Principal Component Analysis

A neural network for Robust Principal Component Analysis

Eric, Mirko, Simon, Aron, Bastian

February 6, 2021

Table of Contents

Table of Contents

Intuition

- Observing a set of points in $\mathbb{R}^n$
- Think of fitting an n-dimensional ellipsoid to the data
- Each axis of the ellipsoid represents a principal component
- Used for dimensionality reduction
- To group the main characteristics of the data
- Makes further calculations more efficient

Axes of an Ellipsoid

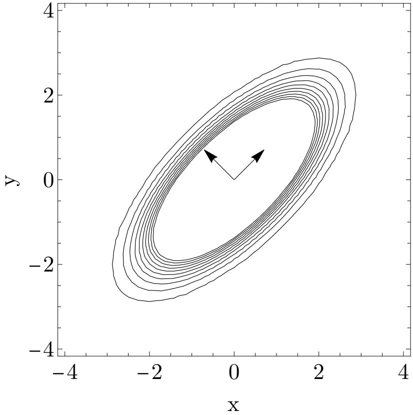


Figure: contour lines of a binormal distribution

Dimensionality Reduction

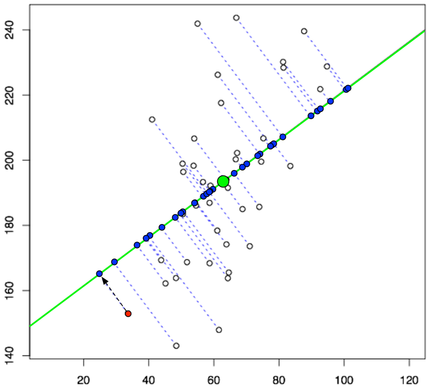


Figure: principal components lie in 1-dimensional subspace

Mathematical Approach

- Observing a set of points in $\mathbb{R}^n$
- Sequence of n direction vectors
- i^{th} vector is direction of a line that best fits the data
- i.e. minimizes average squared distance from points to line
- And being orthogonal to first $i-1$ vectors
- Directions form orthonormal basis of $\mathbb{R}^n$
- Eigenvectors of datas Covariance Matrix

Robust PCA

- Modification of PCA for matrices with faulty entries
- Matrix might be corrupted by imprecise measurements
- Observe matrix $M \in \mathbb{R}^{m \times n}$
- Find decomposition $M = L + S$ where
- L is of low rank
- S is sparse (lots of entries are zero)
- Purpose of S is to filter out corrupted entries

Table of Contents

Principal Component Pursuit

- There are several state of the art algorithms for solving RPCA
- One of them is the so called Principal Component Pursuit (PCP)
- Solves PCA sufficiently 3exact under certain constraints
- Using Singular Value Decomposition and Langrange Multiplier
- Computationally demanding
- Impractical in some applications
- Due to exactness later used for Testing

Table of Contents

Introducing Denise

- Feed forward deep neural network
- Trained on synthetic Dataset
- Solve RPCA for n-by-n symmetric semi-definite matrices
- Learns the decomposition map $M \to L + S$ directly
- Calculates the desired decomposition instantaneously
- Comparable to state-of-the-art algorithms in terms of quality
- Outperforms other algorithms in terms of computation time
- Therefore saves valuable time

Denise

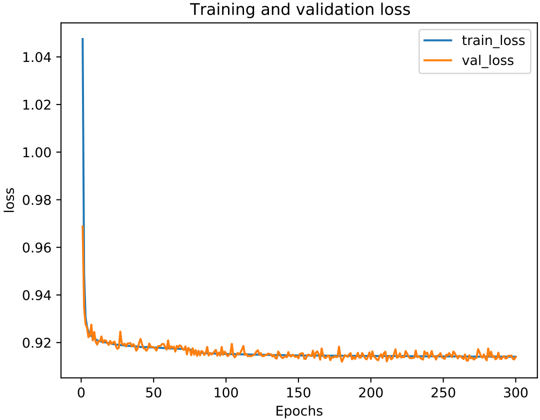
- Input: symmetric positive semi-definite matrix $M \in \mathbb{R}^{n \times n}$
- Goal: $M = L + S$, Note: L is assumed to be symmetric
- By Cholesky decomposition $M = UU^T + S$
- Lossfunction to be minimized is $\|UU^T - M\|_1 = \|S\|_1$
- Input matrix is transformed into a vector and reduced due to symmetry
- Network consists of three hidden layers and output layer
- Each hidden layer has ReLU-activation function and $2 * n$ nodes

Table of Contents

Testing

- Denise is trained on 600000 synthetic matrices
- Once trained Denise is tested on several matrices
- Portfolio Correlations from Dax 30
- We collected ten stocks from Yahoo! Finance
- Empirical correlation matrices are determined retrospectively
- We obtain 472 10-by-10 correlation matrices
- Correlation matrix of personality features
- 25 personality self report items from 2800 individuals
- Aim is to group items into five categories
- PCP is used as benchmark

Finance Data



Finance Data

Figure: Denise

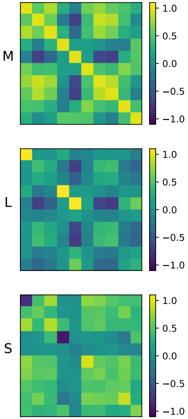
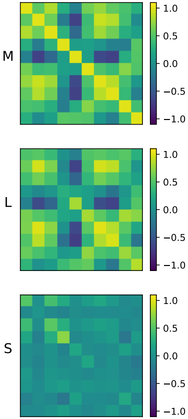
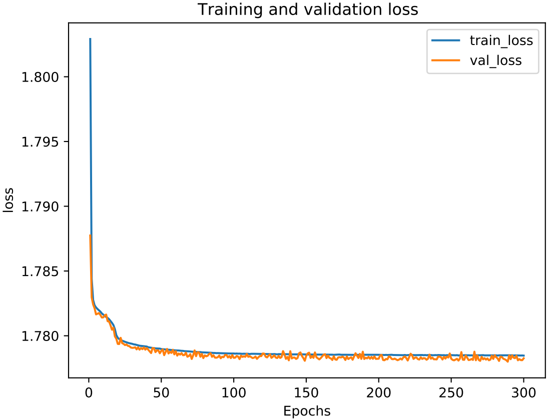


Figure: PCP



Personality Data



Personality Data

Figure: Denise

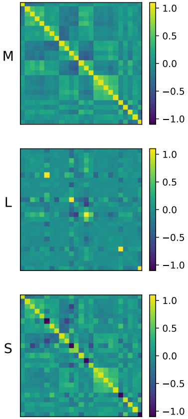


Figure: PCP

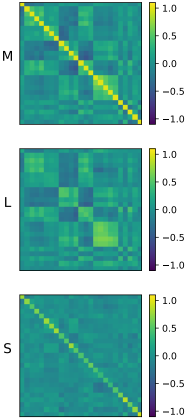


Table of Contents

RSVD for arbitrary matrices

- Using Neural Network for Singular Value Decomposition
- Aim is to find $M = UV^T$ for arbitrary $M \in \mathbb{R}^{n \times m}$ and $U \in \mathbb{R}^{n \times k}, V \in \mathbb{R}^{m \times k}$
- Rely on a collaborative network approach
- Training two Neural Networks $\mathcal{N}_U, \mathcal{N}_V$ in alternating manner
- One for each mapping $M \to U, M \to V$
- By minimizing the loss $\|UV^T - M\|_{\ell_1}$
- Successively optimizing weights of one network by keeping weights of the others fixed

Testing the RSVD