V. Linear Method and Auto-regressive Model

Young-geun Kim

Department of Statistics and Probability

STT 997 (SS 2025)

MICHIGAN STATE
UNIVERSITY

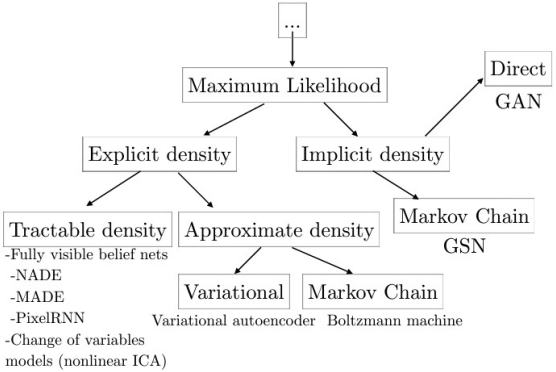
Lecture Topics

1. Introduction
2. Preliminary Knowledge - Statistics
3. Preliminary Knowledge - Statistical Learning
4. Preliminary Knowledge - Python and PyTorch Implementations
5. Linear Method and Auto-regressive Model
6. Energy-based Model
7. Variational Autoencoders
8. Generative Adversarial Networks
9. PyTorch Implementation
10. Optimal Transport-based Method
11. Score-based Method

Lecture Topics

1. Introduction
2. Preliminary Knowledge - Statistics
3. Preliminary Knowledge - Statistical Learning
4. Preliminary Knowledge - Python and PyTorch Implementations
5. Linear Method and Auto-regressive Model
6. Energy-based Model
7. Variational Autoencoders
8. Generative Adversarial Networks
9. PyTorch Implementation
10. Optimal Transport-based Method
11. Score-based Method

A Taxonomy of Deep Generative Models

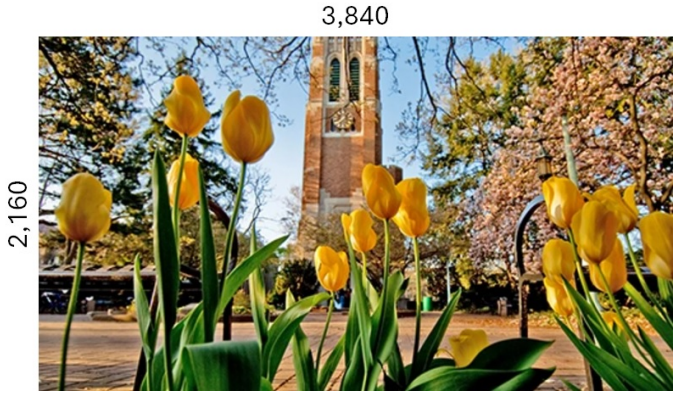


- The goal of the second module is to provide a comprehensive understanding of generative models based on maximum likelihood estimation.

The figure is from Goodfellow (2016).

Outline

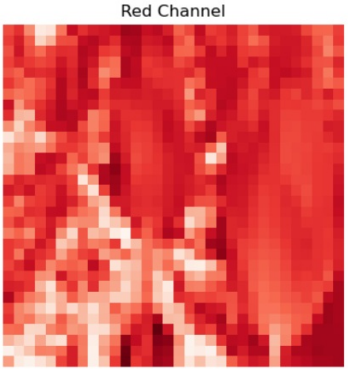
Dimension Reduction: Motivating Example

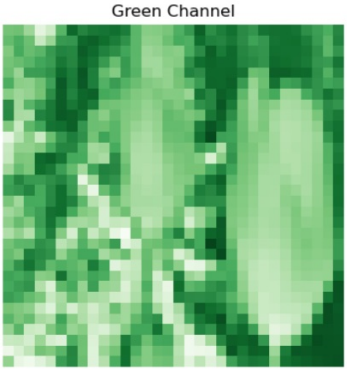


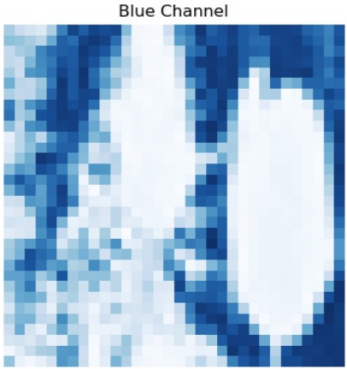
- Let $\vec{X} := (X_1, \ldots, X_p)^T \in \mathcal{X}^p$ represent a p-dimensional random vector for the 4K-resolution color images where $p \approx 24M$.
- Each color channel value is discrete, ranging from 0 to 255 ($|\mathcal{X}| = 256$).

