Statistical Learning with Sparsity

Matrix related topics and sparse multivariate methods

Boen Jiang

Fudan University

May 4, 2025

Matrix completion

- Given data in the form of an $m \times n$ matrix $\mathbf{Z} = \{z_{ij}\}$, we look for a matrix $\hat{\mathbf{Z}}$ that approximates $\mathbf{Z}$.
- We want to gain understanding of the matrix $\mathbf{Z}$ through an approximation $\hat{\mathbf{Z}}$ that has a simple structure.
- The general approach is to consider estimators based on optimisation problems of the form

$$\hat{\mathbf{Z}} = \arg \min_{\mathbf{M} \in \mathbb{R}^{m \times n}} \|\mathbf{Z} - \mathbf{M}\|_F^2 \quad \text{subject to} \quad \Phi(\mathbf{M}) \leq c$$

  where $\|\cdot\|_F^2$ is the Frobenius norm and $\Phi(\cdot)$ is a constraint function that encourages $\hat{\mathbf{Z}}$ to be sparse in some general sense.
- When $\mathbf{Z}$ has missing entries we would like to impute or fill the missing entries in $\mathbf{Z}$. This problem is known as matrix completion.

Matrix rank

Rank-1 matrix

An equivalent definition of a rank- $1 m \times n$ matrix is as the outer product $\mathbf{u}\mathbf{v}^1$ of an m-vector $\mathbf{u} \neq 0$ and an n-vector $\mathbf{v} \neq 0$

$$\mathbf{A} = \mathbf{u}\mathbf{v}^\top = \begin{bmatrix} -- & u_1\mathbf{v}^\top & -- \\ -- & u_2\mathbf{v}^\top & -- \\ & \vdots & \\ -- & u_m\mathbf{v}^\top & -- \end{bmatrix} = \begin{bmatrix} | & | & & | \\ v_1\mathbf{u} & v_2\mathbf{u} & \cdots & v_n\mathbf{u} \\ | & | & & | \end{bmatrix}$$

Note that each row is a multiple of $\mathbf{v}^\top$, and each column is multiple of $\mathbf{u}$.

Singular Value Decomposition

- Let $\mathbf{Z}$ be a $m \times n$ matrix with $m \geq n$.
- Its singular value decomposition takes the form $\quad \mathbf{Z} = UDV^\top$

$$([u_1] \dots [u_n]) \begin{pmatrix} d_1 & & \\ & \ddots & \\ & & d_n \end{pmatrix} \begin{pmatrix} [& v_1 &] \\ & \vdots & \\ [& v_n &] \end{pmatrix}$$

- $\mathbf{U}$ is an $m \times n$ orthogonal matrix $\left(\mathbf{U}^\top \mathbf{U} = \mathbf{I}_n\right)$ whose columns $\mathbf{u}_j \in \mathbb{R}^m$ are called left singular vectors.
- $\mathbf{D}$ is an $n \times n$ diagonal matrix with diagonal elements $d_1 \geq d_2 \geq \cdots \geq d_n \geq 0$ known as the singular values. $\mathrm{rank}(Z) = \sharp\{i : d_i > 0\}$.
- $\mathbf{V}$ is an $n \times n$ orthogonal matrix $\left(\mathbf{V}^T \mathbf{V} = \mathbf{I}_n\right)$ whose columns $\mathbf{v}_j \in \mathbb{R}^m$ are called right singular vectors.
- Another formulation :U, V are m and n unitary (orthogonal) matrices, respectively, and $D_{m \times n}$ bind $D_{n \times n}$ and 0s.

Matrix rank

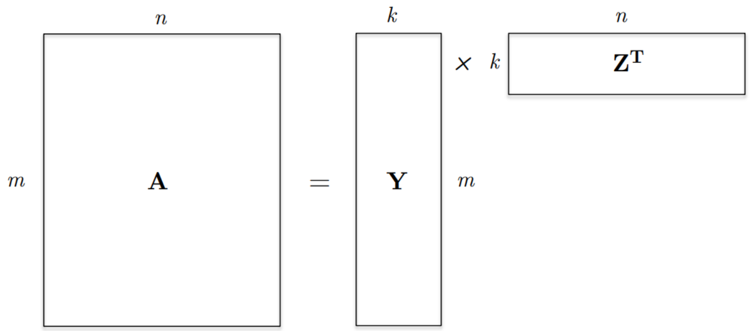


Figure 1: Any matrix $\mathbf{A}$ of rank k can be decomposed into a long and skinny matrix times a short and long one.

Rank-r SVD

- The rank-r SVD is a decomposition based on the optimization problem

$$\underset{\mathrm{rank}(\mathbf{M})=r}{\mathrm{minimize}} \|\mathbf{Z} - \mathbf{M}\|_F$$

  where we have $\mathbf{Z} = \mathbf{U}\mathbf{D}\mathbf{V}^T$ and assume that $r \leq \mathrm{rank}(\mathbf{Z})$.
- It has a closed form solution $\hat{\mathbf{Z}}_r = \mathbf{U}\mathbf{D}_r\mathbf{V}^T$, where $\mathbf{D}_r$ is a diagonal matrix $\mathbf{D}$ with all but the first r diagonal entries set to zero.
- The solution is sparse in the sense that has all but r singular values that are zero.

Updating Huge ML Models

- One quite modern application of low-rank matrix approximations is for ”fine-tuning” huge models. In the setting of large language models (LLMs), one often has some off-the-shelf huge model, with billions (or more) parameters.
- Given this large model that has been trained on an enormous but generic corpus (text from the web), one often performs “fine-tuning”.
  - training on a domain-specific dataset
  - forum question and answers, medical reports, etc.
  - it is computationally extremely expensive to update such huge models on edge devices.

Low-Rank Adaptation (LoRA)

- Full parameter fine-tuning is expensive and slow, and often not necessary.
    - 7B-model means 7 billion parameters,
    - all weights get updated repeatedly for multiple “epochs”
    - storing and updating weights requires a lot of memory, which limits fine-tuning to large GPUs
- The 2021 paper *LORA: Low-Rank Adaptation of Large Language Models* considers a generalization of full fine-tuning
    1. Do we need fine-tune all the parameters? $\rightsquigarrow$ fine-tuning updates are generally close to low-rank
    2. How expressive should the matrix updates be? $\rightsquigarrow$ one can explicitly learns these updates to the original model in their factorized form, essentially training a model with 1000x or 10,000x fewer parameters
- Many downstrean tasks are intrinsically low-rank. (Aghajanyan et al., 2020)

Rank minimization

Candes and Recht:

$$\begin{array}{ll} \underset{\boldsymbol{M}}{\text{minimize}} & \text{rank}(\boldsymbol{M}) \\ \text{subject to} & Z_{ij} = M_{ij}, \quad (i,j) \in \Omega \end{array}$$

- $\Omega = \{(i,j) \mid Z_{ij}$ is available/observed$\}$
- Restriction can be relaxed to $\sum_{(i,j)\in\Omega} (z_{ij} - m_{ij})^2 \leq \delta$
- Finds the matrix with the minimum rank
- NP-hard and non-convex

Example

$$Z = \begin{pmatrix} 1 & ? & 3 \\ ? & 2 & ? \\ 4 & ? & ? \end{pmatrix}, \quad \Omega = \{(1,1), (1,3), (2,2), (3,1)\}$$

Nuclear norm minimization

The nuclear norm minimization problem is defined as

$$\begin{aligned} \underset{\boldsymbol{M}}{\text{minimize}} \quad & \|\boldsymbol{M}\|_* \\ \text{subject to} \quad & Z_{ij} = M_{ij}, \quad (i,j) \in \Omega \end{aligned}$$

Comparison between Nuclear Norm and Frobenius Norm:

- $\|\boldsymbol{X}\|_* = \operatorname{tr}\left(\sqrt{\boldsymbol{X}^T\boldsymbol{X}}\right) = \sum_{i=1}^{n} \sigma_i(\boldsymbol{X})$
- $\|\boldsymbol{X}\|_F = \sqrt{\operatorname{tr}\left(\boldsymbol{X}^T\boldsymbol{X}\right)} = \sqrt{\sum_{i=1}^{n} \sigma_i^2(\boldsymbol{X})}$

A relaxed version in Lagrange form:

$$\underset{\mathbf{M}\in\mathbb{R}^{m\times n}}{\text{minimize}} \left\{ \frac{1}{2} \sum_{(i,j)\in\Omega} (z_{ij} - m_{ij})^2 + \lambda \|\mathbf{M}\|_* \right\}$$

Soft impute / spectual regularization

- Given an observed subset Ω of matrix entries, we define the projection operator $\mathcal{P}_\Omega(\mathbf{Z})_{i,j} = \begin{cases} z_{ij} & \text{if } (i,j) \in \Omega \\ 0 & \text{if } (i,j) \notin \Omega \end{cases}$
- Rewrite the previous problem as

$$\text{minimize}_{\mathbf{M} \in \mathbb{R}^{m\times n}} \left\{ \frac{1}{2} \|\mathcal{P}_\Omega(\mathbf{Z}) - \mathcal{P}_\Omega(\mathbf{M})\|_F^2 + \lambda \|\mathbf{M}\|_* \right\}$$

