# Ch13. Linear Factor Models

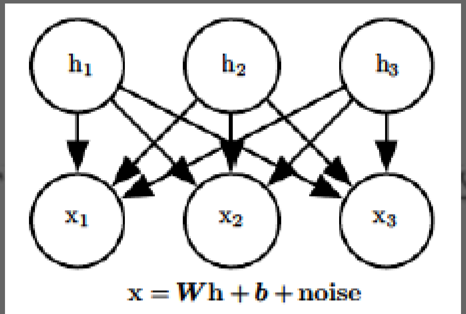
Jihyeong Jung
Department of Mathematics, KAIST

# Contents

# Intro.

- Linear Factor Models

- About Linear Factor Model
  - The simplest probabilistic. models with latent variables
  - Defined by the use of stochastic linear decoder function; $\boldsymbol{x} = \boldsymbol{W}\boldsymbol{h} + \boldsymbol{b} + \boldsymbol{\epsilon}$
- Data generation process
  - Sample $\boldsymbol{h}$ from dist. $\boldsymbol{h} \sim p(\boldsymbol{h})$ s.t. $p(\boldsymbol{h}) = \prod_i p(h_i)$
  - Sample $\boldsymbol{x}$ based on given factors; $\boldsymbol{x} = \boldsymbol{W}\boldsymbol{h} + \boldsymbol{b} + \boldsymbol{\epsilon}$
    - Noise $\boldsymbol{\epsilon}$ is typically Gaussian & diagonal; indep. across dimensions.
  - Illustration :



# PPCA & FA

- Outlines & about FA

- Probabilistic PCA & FA, etc. are special cases of prev. slide
  - Difference : Choosing dist. of noise $\boldsymbol{\epsilon}$ and prior of $\boldsymbol{h}$ before observing $\boldsymbol{x}$
- FA : prior of latent variable $\boldsymbol{h}$ is just; Gaussian $\boldsymbol{h} \sim \mathcal{N}(\boldsymbol{h}; \mathbf{0}, \boldsymbol{I})$
  - Given $\boldsymbol{h}$, observed $x_i$s are assumed to be conditionally indep.
  - Noise $\boldsymbol{\epsilon}$ is assumed to be; $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \boldsymbol{\Psi}),\ s.t. \boldsymbol{\Psi} = \boldsymbol{diag}(\boldsymbol{\sigma}^2)$ & $\boldsymbol{\sigma}^2 = [\sigma_i^2]^T$
- The role of latent variables : *'capture the dependencies'* b/w the different observed $\boldsymbol{x}$, it's easy to see that $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}; \boldsymbol{b}, \boldsymbol{W}\boldsymbol{W}^T + \boldsymbol{\Psi})$

# PPCA & FA

- About Probabilistic PCA

- PCA in a ‘Probabilistic’ framework; modify to FA model
  - Set $\sigma_i^2 = \sigma^2\ for\ \forall i = 1, 2, \dots, n; so\ that\ \boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}; \boldsymbol{b}, \boldsymbol{W}\boldsymbol{W}^T + \sigma^2 \boldsymbol{I})$
  - Equivalently, $\boldsymbol{x} = \boldsymbol{W}\boldsymbol{h} + \boldsymbol{b} + \sigma\boldsymbol{\epsilon}\ s.t.\ \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \boldsymbol{0}, \boldsymbol{I})$
  - By iterative EM algorithm, we can estimate $\boldsymbol{W}$ & $\sigma^2$

- Based on the observation that most variations in the data can be captured by $\boldsymbol{h}$, up to some small residual reconstruction error
  - As $\sigma \to 0$, then $\boldsymbol{x} = \boldsymbol{W}\boldsymbol{h} + \boldsymbol{b} + \sigma\boldsymbol{\epsilon} \xrightarrow{p} \boldsymbol{x} = \boldsymbol{W}\boldsymbol{h} + \boldsymbol{b}$ (PPCA $\xrightarrow{p}$ PCA)

# ICA

- About Independent Component Analysis

- Forming observed data by separating observed signal into many underlying signals that are scaled & added together
  - An approach to modeling linear factors
  - Signals are intended to be not only uncorrelated from each other, but also fully independent.
- There exists many variances referred to as ICA
- Introduced variant : training fully parametric model
  - Prior of the underlying factors, $p(\boldsymbol{h})$, must be fixed in advance.
  - Then the model generates $\boldsymbol{x} = \boldsymbol{W}\boldsymbol{h}$, determine $p(\boldsymbol{x})$ by transformation
  - And learning model using maximum likelihood

# ICA

- Motivation of ‘The’ Variant/some examples

- By choosing $p(\boldsymbol{h})$ to be independent, we can recover underlying factors as close as possible to be independent
  - Commonly used to recover mixed low-level signals
  - Ex) commonly used in neuroscience, for the electro-encephalography
    - To separate signals of the heart from signals of the brain
    - And to separate signals in different regions of the brain from each other