Dimension Reduction: Motivating Example

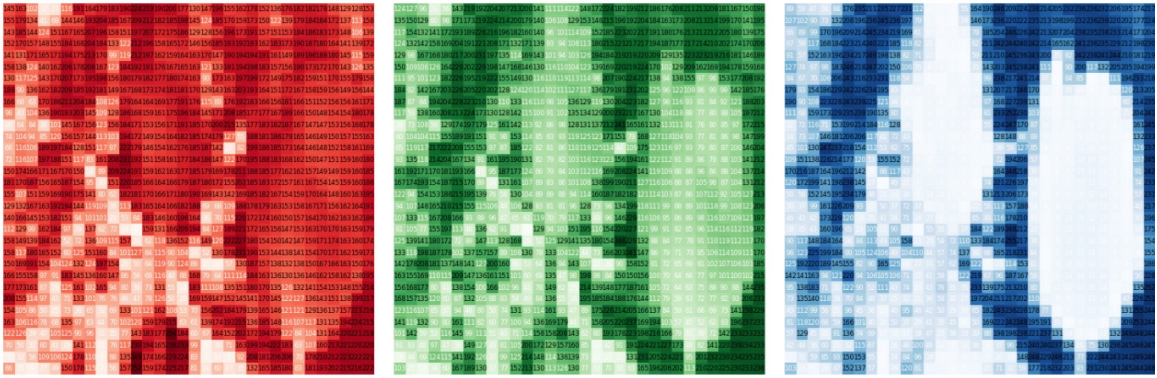
Dimension Reduction: Motivating Example







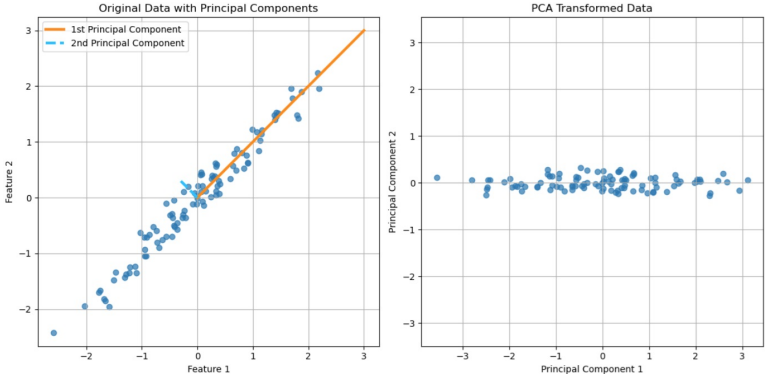
Dimension Reduction: Motivating Example



Dimension Reduction

- *Dimension reduction* is the task of identifying a low-dimensional feature space that efficiently captures essential information about the original data and transforming the data accordingly.
- In this subsection, we review several traditional learning methods:
  1. Principal Component Analysis
  2. Partial Least Squares Regression
  3. Auto-encoders

Linear Method: Principal Component Analysis



- Principal Component Analysis (PCA) aims to rotate data to a new coordinate system such that the axes of this transformed data maximize the explainable variance.