- Given the singular value decomposition $\mathbf{Z} = \mathbf{UDV}^T$ of a rank- r matrix $\mathbf{Z}$, we define its soft-tresholded version as

$$S_\lambda(\mathbf{Z}) \equiv \mathbf{UD}_\lambda \mathbf{V}^T \quad \text{where} \quad \mathbf{D}_\lambda = \operatorname{diag}\left[(d_1 - \lambda)_+, \ldots, (d_r - \lambda)_+\right]$$

Theorem 2.1(Cai-Candes-Shen 2008)

For each $\lambda \geq 0$ and $\boldsymbol{Z} \in \mathbb{R}^{m\times n}$, the singular value shrinkage operator obeys

$$S_\lambda(\mathbf{Z}) = \arg\min_{\boldsymbol{M}} \left\{ \frac{1}{2} \|\boldsymbol{M} - \boldsymbol{Z}\|_F^2 + \lambda \|\boldsymbol{M}\|_* \right\}$$

Singular Value Thresholding algorithm

Then the optimization probblem can be rewrite as

$$\min_{\mathbf{M}\in\mathbb{R}^{m\times n}} \left\{ \frac{1}{2} \|\mathbf{M} - [\mathcal{P}_\Omega(\mathbf{Z}) + \mathbf{M} - \mathcal{P}_\Omega(\mathbf{M})]\|_F^2 + \lambda\|\mathbf{M}\|_* \right\},$$

then we have

算法 Soft-Impute for Matrix Completion

```
1: Initialize $\mathbf{Z}^{\text{old}} = \mathbf{0}$ and create a decreasing grid $\lambda_1 > \ldots > \lambda_K$.
2: for $k = 1$ to $K$ do
3:     Set $\lambda = \lambda_k$.
4:     repeat
5:         Compute $\widehat{\mathbf{Z}}_\lambda \leftarrow \mathcal{S}_\lambda\left(P_\Omega(\mathbf{Z}) + P_\Omega^\perp\left(\mathbf{Z}^{\text{old}}\right)\right)$.
6:         Update $\mathbf{Z}^{\text{old}} \leftarrow \widehat{\mathbf{Z}}_\lambda$.
7:     until convergence
8: end for
9: Output the sequence of solutions $\widehat{\mathbf{Z}}_{\lambda_1}, \ldots, \widehat{\mathbf{Z}}_{\lambda_K}$.
```

Remarks on soft impute for matrix completion

- Computational tricks:

$$\mathcal{P}_{\Omega}(\mathbf{Z}) + \mathcal{P}_{\Omega}^{\perp}\left(\mathbf{Z}^{\text{old}}\right) = \underbrace{\mathcal{P}_{\Omega}(\mathbf{Z}) - \mathcal{P}_{\Omega}\left(\mathbf{Z}^{\text{old}}\right)}_{\text{sparse}} + \underbrace{\mathbf{Z}^{\text{old}}}_{\text{low rank}} .$$

- Convergence to global minimum.
- Convergence speed: The algorithm converges at least sub-linearly, meaning that $\mathcal{O}(1/\delta)$ iterations are sufficient to compute a solution that is δ-close to the global optimum.
- This has been implemented in the R package `softImpute`.

Toy simulation

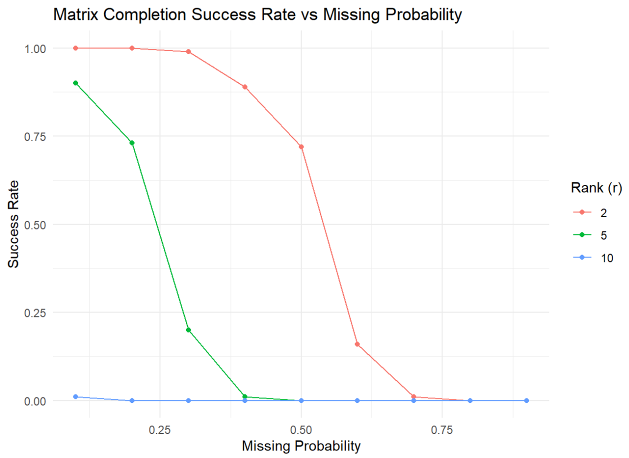


Figure: Imputation of missing values in a 20×20 matrix. $Z = UV^\top$ with rank to be 2, 5, 10. A success means $\left\|\mathcal{P}_\Omega^\perp(\mathbf{Z} - \widehat{\mathbf{Z}})\right\|_2^2 / \left\|\mathcal{P}_\Omega^\perp(\mathbf{Z})\right\|_2^2 < 10^{-2}$, and replicate for 100 times to obtain the frequency. The tuning parameter was set to be 0.001.

Remarks on soft impute for matrix completion

Under exact settings,

- we see that when the rank is a small fraction of the matrix dimension, one can reproduce the missing entries with fairly high probability.
- But this gets significantly more difficult when the true rank is higher.
- As for the effect of the rank, note that we need roughly $\mathcal{O}(rp)$ parameters to specify an arbitrary $p \times p$ matrix with rank r, since it has $\mathcal{O}(r)$ singular vectors, each with p components.

Maximum margin matrix factorization (MMMF)

- Another class of techniques used in collaborative filtering problems are Maximum Margin Matrix Factorization (MMMF) methods. They use a factor model for approximating the matrix $\mathbf{Z}$.
- Let $\mathbf{M}_{m\times n} = \mathbf{A}\mathbf{B}^T$ where $\mathbf{A}$ and $\mathbf{B}$ are $m \times r$ and $n \times r$ respectively.
- Consider the optimization problem

$$\underset{\substack{\mathbf{A}\in\mathbb{R}^{m\times r}\\ \mathbf{B}\in\mathbb{R}^{n\times r}}}{\text{minimize}} \left\{ \left\| \mathcal{P}_\Omega(\mathbf{Z}) - \mathcal{P}_\Omega\left(\mathbf{A}\mathbf{B}^T\right) \right\|_F^2 + \lambda \left(\|\mathbf{A}\|_F^2 + \|\mathbf{B}\|_F^2 \right) \right\}.$$

- This problem turns out to be equivalent to the nuclear norm regularized problem

$$\underset{\mathbf{M}\in\mathbb{R}^{m\times n}}{\text{minimize}} \frac{1}{2} \left\| \mathcal{P}_\Omega(\mathbf{Z}) - \mathcal{P}_\Omega(\mathbf{M}) \right\|_F^2 + \lambda \|\mathbf{M}\|_\star$$

since

$$\|\mathbf{M}\|_\star = \min_{\substack{\mathbf{A}\in\mathbb{R}^{m\times r}, \mathbf{B}\in\mathbb{R}^{n\times r}\\ \mathbf{M}=\mathbf{A}\mathbf{B}^T}} \frac{1}{2} \left(\|\mathbf{A}\|_F^2 + \|\mathbf{B}\|_F^2 \right)$$

for any matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$.

Connections between MMMF and soft impute

Theorem (equivalence)

Let $\mathbf{Z}$ be an $m \times n$ matrix with observed entries indexed by Ω.

a. Let $r = \min(m, n)$. Then the solutions to MMMF and SI coincide for all $\lambda \geq 0$.
b. For some fixed $\lambda^* > 0$ suppose that SI has an optimal solution with rank r^*. Then for any optimal soution ($\hat{\mathbf{A}}, \hat{\mathbf{B}}$), to the MMMF with $r \geq r^*$ and $\lambda = \lambda^*$, the matrix $\hat{\mathbf{M}} = \hat{\mathbf{A}}\hat{\mathbf{B}}^T$ is an optimal solution for SI.

- This implies that the solution space of SI is contained in that of MMMF.
- The MMMF criterion defines a two-dimensional family of models indexed by the pair (r, λ), while the Soft-Impute criterion defines a one-dimensional family.
- This family is a special path in the two-dimensional grid of solutions $\left(\widehat{\mathbf{A}}_{(r,\lambda)}, \widehat{\mathbf{B}}_{(r,\lambda)}\right)$.

Remarks

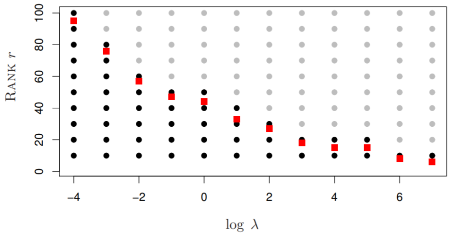


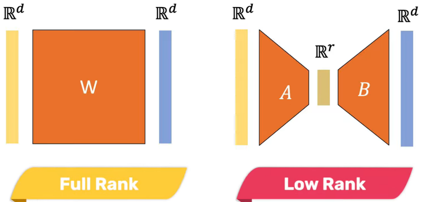
Figure: Source: SLS, Fig 7.7

- Any MMMF model at parameter combinations above the red points are redundant, since their fit is the same at the red point.
- the formulation SI is preferable for two reasons:
  1. it is convex
  2. it does both rank reduction and regularization at the same time.
- Using MMMF we need to choose the rank of the approximation and the regularization parameter λ.

