Visualisation and Dimension Reduction

Christophe Ambroise

christophe.ambroise@univ-evry.fr

UEVE, UMR CNRS 8071

November 7, 2023

Factor Analyser

Factor Analysis

- Using discrete latent variables provides limited summary (clustering)
- An alternative is to use a vector of real-valued latent variables, $\boldsymbol{z} \in \mathbb{R}^L$.
- “Factor analysis (FA) is a statistical method used to describe variability among observed, correlated variables in terms of a potentially lower number of unobserved variables called factors.” Wikipedia quote.
- PCA and FA are related, but not identical.

The model of factor analysis

We consider the observation $\boldsymbol{x} \in \mathbb{R}^D$

$$\boldsymbol{x} = \boldsymbol{W}\boldsymbol{z} + \boldsymbol{\mu} + \epsilon$$

where

- the noise $\boldsymbol{\epsilon} \sim \mathcal{N}_D(\mathbf{0}, \boldsymbol{\Psi})$
- the hidden (latent) vector $\boldsymbol{z} \sim \mathcal{N}_L(\mathbf{0}, \boldsymbol{I}_L)$

$$p(\boldsymbol{x}|\boldsymbol{z}, \theta) = \mathcal{N}(\boldsymbol{W}\boldsymbol{z} + \boldsymbol{\mu}, \boldsymbol{\Psi})$$

the mean is a linear function of the (hidden) inputs

- $\boldsymbol{W}$ is a $D \times L$ matrix, known as the factor loading matrix,
- $\boldsymbol{\Psi}$ is a $D \times D$ covariance matrix that we take to be diagonal

The special case in which $\boldsymbol{\Psi} = \sigma^2 I$ is called probabilistic principal components analysis or PPCA.

Reminder: Joint and conditional Gaussian distribution (see Murphy chapter 4)

Let us recall that if

- $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}_z, \boldsymbol{\Sigma}_{zz})$ and $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_{xx})$ and $\boldsymbol{cov}(\boldsymbol{z}, \boldsymbol{x}) = \boldsymbol{\Sigma}_{zx}$

then

$$p(\boldsymbol{z}, \boldsymbol{x}) = N(\begin{bmatrix} \boldsymbol{z} \\ \boldsymbol{x} \end{bmatrix} | \begin{bmatrix} \boldsymbol{\mu}_z \\ \boldsymbol{\mu}_x \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{zz} & \boldsymbol{\Sigma}_{zx} \\ \boldsymbol{\Sigma}_{xz} & \boldsymbol{\Sigma}_{zz} \end{bmatrix})$$

and

$$p(\boldsymbol{z}, \boldsymbol{x}) = p(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu}_{z|x}, \boldsymbol{\Sigma}_{z|x})\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_{xx})$$

where

- $\boldsymbol{\mu}_{z|x} = \boldsymbol{\mu}_z + \boldsymbol{\Sigma}_{zx}\boldsymbol{\Sigma}_{xx}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_x)$
- $\boldsymbol{\Sigma}_{z|x} = \boldsymbol{\Sigma}_{zz} - \boldsymbol{\Sigma}_{zx}\boldsymbol{\Sigma}_{xx}^{-1}\boldsymbol{\Sigma}_{xz}$

Marginal and posterior distribution

Marginal distribution

$$\boldsymbol{x} \sim \mathcal{N}_D(\boldsymbol{\mu}, \boldsymbol{\Sigma}_{\boldsymbol{xx}} = \boldsymbol{W}\boldsymbol{W}^{\boldsymbol{T}} + \boldsymbol{\Psi})$$

Posterior distribution

$$\boldsymbol{z}|\boldsymbol{x} \sim \mathcal{N}_L(\boldsymbol{\mu}_{z|x}, \boldsymbol{\Sigma}_{z|x})$$

where

- $\boldsymbol{\Sigma}_{z|x} = (\boldsymbol{I_L} + \boldsymbol{W}^{\boldsymbol{T}}\boldsymbol{\Psi}^{-\boldsymbol{1}}\boldsymbol{W})^{-\boldsymbol{1}} = S$
- $\boldsymbol{\mu}_{z|x} = \boldsymbol{\Sigma}_{\boldsymbol{z}|\boldsymbol{x}}\boldsymbol{\Sigma}_{\boldsymbol{xx}}^{-\boldsymbol{1}}(\boldsymbol{x} - \boldsymbol{\mu}) = \boldsymbol{S}\boldsymbol{W}^{\boldsymbol{T}}\boldsymbol{\Psi}^{-\boldsymbol{1}}(\boldsymbol{x} - \boldsymbol{\mu})$