Linear Method: Principal Component Analysis

- We first derive the first principal component. When $(\vec{x}_i)_{i=1}^n$ are embedded to $(\vec{w}_1^T(\vec{x}_i - \vec{m}_{\vec{X}}))_{i=1}^n$, the variance of transformed data can be expressed as $n^{-1}\sum_{i=1}^n \|\vec{w}_1^T\vec{x}_i - \vec{w}_1^T\vec{m}_{\vec{X}}\|^2 = \vec{w}_1^T S_{\vec{X},\vec{X}}\vec{w}_1$ where $\vec{m}_{\vec{X}} := n^{-1}\sum_{i=1}^n \vec{x}_i$ and $S_{\vec{X},\vec{X}} := n^{-1}\sum_{i=1}^n(\vec{x}_i - \vec{m}_{\vec{X}})(\vec{x}_i - \vec{m}_{\vec{X}})^T$.
- Under a constraint $\vec{w}_1^T\vec{w}_1 = 1$ for scale-invariance, finding the variance-maximizing $\vec{w}_1$ can be formulated as: $\max_{\vec{w}_1:\vec{w}_1^T\vec{w}_1=1} \vec{w}_1^T S_{\vec{X},\vec{X}}\vec{w}_1$.
- Since $\vec{w}_1^T S_1 \vec{w}_1$ is smooth w.r.t. $\vec{w}_1$ and $\{\vec{w}_1 | \vec{w}_1^T\vec{w}_1 = 1\}$ is compact, solutions can be found via the method of Lagrange multipliers. Let

$$L(\vec{w}_1, \beta) := \vec{w}_1^T S_{\vec{X},\vec{X}}\vec{w}_1 + \beta(1 - \vec{w}_1^T\vec{w}_1).$$

The stationary points, potential solutions, satisfy the following conditions:

1. $\partial L/\partial\vec{w}_1 = 2S_{\vec{X},\vec{X}}\vec{w}_1 - 2\beta\vec{w}_1 = 0$.
2. $\partial L/\partial\beta = 1 - \vec{w}_1^T\vec{w}_1 = 0$.

Linear Method: Principal Component Analysis

- The first condition demonstrates that $(\beta, \vec{w}_1)$ corresponds to one of the eigenvalue-eigenvector pairs of the covariance matrix $S_{\vec{X},\vec{X}}$. We denote the pairs by $((\lambda_j, \vec{e}_j))_{j=1}^{p}$ where $\lambda_1 \geq \cdots \geq \lambda_p \geq 0$.
- Next, at each stationary point $(\vec{w}_1, \beta) = (\vec{e}_j, \lambda_j)$, $\vec{w}_1^T S_{\vec{X},\vec{X}} \vec{w}_1 = \lambda_j$ holds. This implies that the maximizer $\vec{w}_1^* = \vec{e}_1$ and the corresponding variance is λ_1.
- Thus, the 1st PC score is $(\vec{e}_1^T \vec{x}_i)_{i=1}^{n}$.

Linear Method: Principal Component Analysis

- Next, given the first r PCs, we derive the $(r+1)$-th PC. The optimization problem can be formulated as: $\max_{\vec{w}_{r+1}:\vec{e}_1^T\vec{w}_{r+1}=0,\ldots,\vec{e}_r^T\vec{w}_{r+1}=0,\vec{w}_{r+1}^T\vec{w}_{r+1}=1} \vec{w}_{r+1}^T S_{\vec{X},\vec{X}}\vec{w}_{r+1}$.
- Let $L := \vec{w}_{r+1}^T S_{\vec{X},\vec{X}}\vec{w}_{r+1} + \beta_{r+1}(1-\vec{w}_{r+1}^T\vec{w}_{r+1}) + \sum_{j=1}^{r}\gamma_{r+1,j}(-\vec{w}_{r+1}^T\vec{e}_j)$. The stationary points satisfy the following conditions:
  1. $\partial L/\partial\vec{w}_{r+1} = 2S_{\vec{X},\vec{X}}\vec{w}_{r+1} - 2\beta_{r+1}\vec{w}_{r+1} - \sum_{j=1}^{r}\gamma_{r+1,j}\vec{e}_j = 0$.
  2. $\partial L/\partial\beta_{r+1} = 1-\vec{w}_{r+1}^T\vec{w}_{r+1} = 0$.
  3. $\partial L/\partial\gamma_{r+1,j} = -\vec{w}_{r+1}^T\vec{e}_j = 0$ for $j < r+1$.

  Multiplying $\vec{e}_{j'}^T$ to the first condition yields $\gamma_{r+1,j} = 0$ for $j < r+1$. Thus, $(\beta_{r+1}, \vec{w}_{r+1})$ is one of eigenvalue-eigenvector pairs of $S_{\vec{X},\vec{X}}$. By the third condition, $\vec{w}_{r+1}$ should be one of $\vec{e}_{r+1}, \ldots, \vec{e}_p$.
- Given $\vec{e}_j^T S_{\vec{X},\vec{X}}\vec{e}_j = \lambda_j$, the optimal $\vec{w}_{r+1}^*$ is the eigenvector corresponding to the $(r+1)$-th eigenvalue.