LoRA

- For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d \times k}$, we constrain its update by representing the latter with a low-rank decomposition $W_0 + \Delta W = W_0 + BA$, where $B \in \mathbb{R}^{d \times r}$, $A \in \mathbb{R}^{r \times k}$, and the rank $r \ll \min(d, k)$.
- During training, W_0 is frozen and does not receive gradient updates, while A and B contain trainable parameters. Note both W_0 and $\Delta W = BA$ are multiplied with the same input, and their respective output vectors are summed coordinate-wise. For $h = W_0 x$, the modified forward pass yields:

$$h = W_0 x + \Delta W x = W_0 x + BAx$$



Multivariate multiple linear regression

- We have vector-valued responses $y_i \in \mathbb{R}^K$ and covariates $x_i \in \mathbb{R}^p$, and we wish to build a series of K linear regression models. With N observations on $(\ y_i, x_i\ )$, we can write these regression models in matrix form as

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\Theta} + \mathbf{E}$$

  with $\mathbf{Y} \in \mathbb{R}^{N\times K}, \mathbf{X} \in \mathbb{R}^{N\times p}, \boldsymbol{\Theta} \in \mathbb{R}^{p\times K}$ a matrix of coefficients, and $\mathbf{E} \in \mathbb{R}^{N\times K}$ a matrix of errors.
- Example 4.2 multitask learning
- Decomposition Θ gives

$$\mathbf{Y} = \mathbf{X}\mathbf{A}\mathbf{B}^T + \mathbf{E}$$

  with $\mathbf{A} \in \mathbb{R}^{p\times r}$ and $\mathbf{B} \in \mathbb{R}^{K\times r}$. One can think of having $r < K$ derived features $\mathbf{Z} = \mathbf{X}\widehat{\mathbf{A}}$ which are then distributed among the responses via K separate regressions $\widehat{\mathbf{Y}} = \mathbf{Z}\widehat{\mathbf{B}}^T$.

Reduced rank regression (RRR)

- The usual ordinary least squares (OLS) regression can be formulated as minimizing the following cost function: $L = \|\mathbf{Y} - \mathbf{X}\ \|^2$. Its solution is given by $\hat{\Theta}_{OLS} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{Y}$ and it is equivalent to doing K separate OLS regressions, one for each dependent variable.
- Reduced-rank regression introduces a rank constraint on Θ, namely L should be minimized with $\mathrm{rank}(\Theta) \leq r$, where r is the maximal allowed rank of Θ.
- Notice that the reduced rank objective can be alternatively written as

$$\min_{B} \mathrm{tr}\left[(Y - XB)(Y - XB)^\top\right] \text{ s.t. } \mathrm{rank}(B) \leq r.$$

- Recall that the rank condition makes this equivalent to minimizing

$$\min_{A,C} \mathrm{tr}\left[(Y - XAC)(Y - XAC)^\top\right]$$

where A is a $p \times r$ matrix and C is an $r \times q$ matrix.

Reduced rank regression (RRR)

- Let $\hat{B}_{\mathrm{OLS}} = \left(X^\top X\right)^{-1} X^\top Y$ be the OLS coefficient solution and $\hat{Y}_{\text{OLS}} = X\hat{B}_{\text{OLS}}$ be the fitted values. Also, let $V^{(r)} = (v_1, v_2, \ldots, v_r)$ be a matrix whose columns are the first r eigenvectors of $\hat{Y}_{\mathrm{OLS}}^\top \hat{Y}_{\mathrm{OLS}}$.
- Then the minimum of the above problem is achieved with

$$C = V^{(r)}$$

and

$$A = \left(X^\top X\right)^{-1} X^\top Y V^{(r)\top}$$

- As before, the nuclear norm is a useful convex penalty for enforcing lowrank structure on an estimate. In this case we would solve the optimization problem

$$\underset{\mathbf{\Theta}\in\mathbb{R}^{p\times K}}{\text{minimize}} \left\{\|\mathbf{Y} - \mathbf{X}\mathbf{\Theta}\|_F^2 + \lambda\|\mathbf{\Theta}\|_\star\right\}$$

Penalized matrix decomposition

- Maximum-margin matrix factorization methods lead to other forms of regularization.
- Consider the ℓ_1-penalized version

$$\underset{\mathbf{U}\in\mathbb{R}^{m\times r},\mathbf{V}\in\mathbb{R}^{m\times r},\mathbf{D}\in\mathbb{R}^{r\times r}}{\text{minimize}} \left\{ \left\| \mathbf{Z} - \mathbf{U}\mathbf{D}\mathbf{V}^{T} \right\|_{\mathbf{F}}^2 + \lambda_1 \|\mathbf{U}\|_1 + \lambda_2 \|\mathbf{V}\|_1 \right\}$$

  with $\mathbf{D}$ diagonal and non-negative and $\mathbf{U}\mathbf{D}\mathbf{V}^{\top}$ the SVD. We assume that all values of $\mathbf{Z}$ are observed.
- $\Rightarrow$ obtain sparse left and right singular vectors

Penalized matrix decomposition

- We want to optimize $\min_{\mathbf{U},\mathbf{V},\mathbf{D}} \left\{ \left\| \mathbf{Z} - \mathbf{U}\mathbf{D}\mathbf{V}^\top \right\|_F^2 + \lambda_1 \|\mathbf{U}\|_1 + \lambda_2 \|\mathbf{V}\|_1 \right\}$
- Start from 1d: We write it in the constrained form and we consider the one-dimensional case, i.e. $\mathrm{r} = 1$.

$$\min_{\mathbf{u}\in\mathbb{R}^m, \mathbf{v}\in\mathbb{R}^n, d\geq 0} \left\| \mathbf{Z} - d\mathbf{u}\mathbf{v}^T \right\|_F^2 \text{ s.t. } \|\mathbf{u}\|_1 \leq c_1 \text{ and } \|\mathbf{v}\|_1 \leq c_2$$

- We see that the estimator in the constrained form tends to produce solutions that are too sparse. For avoiding it we add a ℓ_2-norm constraints.

$$\min_{\mathbf{u}\in\mathbb{R}^m, \mathbf{v}\in\mathbb{R}^n, d\geq 0} \left\| \mathbf{Z} - d\mathbf{u}\mathbf{v}^T \right\|_F^2 \text{ s.t. } \begin{array}{r} \|\mathbf{u}\|_1 \leq c_1, \|\mathbf{v}\|_1 \leq c_2 \\ \|\mathbf{u}\|_2 \leq 1, \|\mathbf{v}\|_2 \leq 1 \end{array}$$

- the optimization problem is well defined as long as $1 \leq c_1 \leq \sqrt{m}$ and $1 \leq c_2 \leq \sqrt{n}$.

Penalized matrix decomposition

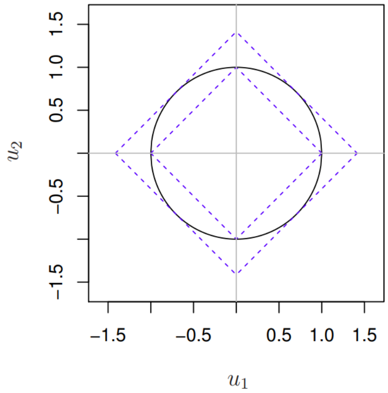


Figure: Source: SLS, fig 7.10

For both the ℓ_1 and ℓ_2 constraints to be active, the constraint radius *c* must be between 1 and $\sqrt{2}$. The constraint on $\|\mathbf{u}\|_1 = 1$ and $\|\mathbf{u}\|_1 = \sqrt{2}$ are shown using the dashed lines.

Penalized matrix decomposition

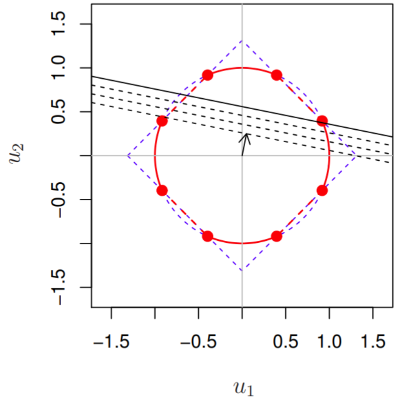


Figure: Source: SLS, fig 7.10

- If we fix the second component $\mathbf{v}$, the criterion is linear in $\mathbf{u}$.
- The figure shows that in the two dimensions, the points where both the ℓ_1 and the ℓ_2 constraints are active have neither u_1 nor u_2 equal to zero.