# ICA

- Non-Gaussian prior requirement

- All variants have common requirement: non-Gaussian prior $p(\boldsymbol{h})$
  - We cannot identify $\boldsymbol{W}$ if $p(\boldsymbol{h}) \sim Gaussian\ Normal$
  - In other words, we can obtain same $p(\boldsymbol{x})$ for many $\boldsymbol{W}$
- Quite different from other LFMs such as PPCA, FA which require Gaussian prior $p(\boldsymbol{h})$ to make operations have closed solutions
- For 'the' variant, we typically use $p(h_i) = \frac{d}{dh_i}\sigma(h_i) = \frac{d}{dh_i}(\frac{1}{1+e^{-h_i}})$
  - Have larger peaks near 0 than Gaussian
- So most ICA implementations aim at learning sparse features

# ICA
- Another Variants

- There's a variant that adds some noise in the generation of $\boldsymbol{x}$ rather than deterministic decoder
  - It aims to make the elements of $\boldsymbol{h} = \boldsymbol{W}^{-1}\boldsymbol{x}$ independent from each other
- Some variants constrains $\boldsymbol{W}$ to be orthogonal to avoid possibly problematic operation determinant
- Some variants is not a 'generative model'
  - Many variants only know transformation b/w $\boldsymbol{x}$ & $\boldsymbol{h}$, but do not have any way of representing p($\boldsymbol{h}$) so it does not impose distribution over $\mathrm{p}(\boldsymbol{x})$
  - Ex) many variants aim to increase sample kurtosis of $\boldsymbol{h} = \boldsymbol{W}^{-1}\boldsymbol{x}$
    - d

# ICA

- Generalizations : NICE/ISA/Topographic ICA

- NICE(Non-linear Independent Components Estimation)
- ISA(Independent Subspace Analysis)
- Topographic ICA

# SFA

- About Slow Feature Analysis

- LFM that uses information from time signals to learn invariant features
- Based on the Slowness principle
  - Idea : important characteristics of scenes change very slowly
  - Can be applied to any differentiable model trained with GD generally
  - By adding a term to cost function of the form $\lambda \sum_t L(f(\boldsymbol{x}^{(t+1)}), f(\boldsymbol{x}^{(t)}))$ to apply the Slowness principle
- Efficient application of the slowness principle; because it is applied to a linear feature extractor

# SFA

- SFA Algorithm

- Defining $f(\boldsymbol{x};\boldsymbol{\theta})$ as a linear transformation
  - And solve; $\min_{\boldsymbol{\theta}} \mathbb{E}_t\left[\left(f\left(\boldsymbol{x}^{(t+1)}\right)_i - f\left(\boldsymbol{x}^{(t)}\right)_i\right)^2\right]$,
  - With constraints; $\mathbb{E}_t f\left(\boldsymbol{x}^{(t)}\right)_i = 0$ & $\mathbb{E}_t\left[\left(f\left(\boldsymbol{x}^{(t)}\right)_i\right)^2\right] = 1$
  - 1st constraint : to obtain a unique solution
  - 2nd constraint : to prevent solution where all features collapse to 0
- SFA features are ordered like PCA
  - The first feature is the slowest one

# SFA

- SFA Algorithm/Generalizations/Advantages

- Additional constraint to learn multiple features;
  - $\mathbb{E}_{\boldsymbol{t}}\left[f\left(\boldsymbol{x}^{(t)}\right)_i f\left(\boldsymbol{x}^{(t)}\right)_j\right] = 0\ \ for\ any\ i < j$
  - So the learned features must be linearly uncorrelated from each other
- Typically used to learn non-linear features by applying nonlinear expansion to input signals before running it
- Major advantage : possible to theoretically predict which features SFA will learn
  - Even in the deep nonlinear setting

# Sparse Coding

- Training Sparse Coding model

- Encoder of the SC model is kind of optimization algorithm;
  - Solving $\boldsymbol{h}^* = \mathrm{argmax}_{\boldsymbol{h}}\, p(\boldsymbol{h}|\boldsymbol{x}) = \mathrm{argmin}_{\boldsymbol{h}}\, \lambda||\boldsymbol{h}||_{\mathbf{1}} + \beta||\boldsymbol{x} - \boldsymbol{W}\boldsymbol{h}||_{\mathbf{2}}^{\mathbf{2}}$
  - This yields a sparse $\boldsymbol{h}^*$ because of the L1 norm imposition on $\boldsymbol{h}$
- To train model, we alternate b/w minimization wrt $\boldsymbol{h}$ and $\boldsymbol{W}$
  - We treat $\beta$ as a hyper-parameter; we typically set it to 1 because it's role is shared with $\lambda$ in opt. problem
  - If we want to learn $\beta$, we must consider discarded terms

# Sparse Coding

- Advantages

- Sparse Coding + non-parametric encoder
  - Can minimize the (reconstruction error + log-prior) better than any specific parametric encoder in principle
  - There's no generalization error; so applying sparse coding to feature extractor for a classifier makes better generalization than applying it as parametric func. to predict the code

# Sparse Coding

- Disadvantages

- non-parametric encoder
  - The primary disadvantage : large time cost of calculating $(\boldsymbol{h}|\boldsymbol{x})$ because of iterative algorithm of non-parametric approach
  - Not easy to use back-prop. through the non-parametric encoder
- Often produces poor samples like other LFMs
  - Motivated the development of improved models