Linear Method: Principal Component Analysis

- We can reconstruct $\vec{x}$ using loading vectors through a two-step process:

$$\vec{x} \xrightarrow{\text{Embedding}} W^T(\vec{x} - \vec{m}_{\vec{X}}) \xrightarrow{\text{De-embedding}} \vec{m}_{\vec{X}} + W(W^T(\vec{x} - \vec{m}_{\vec{X}})).$$

- Note that the mean squared reconstruction error can be computed as:

$$n^{-1}\sum_{i=1}^{n}\|\vec{x}_i - (\vec{m}_{\vec{X}} + WW^T(\vec{x}_i - \vec{m}_{\vec{X}}))\|^2 = \text{tr}(S_{\vec{X},\vec{X}}) - \sum_{j=1}^{r}\vec{w}_j^T S_{\vec{X},\vec{X}}\vec{w}_j.$$

  Hint: $\text{tr}((I - WW^T)S_{\vec{X},\vec{X}}) = \text{tr}(S_{\vec{X},\vec{X}}) - \text{tr}(W^T S_{\vec{X},\vec{X}} W)$.

- Thus, the PCs are the minimizers of the errors from the above reconstruction process, effectively capturing the most variance of the data while reducing the dimensionality.

Linear Method: Partial Least Squares Regression

- When we have pairs $(\vec{X}, \vec{Y})$ where $\vec{X} \in \mathbb{R}^p$ and $\vec{Y} \in \mathbb{R}^q$, Partial Least Squares (PLS) regression aims to independently rotate $\vec{X}$ and $\vec{Y}$ to find linear transformations that maximize the covariance[1] between their respective dimensions.
- When $(\vec{x}_i)_{i=1}^n$ and $(\vec{y}_i)_{i=1}^n$ are embedded to $(\vec{w}_{\vec{X},1}^T(\vec{x}_i - \vec{m}_{\vec{X}}))_{i=1}^n$ and $(\vec{w}_{\vec{Y},1}^T(\vec{y}_i - \vec{m}_{\vec{Y}}))_{i=1}^n$, respectively, the covariance between transformed data can be expressed as:

$$n^{-1}\sum_{i=1}^{n} < \vec{w}_{\vec{X},1}^T(\vec{x}_i - \vec{m}_{\vec{X}}), \vec{w}_{\vec{Y},1}^T(\vec{y}_i - \vec{m}_{\vec{Y}}) >= \vec{w}_{\vec{X},1}^T S_{\vec{X},\vec{Y}} \vec{w}_{\vec{Y},1} \tag{1}$$

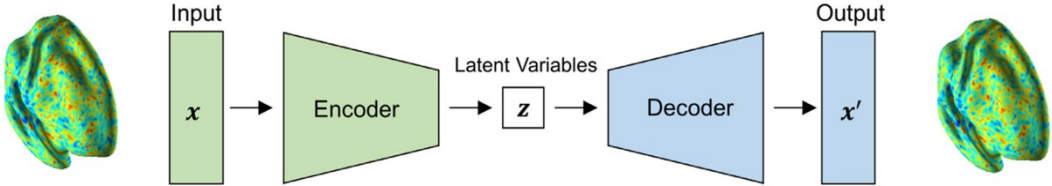
  where $S_{\vec{X},\vec{Y}} := n^{-1}\sum_{i=1}^n (\vec{x}_i - \vec{m}_{\vec{X}})(\vec{y}_i - \vec{m}_{\vec{Y}})^T$.
- Under constraints $\vec{w}_{\vec{X},1}^T \vec{w}_{\vec{X},1} = \vec{w}_{\vec{Y},1}^T \vec{w}_{\vec{Y},1} = 1$, finding the first PLS component can be formulated as: $\max_{(\vec{w}_{\vec{X},1}, \vec{w}_{\vec{Y},1}): \vec{w}_{\vec{X},1}^T \vec{w}_{\vec{X},1} = \vec{w}_{\vec{Y},1}^T \vec{w}_{\vec{Y},1} = 1} \vec{w}_{\vec{X},1}^T S_{\vec{X},\vec{Y}} \vec{w}_{\vec{Y},1}$.