Penalized matrix decomposition

- The previous formula is biconvex and it can be minimized in an alternating fashion $\rightsquigarrow$ the solution is a soft-thresholding operation, where the soft-thresholding operator is applied element-wise on the vector.
- We fix $\mathbf{v} \in \mathbb{R}^n$ such that it satisfy the constraints. The update is $\mathbf{u} \leftarrow \dfrac{S_{\lambda_1}(\mathbf{Z}\mathbf{v})}{\|S_{\lambda_1}(\mathbf{Z}\mathbf{v})\|_2}$/ The update is $\mathbf{v} \leftarrow \dfrac{S_{\lambda_2}\left(\mathbf{Z}^\top \mathbf{u}\right)}{\|S_{\lambda_2}\left(\mathbf{Z}^\top \mathbf{u}\right)\|_2}$.
- The threshold λ_1 (resp. λ_2) must be chosen so that it satisfies the constraints. This means that $\lambda_1 = 0$ if $\|\mathbf{u}\|_1 \leq c_1$ and $\lambda_1 > 0$ if $\|\mathbf{u}\|_1 = c_1$.
- If the ℓ_1 constraints have no effect, then the algorithm reduces to the power method that finds the largest singular vector.

Penalized matrix decomposition

算法 Single-factor algorithm (Algorithm 7.2)

```
1: Initialize $\mathbf{v}$ as the top left singular vector from the SVD of $\mathbf{Z}$.
2: repeat
3:     Update $\mathbf{u} \leftarrow \frac{\mathcal{S}_{\lambda_1}(\mathbf{Zv})}{\|\mathcal{S}_{\lambda_1}(\mathbf{Zv})\|_2}$, where $\lambda_1$ is the smallest value such that $\|\mathbf{u}\|_1 \leq c_1$.
4:     Update $\mathbf{v} \leftarrow \frac{\mathcal{S}_{\lambda_2}(\mathbf{Z}^\top\mathbf{u})}{\|\mathcal{S}_{\lambda_2}(\mathbf{Z}^\top\mathbf{u})\|_2}$, where $\lambda_2$ is the smallest value such that $\|\mathbf{v}\|_1 \leq c_2$.
5: until convergence
6: Return $\mathbf{u}$, $\mathbf{v}$, and $d = \mathbf{u}^\top\mathbf{Zv}$.
```

Penalized matrix decomposition

算法 Multifactor Penalized Matrix Decomposition (Algorithm 7.3)

```
1: Let $\mathbf{R} \leftarrow \mathbf{Z}$.
2: for $k = 1$ to $K$ do
3:     Apply the single-factor algorithm to $\mathbf{R}$ to obtain $\mathbf{u}_k$, $\mathbf{v}_k$, and $d_k$.
4:     Update $\mathbf{R} \leftarrow \mathbf{R} - d_k \mathbf{u}_k \mathbf{v}_k^\top$.
5: end for
```

- The algorithm is a generalization of the single-factor algorithm.
- $\hat{\boldsymbol{Z}} = \sum_{k=1}^{K} d_k u_k v_k^\top$.
- If we assume that the ℓ_1 penalties don't exist, then the multifactor PMD algorithm leads to the rank-K SVD of $\mathbf{Z}$. Recall: the rank-e SVD $\hat{\mathbf{Z}}_r = \mathbf{U}\mathbf{D}_r\mathbf{V}^T$.

Remarks on PMD

In general, the optimization

$$\min_{\mathbf{u}\in\mathbb{R}^m,\mathbf{v}\in\mathbb{R}^n,d\geq 0} \left\|\mathbf{Z} - d\mathbf{u}\mathbf{v}^T\right\|_F^2 \text{ s.t. } \|\mathbf{u}\|_1 \leq c_1, \|\mathbf{v}\|_1 \leq c_2$$
$$\|\mathbf{u}\|_2 \leq 1, \|\mathbf{v}\|_2 \leq 1$$

can be used for also other type of penalties.
For example for the fused lasso penalty

$$\Phi(\mathbf{u}) = \sum_{j=2}^{m} |u_j - u_{j-1}|$$

where $\mathbf{u} = (u_1, u_2, \ldots u_m)$.

Additive matrix decomposition

We want to decompose a matrix into the sum of two or more matrices. The components in this addition must have complementary structure, for example we can decompose a matrix into the sum of a low rank matrix and a sparse matrix.
There are a lot of variety of applications for the additive matrix decomposition, for example:

- robust PCA
- robust matrix completion

Additive matrix decomposition

- We can describe most of these applications in terms of the noisy linear pbservation model, that is $\mathbf{Z} = \mathbf{L}^* + \mathbf{S}^* + \mathbf{W}$, where $\mathbf{W}$ is a noise matrix. Here we have the pair $(\mathbf{L}^*, \mathbf{S}^*)$ that specify the additive matrix decomposition into a low rank matrix $\mathbf{L}^*$ and a sparse components matrix $\mathbf{S}^*$.
- We want to find estimators of the pair $(\mathbf{L}^*, \mathbf{S}^*)$ based on

$$\underset{\mathbf{L},\mathbf{S}\in\mathbb{R}^{m\times n}}{\text{minimize}} \left\{ \frac{1}{2}\|\mathbf{Z} - (\mathbf{L} + \mathbf{S})\|_{\mathrm{F}}^2 + \lambda_1 \Phi_1(\mathbf{L}) + \lambda_2 \Phi_2(\mathbf{S}) \right\},$$

  where we have that Φ_1 and Φ_2 are penalty functions.
- Here we have a look to the situation with a low rank and sparse matrices with $\Phi_1(\mathbf{L}) = \|\mathbf{L}\|_\star$ and $\Phi_2(\mathbf{S}) = \|\mathbf{S}\|_1$.

PCA

Assume that the random r-vector $\mathbf{X} = (X_1, \ldots, X_r)^T$ has mean $\boldsymbol{\mu}_X$ and covariance matrix Σ_{XX}.

PCA seeks to replace the set of r (unordered and correlated) input variables, $X_1, X_2, \ldots, X_r$, by a (potentially smaller) set of t (ordered and uncorrelated) linear projections, $\xi_1, \ldots, \xi_t (t \leq r)$, of the input variables,

$$\xi_j = \mathbf{b}_j^T \mathbf{X} = b_{j1} X_1 + \cdots + b_{jr} X_r, \quad j = 1, \ldots, t$$

where we minimize the loss of information due to replacement.

In PCA, information is interpreted as the total variation of the original input variables,

$$\sum_{j=1}^{r} \mathrm{Var}\left(X_j\right) = \mathrm{tr}\left(\Sigma_{XX}\right)$$

PCA via variance-maximization

- The j th coefficient vector, $\mathbf{b}_j = (b_{j1}, \ldots, b_{jr})^T$, is chosen so that:
  - The first t linear projections $\xi_j, j = 1, \ldots, t$, of $\mathbf{X}$ are ranked in importance through their variances $\{\mathrm{Var}(\xi_j)\}$, which are listed in decreasing order of magnitude: $\mathrm{Var}(\xi_1) \geq \mathrm{Var}(\xi_2) \geq \cdots \geq \mathrm{Var}(\xi_t)$.
  - ξ_j is uncorrelated with all $\xi_k, k < j$.
- This leads to a Lagrangian formulation of the problem, where we want to maximize the variance of the projections $b^\top \Sigma b$ subject to the constraint that the coefficients $\mathbf{b}_j$ are normalized to have unit length $b^\top b = 1$ and orthogonal to the previous ones $b_k^\top \Sigma b = \lambda_k b_k^\top b = 0$.
- b_j are called loading vectors, the solution of b_j is v_j, the eigenvector of the covariance matrix Σ associated with the j-th largest eigenvalue λ_j.

The linear projections are then known as the first t principal components of X.

PCA via least-squares

Let $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_t)^T$ be a $t \times r$-matrix of weights $(t \leq r)$.
The linear projection can be written as a t-vector,

$$\boldsymbol{\xi} = \mathbf{B}\mathbf{X}.$$

- $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_t)^T$
- We want to find an r-vector $\boldsymbol{\mu}$ and an $r \times t$ matrix $\mathbf{A}$ such that the projections $\boldsymbol{\xi}$ have the property that $\mathbf{X} \approx \boldsymbol{\mu} + \mathbf{A}\boldsymbol{\xi}$ in some least-square sense.

We use the least-squares error criterion,

$$\mathrm{E}\left\{(\mathbf{X} - \boldsymbol{\mu} - \mathbf{A}\boldsymbol{\xi})^T(\mathbf{X} - \boldsymbol{\mu} - \mathbf{A}\boldsymbol{\xi})\right\}$$

as our measure of how well we can reconstruct $\mathbf{X}$ by the linear projection $\boldsymbol{\xi}$.