Exercice

Demonstrate the above formulas

Estimation

The mean $\boldsymbol{\mu}$

can be estimated by maximum likelihood

$$\boldsymbol{\mu}_{mle} = \bar{\boldsymbol{x}}$$

$\boldsymbol{W}$ *and* $\boldsymbol{\Psi}$

are estimated using an EM algorithm

EM algorithm

Data

- Observed data **:** $\boldsymbol{x}_{1:n}$
- Missing (or hidden) data : **:** $z_{1:n}$

Principle

- Starting from θ^0
- At step q
  - → E(xpectation) step: $Q(\theta, \theta^q) = E_{Z_{1:n}|\boldsymbol{x}_{1:n}}[\log P(\boldsymbol{x}_{1:n}, \boldsymbol{z}_{1:n}, \theta)]$
  - → M(aximisation) step: $\theta^{q+1} = argmax_\theta Q(\theta, \theta^q)$

EM for factor analysis

Let us assume that $\boldsymbol{\mu} = \mathbf{0}$ (centering of the $\boldsymbol{x}_i$), the complete log-likelihood is

$$
\begin{aligned}
\log p(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\mu}, \boldsymbol{W}, \boldsymbol{\Psi}) = & \sum_i \log \mathcal{N}_L(z_i; \mathbf{0}, \boldsymbol{I}) + \log \mathcal{N}_D(x_i; \\
= & -\frac{n}{2} \log |\boldsymbol{I}_L| - \frac{n}{2} Tr(\hat{\boldsymbol{\Sigma}}_{\boldsymbol{zz}}) \\
& -\frac{n}{2} \log |\boldsymbol{\Psi}| - \frac{n}{2} Tr(\hat{\boldsymbol{\Sigma}}_{\boldsymbol{xx}} \boldsymbol{\Psi}^{-1}) +
\end{aligned}
$$

where

$$\hat{\boldsymbol{\Sigma}}_{\boldsymbol{xx}} = \frac{1}{n}\sum_i(\boldsymbol{x}_i - \boldsymbol{W}\boldsymbol{z_i})(\boldsymbol{x}_i - \boldsymbol{W}\boldsymbol{z_i})^T$$

Exercice

Demonstrate the above formula

E step

The expectation of the complete log-likelihood requires

1. $\mathbb{E}_{z|x}[\boldsymbol{z}_i] = \boldsymbol{S}\boldsymbol{W}^{\boldsymbol{T}}\boldsymbol{\Psi}^{-\boldsymbol{1}}(\boldsymbol{x}_i - \boldsymbol{\mu})$ where $\boldsymbol{S} = (\boldsymbol{I_L} + \boldsymbol{W}^{\boldsymbol{T}}\boldsymbol{\Psi}^{-\boldsymbol{1}}\boldsymbol{W})^{-\boldsymbol{1}}$
2. $\mathbb{E}_{z|x}[\boldsymbol{z_i}\boldsymbol{z_i^T}] = \boldsymbol{E}_{z|x}[\boldsymbol{z}_i]\boldsymbol{E}_{z|x}[\boldsymbol{z}_i^T] + \boldsymbol{S}$

M step

Reminders

$$\frac{\partial (b^T a)}{\partial a} = b$$

$$\frac{\partial (a^T A a)}{\partial a} = (A + A^T)a$$

$$\frac{\partial}{\partial A} tr(BA) = B^T$$

$$\frac{\partial}{\partial A} \log |A| = (A^{-1})^T$$

$$tr(ABC) = tr(CAB) = tra(BCA)$$

Thus if x is a vector

$$x^T A x = tr(x^T A x) = tr(A x x^T)$$

M step for $\boldsymbol{\Psi}$

$$\mathbb{E}_{z|x}\left[\frac{\partial L(\boldsymbol{W}, \boldsymbol{\Psi})}{\partial \boldsymbol{\Psi}^{-1}}\right] = \mathbb{E}_{z|x}\left[\frac{n}{2}(\boldsymbol{\Psi} - \hat{\boldsymbol{\Sigma}}_{\boldsymbol{xx}})\right] = \frac{n}{2}(\boldsymbol{\Psi} - \mathbb{E}_{z|x}$$

