Seminar in Data Science

Lecture 4: PCA and and Factor Models

Laurent El Ghaoui

Seminar in Data Science and Information Technology, Summer 2020
TBSI – UC Berkeley

7/20/2020

Outline

Motivation

Linear Algebra Recap
- Eigenvalues
- Singular values

Principal Component Analysis
- Overview
- Deflation: iterated variance maximization
- Example

Low-rank approximations
- Problem
- Link with PCA
- Explained variance and approximation error
- SVD-based Auto-Encoder
- Factor models

Extensions
- Robust PCA
- Sparse PCA

Outline

Motivation

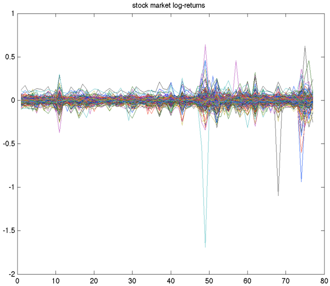
Linear Algebra Recap
- Eigenvalues
- Singular values

Principal Component Analysis
- Overview
- Deflation: iterated variance maximization
- Example

Low-rank approximations
- Problem
- Link with PCA
- Explained variance and approximation error
- SVD-based Auto-Encoder
- Factor models

Extensions
- Robust PCA
- Sparse PCA

Motivation



Daily log-returns of 77 Fortune 500 companies, 1/2/2007—12/31/2008.

High-dimensional data does not make any sense! (Other than tell us: returns are approximately zero . . .)

In this lecture:

- start with a classical unsupervised learning to obtain insights
- examine a newer method that improves interpretability

Outline

Motivation

Linear Algebra Recap
- Eigenvalues
- Singular values

Principal Component Analysis
- Overview
- Deflation: iterated variance maximization
- Example

Low-rank approximations
- Problem
- Link with PCA
- Explained variance and approximation error
- SVD-based Auto-Encoder
- Factor models

Extensions
- Robust PCA
- Sparse PCA

Eigenvalue decomposition for symmetric matrices

Theorem (EVD of symmetric matrices)

We can decompose any symmetric $n \times n$ matrix S as

$$S = U \Lambda U^T = \sum_{i=1}^{n} \lambda_i u_i u_i^T ,$$

where $\Lambda = \mathbf{diag}(\lambda_1, \ldots, \lambda_n)$, with $\lambda_1 \geq \ldots \geq \lambda_n$ the eigenvalues, and $U = [u_1, \ldots, u_n]$ is a $n \times n$ orthogonal matrix ($U^T U = I_n$) that contains the eigenvectors u_i of S, that is:

$$S u_i = \lambda_i u_i, \quad i = 1, \ldots, n.$$

Corollary: If S is square, symmetric:

$$\lambda_{\max}(S) = \max_{x \,:\, \|x\|_2 = 1} x^T S x. \tag{1}$$

Positive semi-definite (PSD) matrices

A (square) symmetric matrix S is said to be *positive semi-definite* (PSD) if

$$\forall\, x, \quad x^T S x \geq 0.$$

In this case, we write $S \succeq 0$.

From EVD theorem: for any square, symmetric matrix S:

$$S \succeq 0 \iff \text{every eigenvalue of } S \text{ is non-negative.}$$

Hence we can numerically (via EVD) check positive semi-definiteness.

Singular Value Decomposition (SVD)

Theorem (SVD of general matrices)

We can decompose any non-zero $n \times m$ matrix X as

$$X = \sum_{i=1}^{r} \sigma_i u_i v_i^T = U \Sigma V^T, \quad \Sigma = \mathbf{diag}(\sigma_1, \ldots, \sigma_r, 0, \ldots, 0) \in \mathbf{R}^{n \times m}$$

where $\sigma_1 \geq \ldots \geq \sigma_r > 0$ are the singular values, and

$$U = [u_1, \ldots, u_n], \quad V = [v_1, \ldots, v_m]$$

are square, orthogonal matrices ($U^T U = I_n$, $V^T V = I_m$). The number $r \leq \min(m, n)$ (the number of non-zero singular values) is called the rank *of X.*

The first r columns of U, V contains the left- and right singular vectors of X, respectively, that is:

$$X v_i = \sigma_i u_i, \quad X^T u_i = \sigma_i v_i, \quad i = 1, \ldots, r.$$

Links between EVD and SVD

The SVD of a $n \times m$ matrix X is related to the EVD of a (PSD) matrix related to X.