PCA via least-squares

- The goal is to choose $\mathbf{A}, \mathbf{B}$, and $\boldsymbol{\mu}$ to minimize

$$\mathrm{E}\left\{tr(\mathbf{X}-\boldsymbol{\mu}-\mathbf{ABX})^T(\mathbf{X}-\boldsymbol{\mu}-\mathbf{ABX})\right\}.$$

- The criterion can be minimized by the reduced-rank regression solution,

$$\mathbf{A}^{(t)}=(\mathbf{v}_1,\dots,\mathbf{v}_t)=\mathbf{B}^{(t)T},\quad \boldsymbol{\mu}^{(t)}=\left(\mathbf{I}_r-\mathbf{A}^{(t)}\mathbf{B}^{(t)}\right)\boldsymbol{\mu}_X$$

- $\mathbf{v}_j=\mathbf{v}_j\left(\boldsymbol{\Sigma}_{XX}\right)$ is the eigenvector associated with the j th largest eigenvalue, λ_j, of Σ_{XX}.
- Thus, our best rank- t approximation to the original $\mathbf{X}$ is given by

$$\widehat{\mathbf{X}}^{(t)}=\boldsymbol{\mu}^{(t)}+\mathbf{C}^{(t)}\mathbf{X}=\boldsymbol{\mu}_X+\mathbf{C}^{(t)}\left(\mathbf{X}-\boldsymbol{\mu}_X\right)$$

where $\mathbf{C}^{(t)}=\mathbf{A}^{(t)}\mathbf{B}^{(t)}=\sum_{j=1}^{t}\mathbf{v}_j\mathbf{v}_j^T$ is the reduced-rank regression coefficient matrix with rank t for the principal components case.

Sample PCA

- In practice, we estimate the principal components using n independent observations, $\{\mathbf{X}_i, i = 1, 2, \ldots, n\}$, on $\mathbf{X}$.
- We estimate μ_X by $\widehat{\boldsymbol{\mu}}_X = \overline{\mathbf{X}} = n^{-1} \sum_{i=1}^{n} \mathbf{X}_i$.
- We estimate Σ_{XX} by the sample covariance matrix, $\widehat{\boldsymbol{\Sigma}}_{XX} = n^{-1}\mathbf{S} = n^{-1}\mathcal{X}_c\mathcal{X}_c^\top$. The ordered eigenvalues of $\widehat{\Sigma}_{XX}$ are denoted by $\widehat{\lambda}_1 \geq \widehat{\lambda}_2 \geq \cdots \geq \widehat{\lambda}_r \geq 0$, and the eigenvector associated with the jth largest sample eigenvalue $\widehat{\lambda}_j$ is the j th sample eigenvector $\widehat{\mathbf{v}}_j, j = 1, \ldots, r$.
- The best rank- t reconstruction of $\mathbf{X}$ is given by

$$\widehat{\mathbf{X}}^{(t)} = \overline{\mathbf{X}} + \widehat{\mathbf{C}}^{(t)}(\mathbf{X} - \overline{\mathbf{X}})$$

where

$$\widehat{\mathbf{C}}^{(t)} = \widehat{\mathbf{A}}^{(t)}\widehat{\mathbf{B}}^{(t)} = \sum_{j=1}^{t} \widehat{\mathbf{v}}_j \widehat{\mathbf{v}}_j^T$$

Robust PCA

- We can't simply approximate the matrix $\mathbf{Z}$ with a low rank matrix $\mathbf{L}$, but we also have to add a sparse matrix $\mathbf{S}$ that model the corrupted variables.
- Given sparsity k and a target rank $\mathrm{r} \rightarrow$ solve the optimization problem

$$\underset{\mathrm{rank}(\mathrm{L}) \leq r, \mathrm{card}(\mathbf{S}) \leq k}{\text{minimize}} \frac{1}{2} \|\mathbf{Z} - (\mathbf{L} + \mathbf{S})\|_{\mathrm{F}}^2$$

  This problem with the rank and cardinality constraints is non-convex.
- We have a convex relaxation with the previous general optimization

$$\text{minimize}_{\mathbf{L},\mathbf{S} \in \mathbb{R}^{m \times n}} \left\{ \frac{1}{2} \|\mathbf{Z} - (\mathbf{L} + \mathbf{S})\|_{\mathrm{F}}^2 + \lambda_1 \Phi_1(\mathbf{L}) + \lambda_2 \Phi_2(\mathbf{S}) \right\}$$

  with the constraints $\phi_1(\mathbf{L}) = \|\mathbf{L}\|_*$ and $\phi_2(\mathbf{S}) = \sum_{i,j} |s_{ij}|$ for

  element-wise sparsity.

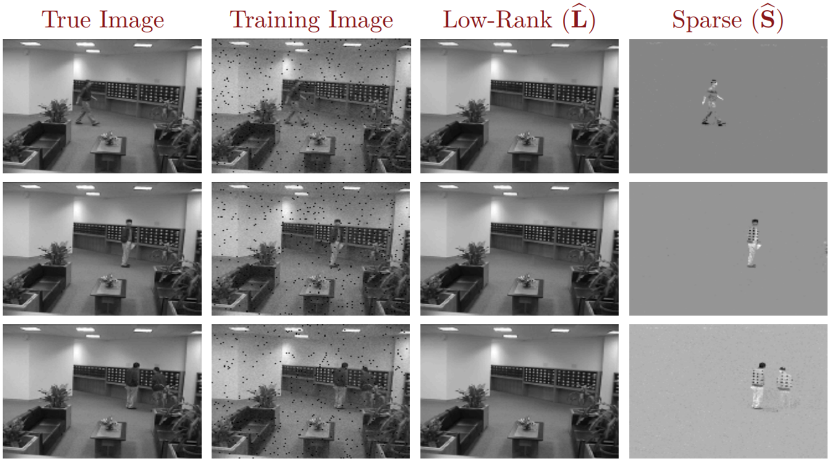

True Image
Training Image
Low-Rank ($\widehat{\mathbf{L}}$)
Sparse ($\widehat{\mathbf{S}}$)

Figure 7.11 *Video surveillance. Shown are the true image, noisy training image with missing-values, the estimated low-rank part, and the sparse part aligned side by side. The true images were sampled from the sequence and include ones with varying illumination and some benchmark test sequences. Despite the missingness and added noise the procedure succeeds in separating the moving components (people) from the fixed background.*

Robust matrix completion

- Sometimes also ratings may be corrupted.
  - a seller may pay users or bots to leave 5-star reviews for their product, inflating the rating artificially.
  - adversarial competition
- As in the robust PCA, we add a sparse component $\mathbf{S}$ to our low rank matrix $\mathbf{L}$.
- The nature of sparsity depends on how we model the adversarial behaviour:
  - small fraction of entries corrupted $\rightsquigarrow$ element-wise sparsity via the ℓ_1-norm;
  - rows (users) are corrupted $\rightsquigarrow$ row-wise sparsity penalty via the group lasso norm $\|\mathbf{S}\|_{1,2} = \sum_{i=1}^{m} \|\mathbf{S}_i\|_2$, where $\mathbf{S}_i \in \mathbb{R}^n$ is the i^{th} row of the matrix.

Robust matrix completion

- We now have a look to the optimization for a row-wise sparsity penalty. This is

$$\underset{\mathbf{L},\mathbf{S}\in\mathbb{R}^{m\times n}}{\text{minimize}}\left\{\frac{1}{2}\sum_{(i,j)\in\Omega}\left[z_{ij}-(\mathbf{L}_{ij}+\mathbf{S}_{ij})\right]^2+\lambda_1\|\mathbf{L}\|_\star+\lambda_2\sum_{i=1}^{m}\|\mathbf{S}_i\|_2\right\}$$

Quick sum up

- We have tried to find a matrix $\hat{\mathbf{Z}}$ that approximates $\mathbf{Z}$.
  - filling in missing data of $\mathbf{Z}$ $\rightsquigarrow$ completion;
  - approximating $\mathbf{Z}$ with a matrix that has a simpler structure $\rightsquigarrow$ decomposition.
- We have seen different methods of optimization, like: singular value decomposition, penalized matrix decomposition and some matrix completion methods.
- These optimization problems are really useful in machine learning and in recommender systems.

Problems of PCA

- As soon as $p \gg N$, PCA can run into problems.
- Sample covariance $\mathrm{Cov}(X)$ is not converging to the population covariance when N and p grow on the same scale (e.g. Marchenko-Pastur distribution). As a result the eigenvectors can be wrong as well.
- To fix this, we need to regularize and impose sparseness on the loading vectors $v_1, \ldots v_r$.
- Usually no way to interpret resulting axes/features of PCA. With sparseness this gets easier.

Sparse (sample) PCA: first optimization task

- Focus on first principle component.
- $$\underset{\|v\|_2=1}{\text{maximize}} \left\{ v^T \mathbf{X}^T \mathbf{X} v \right\} \text{ subject to } \|v\|_0 \leq t,$$

  where $\|v\|_0 = \sum_{j=1}^{p} \mathbb{I}\left[v_j \neq 0\right]$ simply counts the number of nonzeros in the vector v.
- Relax the objective to so called SCoTLASS procedure:

  $$\underset{v:\|v\|_2=1}{\operatorname{argmax}} v^\top X^\top X v \text{ subject to } \|v\|_1 \leq t$$
- Cons: both problems are non-convex. We can't implement simple iterative methods.

