principal Component Analysis

Roy Frostig  
Stanford University  
RF@CS.STANFORD.EDU

Cameron Musco  
MIT  
CNMUSCO@MIT.EDU

Christopher Musco  
MIT  
CPMUSCO@MIT.EDU

Aaron Sidford  
Microsoft Research, New England  
ASID@MICROSOFT.COM

Abstract
We show how to efficiently project a vector onto the top principal components of a matrix, without explicitly computing these components. Specifically, we introduce an iterative algorithm that provably computes the projection using few calls to any black-box routine for ridge regression.

By avoiding explicit principal component analysis (PCA), our algorithm is the first with no runtime dependence on the number of top principal components. We show that it can be used to give a fast iterative method for the popular principal component regression problem, giving the first major runtime improvement over the naive method of combining PCA with regression.

To achieve our results, we first observe that ridge regression can be used to obtain a “smooth projection” onto the top principal components. We then sharpen this approximation to true projection using a low-degree polynomial approximation to the matrix step function. Step function approximation is a topic of long-term interest in scientific computing. We extend prior theory by constructing polynomials with simple iterative structure and rigorously analyzing their behavior under limited precision.

1. Introduction

In machine learning and statistics, it is common—often essential—to represent data in a concise form that decreases noise and increases efficiency in downstream tasks.

Perhaps the most widespread method for doing so is to project data onto the linear subspace spanned by its directions of highest variance—that is, onto the span of the top components given by principal component analysis (PCA). Computing principal components can be an expensive task, a challenge that prompts a basic algorithmic question:

Can we project a vector onto the span of a matrix’s top principal components without performing principal component analysis?

This paper answers that question in the affirmative, demonstrating that projection is much easier than PCA itself. We show that it can be solved using a simple iterative algorithm based on black-box calls to a ridge regression routine. The algorithm’s runtime does not depend on the number of top principal components chosen for projection, a cost inherent to any algorithm for PCA, or even algorithms that just compute an orthogonal span for the top components.

1.1. Motivation: principal component regression

To motivate our projection problem, consider one of the most basic downstream applications for PCA: linear regression. Combined, PCA and regression comprise the principal component regression (PCR) problem:

Definition 1.1 (Principal component regression (PCR)).
Let \( A \in \mathbb{R}^{n \times d} \) be a design matrix whose rows are data points and let \( b \in \mathbb{R}^n \) be a vector of data labels. Let \( A_\lambda \in \mathbb{R}^{n \times d} \) denote the result of projecting each row of \( A \) onto the span of the top principal components of \( A \), i.e. the eigenvectors of the covariance matrix \( \frac{1}{n} A^T A \) whose corresponding eigenvalue exceeds a threshold \( \lambda \). The task of PCR is to find a minimizer of the squared loss \( \| A_\lambda x - b \|_2^2 \). In other words, the goal is to compute \( A_\lambda^\dagger b \), where \( A_\lambda^\dagger \) is the Moore-Penrose pseudoinverse of \( A_\lambda \).

PCR is a key regularization method in statistics, numerical
We show that it is possible to sharpen the weak approximation given by ridge regression. Specifically, there exists a low degree polynomial \( p(\cdot) \) such that
\[
p((A^T A + \lambda I)^{-1} A^T A) y \text{ provides a very accurate approximation to } P_{A^*_\lambda} y \text{ for any } y \in \mathbb{R}^d.
\]
Moreover, the polynomial can be evaluated as a recurrence, which translates into a simple iterative algorithm: we can apply the sharpened approximation to a vector by repeatedly applying any ridge regression routine a small number of times.

Theorem 1.2 (Principal component regression without PCA). Given \( A \in \mathbb{R}^{m \times d} \) and \( y \in \mathbb{R}^d \), Algorithm 1 uses \( O(\gamma^{-2} \log(1/\epsilon)) \) approximate applications of \( (A^T A + \lambda I)^{-1} \) and returns \( x \) with \( \|x - P_{A^*_\lambda} y\|_2 \leq \epsilon \|y\|_2 \).

Like all iterative PCA algorithms, our running time scales inversely with \( \gamma \), the spectral gap around \( \lambda \).\(^1\) Notably, it does not depend on the number of principal components in \( A^*_\lambda \), a cost incurred by any method that applies the projection \( P_{A^*_\lambda} \) directly, either by explicitly computing the top principal components of \( A \), or even by just computing an orthogonal span for these components.

As mentioned, the above theorem also yields an algorithm for principal component regression that computes \( A_{\lambda}^* b \) without finding \( A_{\lambda} \). We achieve this result by introducing a robust reduction, from projection to PCR, that again relies on ridge regression as a computational primitive.

Corollary 1.3 (Principal component regression without PCA). Given \( A \in \mathbb{R}^{n \times d} \) and \( b \in \mathbb{R}^n \), Algorithm 2 uses \( O(\gamma^{-2} \log(1/\epsilon)) \) approximate applications of \( (A^T A + \lambda I)^{-1} \) and returns \( x \) with \( \|x - A_{\lambda}^* b\|_{A^*_\lambda} \leq \epsilon \|b\|_2 \).

Corollary 1.3 gives the first known algorithm for PCR that avoids the cost of principal component analysis.

1.4. Related work

A number of papers attempt to alleviate the high cost of principal component analysis when solving PCR. It has

\(^1\)See Section 3.2 for a discussion of this gap dependence. Aside from a full SVD requiring \( O(nd^2) \) time, any PCA algorithm giving the guarantee of Theorem 1.2 will have a dependence both on \( \gamma \) and on the number principal components in \( A_{\lambda} \). However, the \( \gamma \) dependence can be better – \( \gamma^{-1/2} \) for Krylov methods (Musco & Musco, 2015), giving a runtime tradeoff.
been shown that an approximation to $A_\lambda$ suffices for solving the regression problem (Chan & Hansen, 1990; Boutsidis & Magdon-Ismail, 2014). Unfortunately, even the fastest approximations are much slower than routines for ridge regression and inherently incur a linear dependence on the number of principal components above $\lambda$.

More closely related to our approach is work on the matrix sign function, an important operation in control theory, quantum chromodynamics, and scientific computing in general. Approximating the sign function often involves matrix polynomials similar to our “sharpening polynomial” that converts ridge regression to principal component projection. Significant effort addresses Krylov methods for applying such operators without computing them explicitly (van den Eshof et al., 2002; Frommer & Simoncini, 2008).

Our work differs from these methods in an important way: since we only assume access to an approximate ridge regression algorithm, it is essential that our sharpening step is robust to noise. Our iterative polynomial construction allows for a complete and rigorous noise analysis that is not available for Krylov methods, while at the same time eliminating space and post-processing costs. Iterative approximations to the matrix sign function have been proposed, but lack rigorous noise analysis (Higham, 2008).