Robust Principal Component Analysis - Review

Rupande Shastri

February 2017

Abstract - *Assuming a data matrix that can be written as a sum of a low-rank matrix and a sparse matrix, the paper gives an efficient method to separate the low-rank and sparse component of the given data matrix. The separation problem can be reduced to a problem called Principal Component Persuit(PCP) which when solved results in the exact recovery of the low-rank and sparse matrices. The optimization algorithm utilizing ALM (Augmented Lagrange Multiplier) is used to solve PCP. This method of computing Principal Components is less affected by outliers than the classical method Principal Component Analysis and hence is used in a wide range of applications involving large data matrices prone to gross corruption.*

Introduction

In recent times, an enormous amount of data gets generated, transferred, retrieved and processed every single day. With the ever increasing demand for processed data to be accessible faster, simplifying the process of decorrupting, analyzing and processing data is now vital. PCA is currently the most useful tool to simplfy the processing of large data matrices which are a mixture of useful information and corrupted and irrelevant components.

The classical method for finding PCA involves unrealistic assumptions that the given data matrices has low rank. This makes the separation

$$M = L_0 + N_0$$

where L_0 has low rank and N_0 is the noise matrix, assumed to be independent and normally distributed. The limitation of this method lies in the assupmtion that N_0 is normally distributed. A single outlier of large value in the noise matrix may lead to significant error in calculation of L_0. The method of Robust PCA deals with highly corrupted data by assuming the data matrix to be a sum of a low rank matrix with useful data and a sparse matrix of changes and corrupted data -

$$M = L_0 + S_0$$

Here, the sparse component S_0 can have values of large magnitudes, provided that the data is sparsely distributed.

The applications of Robust PCA are many, including video surveillance, face recognition, foreground-background separation, latent semantic indexing, ranking and collaborative filtering.

Assumptions made

1. The given matrix M has a distinct low-rank component and a sparse component.
2. The low-rank component L0 is not sparse.
3. The sparse component S0 is not low-rank. For this we assume the sparsity pattern of the sparse component to be of uniform random distribution.
4. The rank of the low-rank component is not too large, and the sparse component is reasonably sparse.

Algorithm

The Principal Component Pursuit problem,

$$minimize||L|| * +\lambda||S||1$$

$$subjecttoL + S = M$$

for small problem sizes has many off the shelf algorithms available. For large problem sizes these are inconvenient since the complexity of these algorithms approximates to $O(n^6)$.

Another algorithm using the concept of iterative thresholding shrinks the singular values at every iteration to minimize the nuclear norm of L_0. The complexity of each iteration is that of an SVD operation which is close to $O(n^3)$. However, it converges slowly and needs upto 10^4 iterations to recover an almost exact L_0.

An APG (Accelerated Proximal Gradient) algorithm was proposed in 2009, having an optimal convergence rate $O(1/k^2)$ and performing 50 times faster than iterative thresholding. The algorithm proves to have convergence and accuracy issues in generic cases of the problem.

In the paper, the PCP problem is solved using an Augmented Lagrange Multiplier (ALM). We minimize the Lagrange function during each iteration, setting new values for L and S each time using the shrinkage operator S_τ for S

$$S_\tau[x] = sgn(x)max(|x|\tau, 0)$$

and the singular value thresholding operator D_τ for L

$$D_\tau(X) = US_\tau(X)V^T$$

ROBUSTPCA Algorithm:

1. initialize: $S_0 = Y_0 = 0, \mu > 0$
2. **while** (not converged) **do**
3. compute $L_{k+1} = D_{1/\mu}(MS_k + 1/\mu Y_k)$;
4. compute $S_{k+1} = S_{\lambda/\mu}(ML_{k+1} + 1/\mu Yk)$;
5. compute $Y_{k+1} = Y_k + \mu(ML_{k+1}S_{k+1})$;
6. **end while**
7. **output:** L, S;

The above algorithm is a special use of ALM called Alternating Directions Method. The step that takes most of the computation time in the algorithm is the calculation of $L_k + 1$ using singular value thresholding. Singular values exceeding the threshold mu are kept along with their corresponding singular vectors. Hence, the accuracy and time taken are determined by the value of threshold μ and the stopping criterion.

References

1. Robust Principal Component Analysis(2011) - Candes, Li, Ma, Wright