Sparse PCA via Penalized Matrix Criterion

- Use that PCA is closely related to SVD and apply the penalized matrix criterion:

$$\underset{\|u\|_2=\|v\|_2=1}{\operatorname{argmax}} \; u^\top X v \text{ subject to } \|v\|_1 \leq t$$

- For fixed v , the optimal u is given by $\mathrm{u} = \frac{Xv}{\|Xv\|_2}$
- This cost function is biconvex in (u, v) and every solution $\hat{v}$ to this problem is a solution to the SCoTCLASS problem.
- Use alternating minimization

Alternating Algorithm for Rank-One Sparse PCA

算法 Alternating Algorithm for Rank-One Sparse PCA

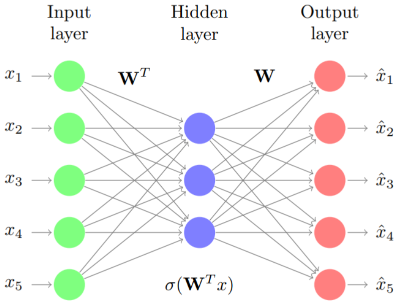
```
1: Initialize $v \in \mathbb{R}^p$ with $\|v\|_2 = 1$.
2: repeat
3:     Update $\mathbf{u} \leftarrow \frac{\mathbf{X}v}{\|\mathbf{X}v\|_2}$.
4:     Compute $\lambda$:
         • If $\|\mathbf{X}^T\mathbf{u}\|_1 \le t$, set $\lambda = 0$.
         • Otherwise, choose $\lambda > 0$ such that $\|v(\lambda, \mathbf{u})\|_1 = t$.
5:     Update $v \leftarrow v(\lambda, \mathbf{u}) = \frac{\mathcal{S}_\lambda(\mathbf{X}^T\mathbf{u})}{\|\mathcal{S}_\lambda(\mathbf{X}^T\mathbf{u})\|_2}$.
6: until convergence
```

Extending to higher rank

- We saw that in the non-sparse case, a simple iterative method was also solving the high rank problem.
- We can adapt the penalized matrix criterion in the following way:
  - Calculate the first solution (u_1, v_1, d_1) with $d_1 = u_1^\top X v_1$
  - Subtract the solution: $X' = X - d_1 u_1 v_1^\top$
  - Calculate the next solution (u_2, v_2, d_2) using X'
  - Iterate the procedure r times.
- Does not solve the multirank problem as in PCA case.
- There exists other methods derived from the reconstruction view.

How can we extend PCA?

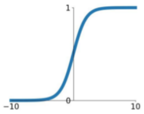
- We saw that PCA produced new features by linear combinating the old ones.
- Extend this by also allowing for non-linear relationships.
- One could also consider more than one consecutive transformation of the features, leading to more “layers”.
- All these extensions are used in the autoencoder.



Nonlinearity: activation function

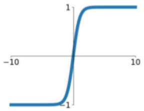
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



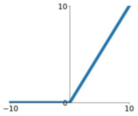
tanh

$\tanh(x)$



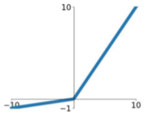
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

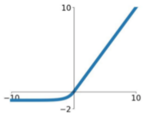


Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Architecture of an Autoencoder

- Input: One observation $x \in \mathbb{R}^p$
- Encoder: Linear activation $W \in \mathbb{R}^{p \times r}$ followed by a non-linear activation function σ :

$$h = \sigma\left(W^\top x\right)$$

- Decoder: Same linear activation $W \in \mathbb{R}^{r \times p}$:

$$\hat{x} = Wh$$

- In total: If we denote the autoencoder by f, we can compactly write:

$$\hat{x} = f(x) = W\sigma\left(W^\top x\right)$$

Remark

Taking σ as the identity gives PCA.

Optimization for autoencoders

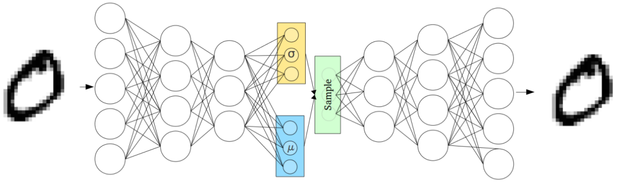
- Goal: Reconstruct input as well as possible by choosing good weights W.
- Optimize the quadratic loss:

$$\min_{W\in\mathbb{R}^{r\times p}} \frac{1}{2}\sum_{i=1}^{N} \|x_i - \hat{x}_i\|_2^2 = \min_{W\in\mathbb{R}^{r}\times p} \frac{1}{2}\sum_{i=1}^{N} \left\|x_i - W\sigma\left(W^T x_i\right)\right\|_2^2$$

- This problem is not convex but is usually solved by iterative methods such as stochastic gradient descent.

Architechture of Variational Autoencoder

- Idea: Let the encoder produce two vectors, a mean vector μ and a variance vector σ^2.
- Create a new random vector *h* by sampling from a normal distribution with mean μ and variance σ^2.
- This *h* is then passed to the decoder which again tries to produce the original input.



Canonical correlation analysis (CCA)

- Canonical variate and correlation analysis (CVA or CCA) is a method for studying linear relationships between two vector variates, which we denote by $\mathbf{X} = (X_1, \dots, X_r)^T$ and $\mathbf{Y} = (Y_1, \dots, Y_s)^T$.
- CCA seeks to replace the two sets of correlated variables, $\mathbf{X}$ and $\mathbf{Y}$, by t pairs of new variables,

$$(\xi_i, \omega_i), \quad i = 1, 2, \dots, t, \quad t \leq \min(r, s)$$

where

$$\begin{cases} \xi_j = \mathbf{g}_j^T \mathbf{X} = g_{1j}X_1 + g_{2j}X_2 + \cdots + g_{rj}X_r \\ \omega_j = \mathbf{h}_j^T \mathbf{Y} = h_{1j}Y_1 + h_{2j}Y_2 + \cdots + h_{sj}Y_s \end{cases}$$

are linear projections of $\mathbf{X}$ and $\mathbf{Y}$, respectively.

Population CCA

The j th pair of coefficient vectors, $\mathbf{g}_j = (g_{1j}, \dots, g_{rj})^T$ and $\mathbf{h}_j = (h_{1j}, \dots, h_{sj})^T$, are chosen so that

- the pairs $\{(\xi_j, \omega_j)\}$ are ranked in importance through their correlations,

$$\rho_j = \operatorname{corr}\{\xi_j, \omega_j\} = \mathbf{g}_j^T \boldsymbol{\Sigma}_{XY} \mathbf{h}_j$$

  which are listed in descending order of magnitude: $\rho_1 \geq \rho_2 \geq \cdots \geq \rho_t$, and $\left(\mathbf{g}_j^T \boldsymbol{\Sigma}_{XX} \mathbf{g}_j\right)^{1/2} = \left(\mathbf{h}_j^T \boldsymbol{\Sigma}_{YY} \mathbf{h}_j\right)^{1/2} = 1$
- ξ_j is uncorrelated with all previously derived ξ_k :

$$\operatorname{cov}\{\xi_j, \xi_k\} = \mathbf{g}_j^T \boldsymbol{\Sigma}_{XX} \mathbf{g}_k = 0, \quad k < j$$

- ω_j is uncorrelated with all previously derived ω_k :

$$\operatorname{cov}\{\omega_j, \omega_k\} = \mathbf{h}_j^T \boldsymbol{\Sigma}_{YY} \mathbf{h}_k = 0, \quad k < j$$

Sample CCA

- the sample covariance between $\mathbf{X}\beta$ and $\mathbf{Y}\theta$ is given by

$$\widehat{\mathrm{Cov}}(\mathbf{X}\beta, \mathbf{Y}\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(x_i^T\beta\right)\left(y_i^T\theta\right) = \frac{1}{N}\beta^T\mathbf{X}^T\mathbf{Y}\theta$$

  where x_i and y_i are the i^{th} rows of $\mathbf{X}$ and $\mathbf{Y}$, respectively.
- CCA solves the problem

$$\mathrm{maximize}_{\beta\in\mathbb{R}^p,\theta\in\mathbb{R}^q}\{\widehat{\mathrm{Cov}}(\mathbf{X}\beta, \mathbf{Y}\theta)\}$$

$$\text{subject to } \widehat{\mathrm{Var}}(\mathbf{X}\beta) = 1 \text{ and } \widehat{\mathrm{Var}}(\mathbf{Y}\theta) = 1$$

- The solution set (β_1, θ_1) are called the first canonical vectors, and the corresponding linear combinations $\mathbf{z}_1 = \mathbf{X}\beta_1$ and $\mathbf{s}_1 = \mathbf{Y}\theta_1$ the first canonical variates.

Sparse CCA

- When $N > \max(p, q)$,

$$\underset{\beta,\theta}{\text{maximize}}\{\widehat{\text{Cov}}(\mathbf{X}\beta, \mathbf{Y}\theta)\}$$
$$\text{subject to } \text{Var}(\mathbf{X}\beta) = 1, \|\beta\|_1 \leq c_1, \text{Var}(\mathbf{Y}\theta) = 1, \|\theta\|_1 \leq c_2$$

- When $N < \max(p, q)$ in this case, the problem is degenerate, and one can find meaningless solutions with correlations equal to one.
  - One approach to avoiding singularity of the sample covariance matrices $\frac{1}{N}\mathbf{X}^T\mathbf{X}$ and $\frac{1}{N}\mathbf{Y}^T\mathbf{Y}$ is by imposing additional restrictions.
  - For instance, the method of ridge regularization is based on adding some positive multiple λ of the identity to each sample covariance matrix.