where

$$\begin{aligned}
\mathbb{E}_{z|x}\left[\hat{\boldsymbol{\Sigma}}_{\boldsymbol{xx}}\right] &= \frac{1}{n}\Big(\sum_i \boldsymbol{x_i}\boldsymbol{x_i^T} + W(\sum_i \mathbb{E}_{z|x}\left[\boldsymbol{z_i}\boldsymbol{z_i^T}\right])W^T - 2 \\
&= \frac{1}{n}\Big(\sum_i \boldsymbol{x_i}\boldsymbol{x_i^T} + W(\sum_i \mathbb{E}_{z|x}\left[\boldsymbol{z_i}\boldsymbol{x_i^T}\right]) - 2W \\
&= \frac{1}{n}\Big(\sum_i \boldsymbol{x_i}\boldsymbol{x_i^T} - W\sum_i \boldsymbol{E_{z|x}}\left[\boldsymbol{z_i}\right]\boldsymbol{x_i^T}\Big)
\end{aligned}$$

M step for $\boldsymbol{W}$

$$\mathbb{E}_{z|x}\left[\frac{\partial L(\boldsymbol{W}, \boldsymbol{\Psi})}{\partial \boldsymbol{W}}\right] = \mathbb{E}_{z|x}\Bigg[-\boldsymbol{\Psi}^{-1}\sum_i x_i z_i^T + \boldsymbol{\Psi}^{-1}W\sum_i z$$

M Step summary

Loading matrix

$$\boldsymbol{W}^{q+1} = \left(\sum_i (\boldsymbol{x}_i - \bar{\boldsymbol{x}})\mathbb{E}_{z|x}[\boldsymbol{z}_i]^T\right)\left(\sum_i \mathbb{E}_{z|x}[\boldsymbol{z}_i\boldsymbol{z}_i^T]\right)^{-1}$$

Noise covariance matrix

$$\boldsymbol{\Psi}^{q+1} = \frac{1}{N} diag\left\{\sum_i \boldsymbol{x_i}\boldsymbol{x_i^T} - \boldsymbol{W}^{q+1}\mathbb{E}_{z|x}[\boldsymbol{z}_i]\boldsymbol{x}_i^T\right\}$$

Log-likelihood

The log-likelihood can be computed using the EM decomposition

$$\log P(X; \boldsymbol{\Theta}) = E_{Z_{1:n}|\boldsymbol{x}_{1:n}}[\log P(\boldsymbol{x}_{1:n}, \boldsymbol{z}_{1:n}; \theta)] - E_{Z_{1:n}|\boldsymbol{x}_{1:n}}[\log$$

Implementation of the algorithm

Initialisation via a PCA

```
initialisation.FA<-function(X,L=1){
  # Return W and Psi
  d<-ncol(X)
  Sigmaxx<-var(X)
  W<-eigen(Sigmaxx)$vectors[,1:L]
  if (L==1) W<-cbind(W)
  Psi<-rep(1,d)
  return(list(W=W,Psi=Psi))
}
```

E step

```
FA.E.step<-function(X,W,Psi){
  # X is assumed to be centered
  # M contain the contionnal expectation of the latent factor
  # S contains the covariance of the latent factor
  L<-ncol(W)
  S <- solve(diag(L) + t(W)%*%diag(1/Psi)%*%W)
  M<- X%*%diag(1/Psi)%*%W%*%S
  return(list(S=S,M=M))
}
```

M Step

```
FA.M.step<-function(X,S,M,W,Psi){
  n<-nrow(X)
  Psi<-1/n*diag(t(X)%*%X -W%*%t(M)%*%X)
  W<- (t(X)%*%M)%*%solve(n*S+t(M)%*%M)
  return(list(Psi=Psi,W=W))
}
```