[1] When targeting correlation, the corresponding method is Canonical Correlation Analysis (CCA). We focus on PLS because explaining CCA without assumptions of the invertibility of $S_{\vec{X},\vec{X}}$ and $S_{\vec{Y},\vec{Y}}$ is technically complex.

Linear Method: Partial Least Squares Regression

- Let $L := \vec{w}_{\vec{X},1}^T S_{\vec{X},\vec{Y}} \vec{w}_{\vec{Y},1} + \beta_{\vec{X},1}(1 - \vec{w}_{\vec{X},1}^T \vec{w}_{\vec{X},1})/2 + \beta_{\vec{Y},1}(1 - \vec{w}_{\vec{Y},1}^T \vec{w}_{\vec{Y},1})/2$. The stationary points satisfy the following conditions:
  1. $\partial L/\partial \vec{w}_{\vec{X},1} = S_{\vec{X},\vec{Y}} \vec{w}_{\vec{Y},1} - \beta_{\vec{X},1} \vec{w}_{\vec{X},1} = 0$.
  2. $\partial L/\partial \vec{w}_{\vec{Y},1} = S_{\vec{X},\vec{Y}}^T \vec{w}_{\vec{X},1} - \beta_{\vec{Y},1} \vec{w}_{\vec{Y},1} = 0$.
  3. $\partial L/\partial \vec{\beta}_{\vec{X},1} = (1 - \vec{w}_{\vec{X},1}^T \vec{w}_{\vec{X},1})/2 = 0$.
  4. $\partial L/\partial \vec{\beta}_{\vec{Y},1} = (1 - \vec{w}_{\vec{Y},1}^T \vec{w}_{\vec{Y},1})/2 = 0$.
- These imply that $\vec{w}_{\vec{X},1}^T S_{\vec{X},\vec{Y}} \vec{w}_{\vec{Y},1} = \beta_{\vec{X},1} = \beta_{\vec{Y},1}$. From now on, we simply denote $\beta_1 = \beta_{\vec{X},1} = \beta_{\vec{Y},1}$. This relation and the first two conditions imply $(S_{\vec{X},\vec{Y}} S_{\vec{X},\vec{Y}}^T)\vec{w}_{\vec{X},1} = \beta_1^2 \vec{w}_{\vec{X},1}$ and $(S_{\vec{X},\vec{Y}}^T S_{\vec{X},\vec{Y}})\vec{w}_{\vec{Y},1} = \beta_1^2 \vec{w}_{\vec{Y},1}$.
- Thus, the optimal $(\beta_1^2, \vec{w}_{\vec{X},1}, \vec{w}_{\vec{Y},1})$ is the first singular value of the sample covariance matrix $S_{\vec{X},\vec{Y}}$ and corresponding left- and right-singular vectors.
- As in PCA, the idea can be extended to the first r PLS components.

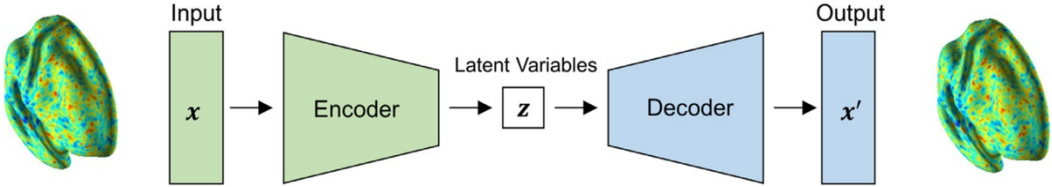
Non-Linear Method: Auto-encoders



- Autoencoders (AEs) (Bengio et al., 2006) consist of pairs of encoders and decoders that efficiently reduce the dimensionality of data.
- Encoders embed observations into a lower-dimensional space (referred to as 'encoding'), while decoders map these encodings back to the original observation space ('decoding' or 'reconstruction').

Images are from Kim et al., 2021.

Non-Linear Method: Auto-encoders



- The prefix 'auto' is used because they autonomously learn to encode data in an unsupervised manner.
- Autoencoders are trained by minimizing the difference between the original observations and their reconstructions, referred to as the *reconstruction error*.