Bayes rule classifier

Start with the simplest case:

- Let

$$P(\mathbf{X} \in \Pi_i) = \pi_i, \quad i = 1, 2$$

  be the prior probabilities that a randomly selected observation $\mathbf{X} = \mathbf{x}$ belongs to either Π_1 or Π_2.
- Suppose also that the conditional multivariate probability density of $\mathbf{X}$ for the ith class is

$$P(\mathbf{X} = \mathbf{x} \mid \mathbf{X} \in \Pi_i) = f_i(\mathbf{x}), \quad i = 1, 2$$

- From Bayes's theorem yields the posterior probability,

$$P(\Pi_i \mid \mathbf{x}) = P(\mathbf{X} \in \Pi_i \mid \mathbf{X} = \mathbf{x}) = \frac{\pi_i f_i(\mathbf{x})}{\pi_1 f_1(\mathbf{x}) + \pi_2 f_2(\mathbf{x})},$$

  that the observed x belongs to Π_i, $i = 1, 2$.

Bayes rule classifier

- Consider a response variable G falling into one of K classes $\{1, 2, \ldots, K\}$, and a predictor vector $X \in \mathbb{R}^p$. Suppose that $f_k(x)$ is the class-conditional density of X in class $G = k$, and let π_k be the prior probability of class k, with $\displaystyle\sum_{k=1}^{K} \pi_k = 1$. A simple application of Bayes' rule gives us

$$\Pr(G = k \mid X = x) = \frac{\pi_k f_k(x)}{\sum_{\ell=1}^{K} \pi_\ell f_\ell(x)}$$

- For a given $\mathbf{x}$, a reasonable classification strategy is to assign $\mathbf{x}$ to that class with the higher posterior probability.
- This strategy is called the Bayes's rule classifier.

Gaussian linear discrimination

- Suppose moreover that each class density is modeled as a multivariate Gaussian $N(\mu_k, \mathbf{\Sigma}_w)$, with density

$$f_k(x) = \frac{1}{(2\pi)^{p/2} |\mathbf{\Sigma}_w|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T \mathbf{\Sigma}_w^{-1}(x-\mu_k)},$$

based on a common covariance matrix $\mathbf{\Sigma}_w$.

- we find that

$$\begin{aligned}\log \frac{\Pr(G=k \mid X=x)}{\Pr(G=\ell \mid X=x)} &= \log \frac{f_k(x)}{f_\ell(x)} + \log \frac{\pi_k}{\pi_\ell} \\ &= \log \frac{\pi_k}{\pi_\ell} - \frac{1}{2}(\mu_k + \mu_\ell)^T \mathbf{\Sigma}_w^{-1} (\mu_k - \mu_\ell) \\ &\quad + x^T \mathbf{\Sigma}_w^{-1} (\mu_k - \mu_\ell),\end{aligned}$$

an equation linear in x. Consequently, the decision boundary between classes k and ℓ are all vectors x for which $\Pr(G=k \mid \mathbf{X}=x) = \Pr(G=\ell \mid \mathbf{X}=x)$

High dimensional LDA via nearest centroid rule

- In very high dimensions, it is often effective to assume that predictors are uncorrelated, which translates into a diagonal form for $\boldsymbol{\Sigma}_w$.
- Doing so yields the so-called naive Bayes classifier, or alternatively diagonal linear discriminant analysis.
- Letting $\widehat{\sigma}_j^2 = s_j^2$ be the pooled within-class variance for feature j, the estimated classification rule simplifies to

$$\widehat{G}(x) = \underset{\ell=1,\ldots,K}{\arg\min} \left\{ \sum_{j=1}^{p} \frac{\left(x_j - \widehat{\mu}_{j\ell}\right)^2}{\widehat{\sigma}_j^2} - \log \hat{\pi}_k \right\},$$

known as the nearest centroid rule.

Supervised LDA

- In practice, the parameters of the Gaussian class-conditional distributions are not known. However, given N samples $\{(x_1, g_1), \ldots, (x_N, g_N)\}$ of featurelabel pairs, we can estimate the parameters as follows.
  - Let C_k denote the subset of indices i for which $g_i = k$, and let $N_k = |C_k|$ denote the total number of class- k samples.
  - We then form the estimates $\widehat{\pi}_k = N_k/N$, and

$$\widehat{\mu}_k = \frac{1}{N_k} \sum_{i \in C_k} x_i, \text{ and}$$

$$\widehat{\boldsymbol{\Sigma}}_w = \frac{1}{N-K} \sum_{k=1}^{K} \sum_{i \in C_k} (x_i - \widehat{\mu}_k)(x_i - \widehat{\mu}_k)^T.$$

- Note that $\widehat{\boldsymbol{\Sigma}}_w$ is an unbiased estimate of the pooled within-class covariance.

Sample Fisher's LDA

- Let $\mathbf{X}$ be an $N \times p$ matrix of observations, and assume that its columns, corresponding to features, have been standardized to have mean zero. Given such an observation matrix, we seek a low-dimensional projection such that the between-class variance is large relative to the within-class variance.
- As before, let $\widehat{\mathbf{\Sigma}}_w$ be the pooled within-class covariance matrix and $\widehat{\mu}_k$ the classspecific centroids.
- The between-class covariance matrix $\widehat{\mathbf{\Sigma}}_b$ is the covariance matrix of these centroids, given by

$$\widehat{\mathbf{\Sigma}}_b = \sum_{k=1}^{K} \widehat{\pi}_k \widehat{\mu}_k \widehat{\mu}_k^T,$$

treating them as multivariate observations with mass $\hat{\pi}_k$. Note that

$$\widehat{\mathbf{\Sigma}}_t = \frac{1}{N} \mathbf{X}^T \mathbf{X} = \widehat{\mathbf{\Sigma}}_b + \widehat{\mathbf{\Sigma}}_w.$$

Fisher's LDA with sparsity

- Fisher's LDA proceeds by sequentially solving the following problem:

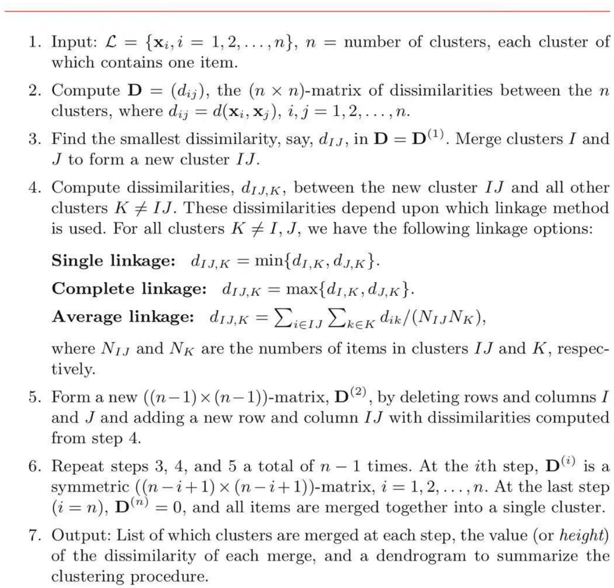
$$\max_{\beta \in \mathbb{R}^p} \left\{ \beta^T \widehat{\boldsymbol{\Sigma}}_b \beta \right\} \text{ s.t. } \beta^T \widehat{\boldsymbol{\Sigma}}_w \beta \leq 1, \text{ and } \beta^T \widehat{\boldsymbol{\Sigma}}_w \widehat{\beta}_\ell = 0 \text{ for all } \ell < k.$$

  for $k = 1, 2, \ldots, \min(K-1, p)$.
- Witten and Tibshirani (2011) proposed a way to "sparsify" the above objective, in particular by solving

$$\underset{\beta}{\text{maximize}} \left\{ \beta^T \widehat{\boldsymbol{\Sigma}}_b \beta - \lambda \sum_{j=1}^{p} \hat{\sigma}_j \left| \beta_j \right| \right\} \text{ subject to } \beta^T \widetilde{\boldsymbol{\Sigma}}_w \beta \leq 1$$

  where $\hat{\sigma}_j^2$ is the j^{th} diagonal element of $\widehat{\boldsymbol{\Sigma}}_w$, and $\widetilde{\boldsymbol{\Sigma}}_w$ is a positive definite estimate for $\boldsymbol{\Sigma}_w$. This produces a first sparse discriminant vector $\widehat{\beta}_1$ with level of sparsity determined by the choice of λ.

Agglomerative Hierarchical Clustering



1. Input: $\mathcal{L} = \{\mathbf{x}_i, i = 1, 2, \ldots, n\}$, n = number of clusters, each cluster of which contains one item.
2. Compute $\mathbf{D} = (d_{ij})$, the $(n \times n)$-matrix of dissimilarities between the n clusters, where $d_{ij} = d(\mathbf{x}_i, \mathbf{x}_j)$, $i, j = 1, 2, \ldots, n$.
3. Find the smallest dissimilarity, say, d_{IJ}, in $\mathbf{D} = \mathbf{D}^{(1)}$. Merge clusters I and J to form a new cluster IJ.
4. Compute dissimilarities, $d_{IJ,K}$, between the new cluster IJ and all other clusters $K \neq IJ$. These dissimilarities depend upon which linkage method is used. For all clusters $K \neq I, J$, we have the following linkage options:

   Single linkage: $d_{IJ,K} = \min\{d_{I,K}, d_{J,K}\}$.