If $X = U\Sigma V^T$ is the SVD of X, then

- The EVD of XX^T is $U\Lambda U^T$, with $\Lambda = \Sigma^2$.
- The EVD of X^TX is $V\Lambda V^T$.

Hence the left (resp. right) singular vectors of X are the eigenvectors of the PSD matrix XX^T (resp. X^TX).

Computing SVD

Power iteration algorithm

For a large, sparse matrix X, we can find left and right singular vectors corresponding to the largest singular value of X with the *power iteration* algorithm:

$$u \to \frac{Xv}{\|Xv\|_2}, \quad v \to \frac{X^T u}{\|X^T u\|_2}.$$

This converges (for arbitrary initial u, v) under mild conditions on X.

Interpretation: power iteration can be obtained by solving

$$\min_{p,q} \|X - pq^T\|_F$$

alternatively over p, q (in the algorithm above, u, v are just normalized versions of p, q).

Similar efficient algorithm when X is centered (thus, not necessarily sparse, even if data is).

Outline

Motivation

Linear Algebra Recap
- Eigenvalues
- Singular values

Principal Component Analysis
- Overview
- Deflation: iterated variance maximization
- Example

Low-rank approximations
- Problem
- Link with PCA
- Explained variance and approximation error
- SVD-based Auto-Encoder
- Factor models

Extensions
- Robust PCA
- Sparse PCA

Principal Component Analysis

Overview

Principal Component Analysis (PCA) originated in psychometrics in the 1930's. It is now widely used in

- Exploratory data analysis.
- Simulation.
- Visualization.

Application fields include

- Finance, marketing, economics.
- Biology, medecine.
- Engineering design, signal compression and image processing.
- Search engines, data mining.

Solution principle of PCA

PCA finds “principal components” (PCs), *i.e.* orthogonal directions of *maximal variance* .

- PCs are computed via EVD of covariance matrix.
- Alternatively, PCs can be found directly via SVD of (centered) data matrix.
- Can be interpreted as a “factor model” of original data matrix.

Applications in finance:

- General understanding of market data.
- Underlies many theoretical models (such as CAPM).
- Modeling of term structure of interest rates.
- Portfolio hedging and immunization.
- Risk analysis, scenario generation.
- Obtain speed-ups in some portfolio optimization problems.

Variance maximization problem

Let C be the (empirical) $n \times n$ covariance matrix. *Variance maximization problem:*

$$\max_u \ u^T C u \ : \ \|u\|_2 = 1.$$

Assume that the EVD of C is given:

$$C = \sum_{i=1}^{n} \lambda_i u_i u_i^T = U \Lambda U^T,$$

with $\Lambda = \mathbf{diag}(\lambda)$, $\lambda_1 \geq \ldots \lambda_n$, and $U = [u_1, \ldots, u_n]$ is orthogonal ($U^{-1} = U^T$). Then a solution to the problem

$$\max_{u \,:\, \|u\|_2 = 1} \ u^T C u$$

is $u^* = u_1$, with u_1 an eigenvector of C that corresponds to its largest eigenvalue λ_1.

Alternatively, u_1 can be found directly via SVD of (centered) data matrix (see later).

Proof

We have

$$u^T C u = u^T U \Lambda U^T u = v^T \Lambda v, \quad v := U^T u.$$

Noting that, since $U^T = U^{-1}$:

$$v^T v = u^T U U^T u = u^T u,$$

and defining $p_i := v_i^2$, $i = 1, \ldots, n$:

$$\begin{aligned} \max_{u \,:\, u^T u = 1} u^T C u &= \max_{v \,:\, v^T v = 1} v^T \Lambda v \\ &= \max_{p \geq 0,\, p^T \mathbf{1} = 1} p^T \lambda \\ &= \max_{1 \leq i \leq n} \lambda_i. \end{aligned}$$

Finding orthogonal directions

A deflation method

Once we've found a direction with high variance, can we repeat the process and find other ones?

Deflation method:

- Project data points on the subspace orthogonal to the direction we found.
- Find a direction of maximal variance for projected data.

The process stops after *n* steps (*n* is the dimension of the data), but can be stopped earlier (to find only *k* directions, with $k << n$).