Computation of the criterion

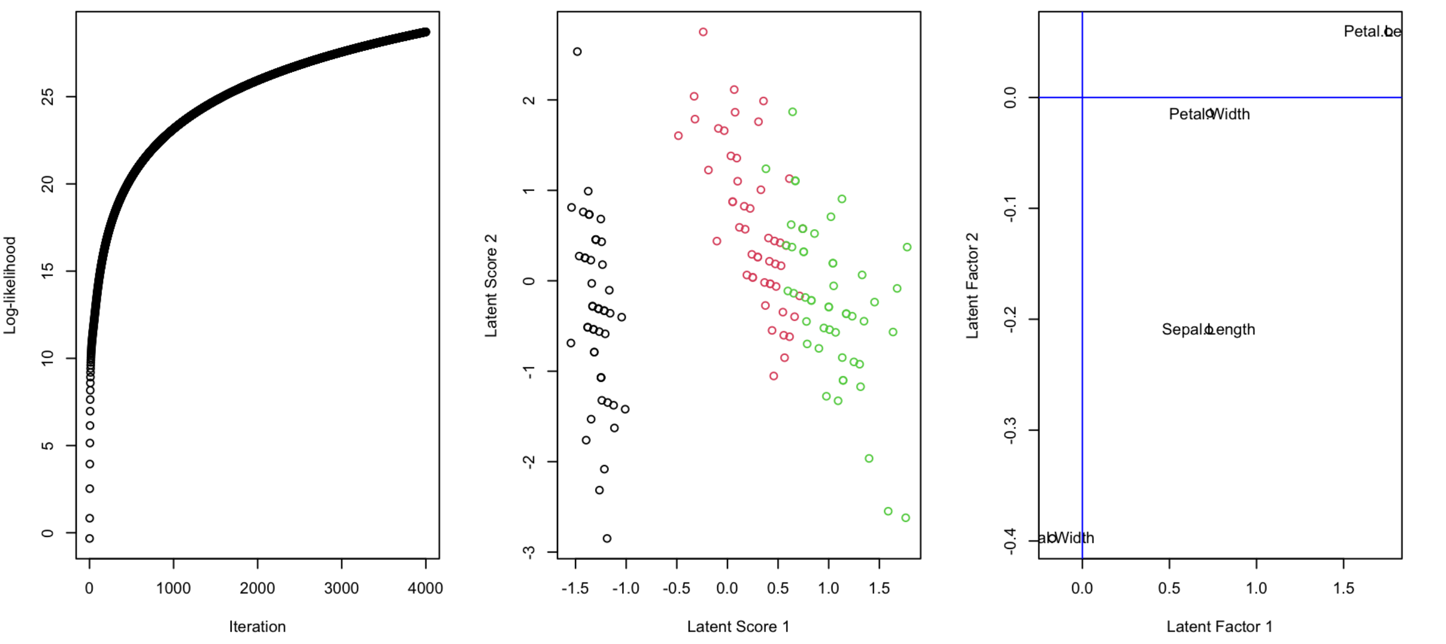
```
log.likelihood.FA<-function(X,S,M,Psi,W){
  n<-nrow(X)
  Sigmax<-(t(X)%*%X-W%*%t(M)%*%X)
  return(-(sum(diag(S+t(M)%*%M/n))
          +log(det(diag(Psi)))+
            log(det(S))+
            1/n*sum(diag(Sigmax%*%diag(1/Psi)))))
}
```

Putting it all together

```
FA.EM<-function(X,L=1,max.iter=50){
  X<-scale(X,scale=FALSE);mu<-attr(X,"scaled:center")
  log.likelihood<-NULL; init<-initialisation.FA(X,L)
```

```
  W<-init$W;  Psi<-init$Psi; criterion<- Inf; iteration<-1;
  log.likelihood[iteration]<--Inf
  while ((criterion>1e-6)&&(iteration<=max.iter)){
    E.step<-FA.E.step(X,W,Psi); E.step$S->S; E.step$M->M
    M.step<-FA.M.step(X,S,M,W,Psi); M.step$Psi->Psi; M.step$W->W
    iteration<-iteration+1
    log.likelihood[iteration]<-log.likelihood.FA(X,S,M,Psi,W)
    criterion<-abs((log.likelihood[iteration] - log.likelihood[iteration-1])/max(log
  }
  return(list( W=data.frame(W), Psi=Psi,
               M=data.frame(M), S=S,mu=mu,
               log.likelihood= log.likelihood[-1]))
}
```


Example with the Iris



Unidentifiability

- If we consider $\boldsymbol{R}$ an orthogonal rotation matrix such that

$$\boldsymbol{R}\boldsymbol{R}^T = \boldsymbol{I}$$

It appears that $\tilde{\boldsymbol{W}} = \boldsymbol{W}\boldsymbol{R}$ produces the same log-likelihood.