   Complete linkage: $d_{IJ,K} = \max\{d_{I,K}, d_{J,K}\}$.

   Average linkage: $d_{IJ,K} = \sum_{i \in IJ} \sum_{k \in K} d_{ik}/(N_{IJ}N_K)$,

   where N_{IJ} and N_K are the numbers of items in clusters IJ and K, respectively.
5. Form a new $((n-1) \times (n-1))$-matrix, $\mathbf{D}^{(2)}$, by deleting rows and columns I and J and adding a new row and column IJ with dissimilarities computed from step 4.
6. Repeat steps 3, 4, and 5 a total of $n-1$ times. At the ith step, $\mathbf{D}^{(i)}$ is a symmetric $((n-i+1) \times (n-i+1))$-matrix, $i = 1, 2, \ldots, n$. At the last step $(i = n)$, $\mathbf{D}^{(n)} = 0$, and all items are merged together into a single cluster.
7. Output: List of which clusters are merged at each step, the value (or *height*) of the dissimilarity of each merge, and a dendrogram to summarize the clustering procedure.

Sparse Hierarchical Clustering

- Problem: High influence of variables that are irrelevant for clustering.
- Let

$$\Delta = \begin{bmatrix} d_{1,1,1} & \dots & d_{1,1,p} \\ d_{1,2,1} & \dots & \\ \vdots & \ddots & \vdots \\ d_{N,N,1} & \dots & d_{N,N,p} \end{bmatrix} \in \mathbb{R}^{N^2,p}$$

where $d_{i,i',j} = \left(x_{ij} - x_{i'j}\right)^2$. Define $\tilde{D} = \Delta w$, which means that $\tilde{D}_{i,i'} = \sum_{j=1}^{p} w_j d_{i,i',j}$. We seek a w such that $\tilde{D}$ "captures" as much of Δ as possible (can be measured by Frobenius norm). This can be seen as a sparse PCA problem.

Sparse Hierarchical Clustering

- $\Delta \in \mathbb{R}^{N^2 \times p}$ is a matrix with column j consisting of N^2 pairwise dissimilarities for feature j.
- It can be shown that finding w is equivalent with:

$$\begin{aligned}&\text{maximize}_{u \in \mathbb{R}^{N^2}, w \in \mathbb{R}^p} \left\{ u^\top \Delta w \right\} \\ &\text{subject to } \|u\|_2 \leq 1, \|w\|_2 \leq 1, \|w\|_1 \leq s, w \succeq 0\end{aligned}$$

- $\Delta w = \tilde{D} \in \mathbb{R}^{N^2}$, reshaped to $\tilde{D} \in \mathbb{R}^{N \times N}$. Then apply hierarchical clustering.

K-means

- Goal: Partition observations into K homogeneous groups based on minimizing cluster sum of squares:

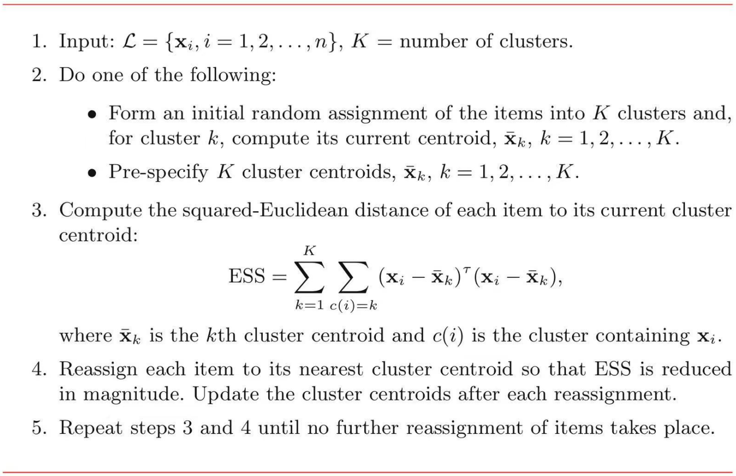
$$W(\mathcal{C}) = \sum_{k=1}^{K} \sum_{i \in \mathcal{C}_k} \|x_i - \bar{x}_k\|_2^2$$

- $\{\bar{x}_k\}_1^K$ codebook vectors. $\bar{x}_k = \sum_{i \in \mathcal{C}_k} \frac{x_i}{N_k}$, where $N_k = |\mathcal{C}_k|$.
- $c(i)$ encoder which assigns x_i to the cluster of closest centroid.

$$C_k = \{i : c(i) = k\}$$

- Iterative solution: $C(i) = \underset{1 \le k \le K}{\operatorname{argmin}} \|x_i - m_k\|^2$.

K-means



1. Input: $\mathcal{L} = \{\mathbf{x}_i, i = 1, 2, \ldots, n\}$, K = number of clusters.
2. Do one of the following:
   - Form an initial random assignment of the items into K clusters and, for cluster k, compute its current centroid, $\bar{\mathbf{x}}_k$, $k = 1, 2, \ldots, K$.
   - Pre-specify K cluster centroids, $\bar{\mathbf{x}}_k$, $k = 1, 2, \ldots, K$.
3. Compute the squared-Euclidean distance of each item to its current cluster centroid:
$$\text{ESS} = \sum_{k=1}^{K} \sum_{c(i)=k} (\mathbf{x}_i - \bar{\mathbf{x}}_k)^\tau (\mathbf{x}_i - \bar{\mathbf{x}}_k),$$
   where $\bar{\mathbf{x}}_k$ is the kth cluster centroid and $c(i)$ is the cluster containing $\mathbf{x}_i$.
4. Reassign each item to its nearest cluster centroid so that ESS is reduced in magnitude. Update the cluster centroids after each reassignment.
5. Repeat steps 3 and 4 until no further reassignment of items takes place.

Sparse K-means

- It holds that

$$\sum_{i,i' \in \mathcal{C}_k} \|x_i - x_{i'}\|_2^2 = 2N_k \sum_{i \in \mathcal{C}_k} \|x_i - \bar{x}_k\|_2^2$$

- Thus the cost-function we try to minimize is

$$W(\mathcal{C}) = \sum_{k=1}^{K} \frac{1}{N_k} \sum_{i,i' \in \mathcal{C}_k} \sum_{j=1}^{p} d_{i,i',j}$$

- Again suppose not all variables are important for clustering:

$$\begin{array}{ll} \text{minimize}_{\mathcal{C}, w \in \mathbb{R}^p} & \sum_{j=1}^{p} w_j \left(\sum_{k=1}^{K} \frac{1}{N_k} \sum_{i,i' \in \mathcal{C}_k} d_{i,i',j} \right) \\ \text{subject to} & \|w\|_2 \leq 1, \|w\|_1 \leq s, w \succeq 0 \end{array}$$

- problem is convex in w. However, it has the solution $\hat{w} = 0$.

Sparse K-means

We have the following observation

$$\text{minimize } W(\mathcal{C}) \iff \text{maximize } B(\mathcal{C})$$

Can be seen from

$$\begin{aligned} T &:= \sum_{i=1}^{N} \sum_{i'=1}^{N} \left(\sum_{j=1}^{p} d_{i,i',j} \right) \\ &= \sum_{k=1}^{K} \sum_{i \in \mathcal{C}_k} \left(\sum_{i' \in \mathcal{C}_k} \sum_{j=1}^{p} d_{i,i',j} + \sum_{i' \notin \mathcal{C}_k} \sum_{j=1}^{p} d_{i,i',j} \right) \\ &= W(\mathcal{C}) + B(\mathcal{C}) \end{aligned}$$

where T (total cost) is constant.

Sparse K-means

- This leads us to the optimization problem

$$\begin{array}{ll} \text{maximize}_{\mathcal{C}, w \in \mathbb{R}^p} & \sum_{j=1}^{p} w_j \left(\frac{1}{N} \sum_{i=1}^{N} \sum_{i'=1}^{N} d_{i,i',j} - \sum_{k=1}^{K} \frac{1}{N_k} \sum_{i,i' \in \mathcal{C}_k} d_{i,i',j} \right) \\ \text{subject to} & \|w\|_2 \leq 1, \|w\|_1 \leq s, w \succeq 0 \end{array}$$

- Iterative scheme:
  - Hold $\mathcal{C}$ fixed. Maximize with respect to w.
  - Hold w fixed. Optimize with respect to $\mathcal{C}$ (Weighted K-means).
- Further discussions can be found in *A framework for feature selection in clustering* (Witten et al., 2010)

References

- ISLR, SLS and ESL,
- Modern Multivariate Statistical Techniques: Regression, Classification, and Manifold Learning
- Multivariate analysis I, lecture notes, SYSU
- CS 168: The Modern Algorithmic Toolbox, Stanford
- Sparse learning slides, ETHZ

Questions or comments?