Finding orthogonal directions

Result

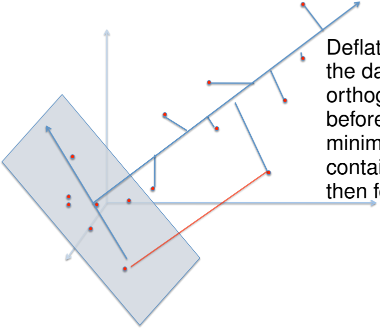
It turns out that the direction that solves

$$\max_x \ \mathbf{var}(x) \ : \ x^T u_1 = 0$$

is u_2, an eigenvector corresponding to the second-to-largest eigenvalue.

After k steps of the deflation process, the directions returned are $u_1, \ldots, u_k$. Thus we can compute k directions of largest variance in *one* eigenvalue decomposition of the covariance matrix.

Geometry of deflation



Deflation consists in projecting the data on a hyperplane orthogonal to the line found before; a new minimum-distance line contained in the hyperplane is then found.

Measuring quality

How well is data approximated by its projections on the successive subspaces?

Approach: compare sum of variances contained in the k directions found, with total variance.

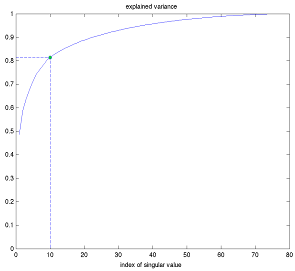
Explained variance: measured by the ratio

$$\frac{\lambda_1 + \ldots + \lambda_k}{\lambda_1 + \ldots + \lambda_n} = \frac{\sigma_1^2 + \ldots + \sigma_k^2}{\sigma_1^2 + \ldots + \sigma_n^2},$$

where $\lambda_1 \geq \ldots \geq \lambda_n$ are the eigenvalues of the covariance matrix, and $\sigma_1 \geq \ldots \geq \sigma_n$ are the singular values of the (centered) data matrix.

Example

PCA of market data



Data: Daily log-returns of 77 Fortune 500 companies, 1/2/2007—12/31/2008.

- Plot shows the eigenvalues of covariance matrix in decreasing order.
- First ten components explain 80% of the variance.
- Largest magnitude of eigenvector for 1st component correspond to financial sector (FABC, FTU, MER, AIG, MS).

Outline

Motivation

Linear Algebra Recap
- Eigenvalues
- Singular values

Principal Component Analysis
- Overview
- Deflation: iterated variance maximization
- Example

Low-rank approximations
- Problem
- Link with PCA
- Explained variance and approximation error
- SVD-based Auto-Encoder
- Factor models

Extensions
- Robust PCA
- Sparse PCA

Low-rank approximation of a matrix

For a given $n \times m$ matrix X, and integer $k \leq m, n$, the *k-rank approximation* problem is

$$\hat{X}^{(k)} := \arg\min_{\hat{X}} \ \|\hat{X} - X\|_F \ : \ \textbf{Rank}(\hat{X}) \leq k,$$

where $\|\cdot\|_F$ is the Frobenius norm (Euclidean norm of the vector formed with all the entries of the matrix). The solution is

$$\hat{X}^{(k)} = \sum_{i=1}^{k} \sigma_i u_i v_i^T,$$

where

$$X = U\Sigma V^T = \sum_{i=1}^{r} \sigma_i u_i v_i^T$$

is an SVD of the matrix X.

Low-rank approximation

Interpretation: rank-one case

Assume data matrix $X \in \mathbf{R}^{n \times m}$ represents time-series data (each column is a time-series):

$$X = \begin{pmatrix} x_1 & \cdots \\ x_m & \end{pmatrix}, \quad x_i \in \mathbf{R}^n, \;\; 1 \le i \le m.$$

Assume also that X is rank-one, that is, $X = uv^T \in \mathbf{R}^{n \times m}$, where u, v are vectors. Then

$$x_j(t) = X_{t,j} = \sigma_1 u(t) v(j), \;\; 1 \le j \le m, \;\; 1 \le t \le n.$$

Thus, each time-series is a “scaled” copy of the time-series represented by u, with scaling factors given in v. We can think of u as a “factor” that drives all the time-series.

Geometry: if a data matrix is rank-one, then all the data points are on a single line.

Low-rank approximation

Interpretation: low-rank case

When A is rank k, that is,

$$X = USV^T, \quad U \in \mathbf{R}^{n \times k}, \quad S = \textbf{diag}(\sigma_1, \ldots, \sigma_k) \in \mathbf{R}^{k \times k}, \quad V \in \mathbf{R}^{m \times k},$$

we can express the j-th column of X as

$$x_j(t) = \sum_{i=1}^{k} \sigma_i u_i(t) v_i(j), \quad 1 \leq t \leq n \;\; 1 \leq j \leq m.$$

Thus, each time-series is the sum of scaled copies of k time-series represented by $v_1, \ldots, v_k$, with scaling factors given in $u_1, \ldots, u_k$.