- $\boldsymbol{W}$ cannot be uniquely identified.

Possible rotations

1. Forcing $\boldsymbol{W}$ to be orthogonal with colmuns ordered by deacresing variance
2. Forcing $\boldsymbol{W}$ to be lower triangular (problem of founder variables)
3. Choosing an informative rotation matrix. For example the varimax rotation.
4. …

Varimax

Varimax rotation maximizes the sum of the variance of the squared correlations between variables and factors

$$R_{\text{VARIMAX}} = \arg\max_{R} \left(\frac{1}{p} \sum_{j=1}^{k} \sum_{i=1}^{p} (WR)_{ij}^{4} - \sum_{j=1}^{k} \left(\frac{1}{p} \sum_{i=1}^{p} (\right.\right.$$

This results in high factor loadings for a small number of variables and low factor loadings for the rest.

Varimax

```
W.FA<-FA.result$W
W.Varimax<-varimax(as.matrix(FA.result$W))$loadings
print(W.Varimax)
```

```
Loadings:
             Latent Factor 1 Latent Factor 2
Sepal.Length  0.756                         
Sepal.Width                  -0.429         
Petal.Length  1.683           0.509         
Petal.Width   0.711           0.174         

               Latent Factor 1 Latent Factor 2
SS loadings              3.916           0.473
Proportion Var           0.979           0.118
Cumulative Var           0.979           1.097
```

Mixture of factor analysers

- Factor analyses is a way to estimate a variance matrix with few parameters
- This property can be used in the context of Gaussian mixture model assuming the following parameterization for component densities:

$$p(\boldsymbol{x}_i|\boldsymbol{z}_i, q_i = k) = \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_k + \boldsymbol{W}_k\boldsymbol{z}_i + \boldsymbol{\Psi})$$

  where $\boldsymbol{k}$ is the component number and W_k is a loading matrix defining the relation between the observation $\boldsymbol{x}_i$ and the latent vector $\boldsymbol{z}_i$
- This approach is simular to the Banfield-Raftery idea of decomposing the component variance matrix $\boldsymbol{k}$ in volume, form et direction.

Relation to principal component analysis

Assumption

If

- $\boldsymbol{\Psi} = \sigma^2 \boldsymbol{I}$
- $\boldsymbol{W}$ is orthonal

and

- $\sigma^2 \to 0$

Then

Tipping, M. and C. Bishop (1999, Probabilistic principal component analysis. J. of Royal Stat. Soc. Series B 21(3), 611–622) showed that FA is equivalent to PCA

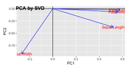
Criterion

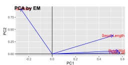
$$J(\boldsymbol{W}, \boldsymbol{Z}) = \|\boldsymbol{X} - \boldsymbol{Z}\boldsymbol{W}^T\|_F^2$$

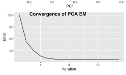
where $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}$

A Constrained EM Algorithm for PCA (from Ahn, J.-H. and J.-H. Oh, 2003)

```r
upper<-function(A){A[lower.tri(A,diag=FALSE)]<-0;return(A)}
lower<-function(A){A[upper.tri(A,diag=FALSE)]<-0;return(A)}
PCA.EM<-function(X,q=2){
  p<-ncol(X); n<-nrow(X)
  W<-diag(p)[,1:q]; M<-X%*%W # Initialisation
  Jold<-0; J<-1; iteration<-0; Error<-NULL
  while ((abs(J - Jold)>1e-3)){
    Jold<-sum((X-M%*%t(W))^2)
    S <- solve(upper(t(W)%*%W)); M<- X%*%W%*%S # E-step
    W<- (t(X)%*%M)%*%solve(lower(n*S+t(M)%*%M))# M-step
    W<-apply(W,2,function(x){x/sqrt(sum(x^2))})#orthogonalisation
    J<-sum((X-M%*%t(W))^2); Error[iteration<-iteration+1]<-J
  }
  return(list(W=data.frame(W),M=data.frame(M),Error=Error))}
```







Neural networks and unsupervised learning