We can think of v_i's as the few “factors” that drive all the time-series.

PCA and low-rank approximations

PCA can be obtained directly (without forming the covariance matrix) via a *low-rank approximation* to the centered data matrix X_c:

$$X_c \approx \hat{X}_c^{(k)} := \sum_{i=1}^{k} \sigma_i u_i v_i^T$$

Each v_i is a particular factor, and u_i's contain scalings.

That is, $\hat{X}_c^{(k)}$ solves the problem

$$\arg\min_{\hat{X}} \|\hat{X} - X_c\|_F \ : \ \mathbf{Rank}(\hat{X}) \leq k,$$

where $\|X\|_F^2$ is the sum of the squares of the entries of X.

Low-rank approximation of covariance matrix

PCA also (implicitly) forms a low-rank approximation of the empirical covariance matrix.

The corresponding approximate covariance matrix is rank k:

$$C \approx C^{(k)} := U\Lambda^{(k)}U^T = FF^T$$

where $\Lambda^{(k)} = \textbf{diag}(\lambda_1, \ldots, \lambda_k, 0, \ldots, 0)$, with $\lambda_i = \sigma_i^2$, and

$$F = U^{(k)} \, \textbf{diag}(\sigma_1, \ldots, \sigma_k),$$

where $U^{(k)}$ contains the first k columns of U.

Explained variance and approximation error

Recall the *explained variance* ratio

$$\frac{\lambda_1 + \ldots + \lambda_k}{\lambda_1 + \ldots + \lambda_n} = \frac{\sigma_1^2 + \ldots + \sigma_k^2}{\sigma_1^2 + \ldots + \sigma_n^2},$$

The explained variance ratio is related to the relative approximation error:

$$\frac{\|\hat{X}_c^{(k)} - X_c\|_F^2}{\|X_c\|_F^2} = 1 - \frac{\sigma_1^2 + \ldots + \sigma_k^2}{\sigma_1^2 + \ldots + \sigma_p^2}.$$

We can also express the above in terms of the eigenvalues of the covariance matrix C

$$\frac{\mathbf{Tr}(C - C^{(k)})}{\mathbf{Tr}\, C} = \frac{\lambda_{k+1} + \ldots + \lambda_n}{\lambda_1 + \ldots + \lambda_n}.$$

Low-dimensional representation of data

Given a $n \times m$ data matrix $X = [x_1, \ldots, x_m]$, and the SVD-based rank-k approximation $\tilde{X}$ to X, we have

$$X = \sum_{i=1}^{r} \sigma_i u_i v_i^T = U \Sigma V^T \approx \tilde{X} = \sum_{i=1}^{k} \sigma_i u_i v_i^T = \tilde{U} \tilde{\Sigma} \tilde{V}^T,$$

where $\tilde{U} = [u_1, \ldots, u_k] \in \mathbf{R}^{n \times k}$, $\tilde{V} = [v_1, \ldots, v_k] \in \mathbf{R}^{m \times k}$, $\tilde{\Sigma} = \mathbf{diag}(\sigma_1, \ldots, \sigma_k) \in \mathbf{R}^{k \times k}$.

Thus: for any data point x_j

$$x_j = Xe_j \approx x_j' := \tilde{X} e_j = \left(\sum_{i=1}^{k} \sigma_i u_i v_i^T \right) e_j = \sum_{i=1}^{k} (\sigma_i v_i^T e_j) u_i = \tilde{U} h_j,$$

where

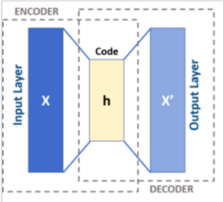
$$h_j := (\sigma_i v_i^T e_j)_{1 \le i \le k} = \tilde{\Sigma} \tilde{V}^T e_j \in \mathbf{R}^k.$$

Vector h_j is a low-dimensional representation of data point x_j.

SVD-based auto-encoder

For any data point:

$$x_j \approx \tilde{X} e_j = \tilde{U} h_j, \quad h_j := \tilde{\Sigma} \tilde{V}^T e_j \in \mathbf{R}^k.$$



- Thus any data point x_j can be encoded as a low-dimensional vector $h_j = \tilde{\Sigma} \tilde{V}^T e_j$.
- Given a low-dimensional representation h of x, we can decode , *i.e.* go back to the approximation x', via $x' = \tilde{U} h$.

The SVD-based low-dimensional representation is a special case of an ”auto-encoder”.

Stochastic interpretation of low-rank approximations

Assume that the (*e.g.*, price) observations y are the generated by stochastic model (here $F \in \mathbf{R}^{n\times k}$)

$$y = F\xi$$

where $\xi \in \mathbf{R}^k$ are independent random variables ($k \le p$), with zero mean and identity covariance matrix. Here, $F \in \mathbf{R}^{n\times k}$ is the *loading* matrix.

Then the covariance matrix of y is

$$C = \mathbf{E}(F\xi\xi^T F^T) = FF^T.$$

In effect we are postulating that a few factors drive the market observations. This is the *same* as the PCA seen before!

Factor models

More generally we can assume that market observations are of the form

$$y = F\xi + \sigma e,$$

with (ξ, e) a zero-mean, random variable with identity covariance matrix, and $\sigma > 0$ is a parameter.

- ξ contains the market factors;
- e is a noise term that affect each observation independently ("idiosyncratic noise").

The covariance matrix of y is of the form

$$S = FF^T + D^2$$

with $D^2 = \sigma^2 I$. We can fit this model (*i.e.*, find F, σ) via SVD.

More general factor models allow for idiosyncratic noises with different variances (see lecture 3). However SVD cannot be used directly and the fitting problem is more challenging.

Outline

Motivation

Linear Algebra Recap
- Eigenvalues
- Singular values

Principal Component Analysis
- Overview
- Deflation: iterated variance maximization
- Example

Low-rank approximations
- Problem
- Link with PCA
- Explained variance and approximation error
- SVD-based Auto-Encoder
- Factor models

Extensions
- Robust PCA
- Sparse PCA

Robust PCA

PCA is based on the assumption that the data matrix can be (approximately) written as a low-rank matrix:

$$X = LR^T,$$

with $L \in \mathbf{R}^{n \times k}$, $R \in \mathbf{R}^{m \times k}$, with $k << m, n$.

Robust PCA [2] assumes that A has a “low-rank plus sparse” structure:

$$X = N + LR^T$$

where “noise” matrix N is sparse (has many zero entries).

How do we discover N, L, R based on A?

Robust PCA model

In robust PCA, we solve the convex problem

$$\min_{N} \|X - N\|_* + \lambda \|N\|_1$$

where $\|\cdot\|_*$ is the so-called nuclear norm (sum of singular values) of its matrix argument. At optimum, $X - N$ has usually low-rank.

Motivation: the nuclear norm is akin to the l_1-norm of the vector of singular values, and l_1-norm minimization encourages sparsity of its argument.

CVX syntax

Here is a matlab snippet that solves a robust PCA problem via CVX, given integers n, m, a $n \times m$ matrix X and non-negative scalar λ exist in the workspace:

```
cvx_begin
variable Xhat(n,m);
minimize( norm_nuc(X-Xhat)+ lambda*norm(Xhat(:),1))
cvx_end
```

- Note the use of `norm_nuc`, which stands for the nuclear norm.
- In practice, this CVX code does not run on large matrices, for memory limitations.
- Efficient specialized algorithms exist [2].

Alternatively, we can use a power iteration-like algorithm [6], alternating over L, R

$$\min_{L,R} \|X - LR^T\|_1 \;:\; L \in \mathbf{R}^{m\times k}, \;\; R \in \mathbf{R}^{n\times k},$$

Each step is a convex problem.

Motivation

One of the issues with PCA is that it does not yield principal directions that are easily interpretable:

- The principal directions are really combinations of all the relevant features (say, assets).
- Hence we cannot interpret them easily.
- The previous thresholding approach (select features with large components, zero out the others) can lead to much degraded explained variance.

Sparse PCA

Problem definition

Modify the variance maximization problem:

$$\max_x \; x^T S x - \lambda \, \mathbf{Card}(x) \;:\; \|x\|_2 = 1,$$

where penalty parameter $\lambda \geq 0$ is given, and $\mathbf{Card}(x)$ is the cardinality (number of non-zero elements) in x.

The problem is hard but can be approximated via convex relaxation.

Safe feature elimination

Express S as $S = R^T R$, with $R = [r_1, \ldots, r_p]$ (each r_i corresponds to one feature, *e.g.* asset).

Theorem (Safe feature elimination)

We have

$$\max_{x \,:\, \|x\|_2=1} \; x^T S x - \lambda\, \mathbf{Card}(x) = \max_{z \,:\, \|z\|_2=1} \sum_{i=1}^{p} \max(0, (r_i^T z)^2 - \lambda).$$

- Reduces to ordinary formula when $\lambda = 0$.
- When $\lambda > 0$ problem is hard, not amenable to SVD methods.

SAFE

Corollary

If $\lambda > \|r_i\|_2^2 = S_{ii}$, we can safely remove the i-th feature (row/column of S).

Proof.

If $\|z\|_2 = 1$, then $|r_i^T z| \leq \|r_i\|_2$. □

- The presence of the penalty parameter allows to prune out dimensions in the problem.
- Criterion simply based on variance of each feature (*i.e.*, directional variance along unit vectors).
- In practice, we want λ high as to allow better interpretability.
- Hence, interpretability requirement makes the problem easier in some sense!

Sparse PCA Algorithms

- The Sparse PCA problem remains challenging due to the huge number of variables.
- SAFE technique does allow big reduction in problem size.
- Still area of active research. (Like SVD in the 70's-90's. . .)

Sparse PCA

Thresholded power iteration

Efficient heuristic to solve, for given $n \times m$ matrix M:

$$\min_{p,q} \|X - pq^T\|_F \ : \ \mathbf{Card}(p) \leq k, \ \ \mathbf{Card}(q) \leq h.$$

Initialize p, q to be random and

$$p \rightarrow P(T_k(Xq)), \ \ q \rightarrow P(T_h(X^T p)),$$

where

- X is the (centered) data matrix,
- P is the l_2 normalization operator (for $z \neq 0$, $P(z) = z/\|z\|_2$),
- operator T_k removes all but the k largest-magnitude components of its input.
- Reduces to a standard method for PCA (power iteration) when $k = n$, $h = m$.

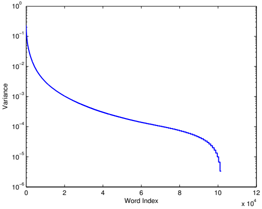
For sparse data, can be used in very high dimensions.

Example

Sparse PCA of New York Times headlines

Data: NYTtimes text collection contains $300,000$ articles and has a dictionary of $102,660$ unique words.

The variance of the features (words) decreases very fast:



Sorted variances of 102,660 words in NYTimes data.

With a target number of words less than 10, SAFE allows to reduce the number of features from $n \approx 100,000$ to $n = 500$.

Example

Sparse PCA of New York Times headlines

Words associated with the top 5 sparse principal components in NYTimes

1st PC	2nd PC	3rd PC	4th PC	5th PC
(6 words)	(5 words)	(5 words)	(4 words)	(4 words)
million	point	official	president	school
percent	play	government	campaign	program
business	team	united_states	bush	children
company	season	u_s	administration	student
market	game	attack		
companies				

Note: the algorithm found those terms without any information on the subject headings of the corresponding articles (unsupervised problem).

NYT Dataset

Comparison with thresholded PCA

Thresholded PCA involves simply thresholding the principal components.

$k = 2$	$k = 3$	$k = 9$	$k = 14$
even	even	even	would
like	like	we	new
	states	like	even
		now	we
		this	like
		will	now
		united	this
		states	will
		if	united
			states
			world
			so
			some
			if

1st PC from Thresholded PCA for various cardinality k. The results contain a lot of non-informative words.

References

G. Calafiore and L. El Ghaoui.
Optimization Models.
Cambridge University Press, 2014.

E.J. Candes, X. Li, Y. Ma, and J. Wright.
Robust principal component analysis.
Arxiv preprint ArXiv:0912.3599, 2009.

L. El Ghaoui.
Livebook: Optimization models and applications, 2016.
(Register to the livebook web site).

G.H. Golub and C.F. Van Loan.
Matrix computations, volume 3.
Johns Hopkins Univ Pr, 1996.

G. Strang.
Introduction to linear algebra.
Wellesley Cambridge Pr, 2003.

Madeleine Udell, Corinne Horn, Reza Zadeh, Stephen Boyd, et al.
Generalized low rank models.
Foundations and Trends® in Machine Learning, 9(1):1–118, 2016.