Images are from Kim et al., 2021.

PCA As an Optimal Linear Auto-encoder

- PCA is an optimal linear auto-encoder, i.e., auto-encoders are nonlinear extensions of PCA (Kramer, 1991; Plaut, 2018).
- Let $n^{-1}\sum_{i=1}^{n}\|\vec{x}_i - (\vec{\mu} + W\vec{\lambda}_i)\|^2$ represent the sample mean squared error from the encoding results $\vec{\lambda}_i \in \mathbb{R}^r$ and the decoder $\vec{z} \to \vec{\mu} + W\vec{z}$.
- Then, for any $p \times r$ matrix W s.t. $W^T W = I_r$,

$$n^{-1}\sum_{i=1}^{n}\|\vec{x}_i - (\vec{\mu} + W\vec{\lambda}_i)\|^2 \geq \text{tr}(S_{\vec{X},\vec{X}}) - n^{-1}\sum_{i=1}^{n}\|W^T(\vec{x}_i - \vec{m}_{\vec{X}})\|^2 \tag{2}$$

  where equality holds when $\vec{\mu} = \vec{m}_{\vec{X}}$ and $\vec{\lambda}_i = W^T(\vec{x}_i - \vec{m}_{\vec{X}})$, i.e.,

  $$\text{encoder } \vec{x} \to W^T(\vec{x} - \vec{m}_{\vec{X}}) \text{ and decoder } \vec{z} \to \vec{m}_{\vec{X}} + W\vec{z}$$

  have a mirrored structure using the shared W. They are in an inverse relation when $r = p$.
- Thus, the W that maximizes the explainable variances by linear embeddings of centered data, defining PCs of $(\vec{x}_i - \vec{m}_{\vec{X}})_{i=1}^{n}$, identifies an optimal linear auto-encoder.

Sketch of Proof I

- Let $\mathrm{MSE}(\vec{\mu}, W, (\vec{\lambda}_i)_{i=1}^n) := n^{-1}\sum_{i=1}^n \|\vec{x}_i - (\vec{\mu} + W\vec{\lambda}_i)\|^2$. Utilizing the derivative formula $\frac{\partial}{\partial \vec{x}}(A\vec{x}+\vec{b})(C\vec{x}+\vec{d}) = A^T(C\vec{x}+\vec{d}) + C^T(A\vec{x}+\vec{b})$, we can show that:

$$\frac{\partial^2 \mathrm{MSE}(\vec{\mu}, W, (\vec{\lambda}_i)_{i=1}^n)}{\partial \vec{\mu}\partial \vec{\mu}^T} = 2I_p > O_p. \tag{3}$$

This implies that the solution to $\partial \mathrm{MSE}(\vec{\mu}, W, (\vec{\lambda}_i)_{i=1}^n)/\partial\vec{\mu} = 0$, $\vec{\mu}^*(W, (\vec{\lambda}_i)_{i=1}^n) = \vec{m}_{\vec{x}} - W\vec{m}_{\vec{\lambda}}$ where $\vec{m}_{\vec{\lambda}} := n^{-1}\sum_{i=1}^n \vec{\lambda}_i$, is where minimizers are located.

By substituting $\vec{\mu}^*(W, (\vec{\lambda}_i)_{i=1}^n)$, we have

$$\mathrm{MSE}(\vec{\mu}^*(W, (\vec{\lambda}_i)_{i=1}^n), W, (\vec{\lambda}_i)_{i=1}^n) = n^{-1}\sum_{i=1}^n \|(\vec{x}_i - \vec{m}_{\vec{x}}) - W(\vec{\lambda}_i - \vec{m}_{\vec{\lambda}})\|^2. \tag{4}$$

Since this quantity is invariant to the translation on $\vec{\lambda}$, $\vec{m}_{\vec{\lambda}} = 0$ without loss of generality: $\mathrm{MSE}(\vec{\mu}^*(W, (\vec{\lambda}_i)_{i=1}^n), W, (\vec{\lambda}_i)_{i=1}^n) = n^{-1}\sum_{i=1}^n \|(\vec{x}_i - \vec{m}_{\vec{x}}) - W\vec{\lambda}_i\|^2$.

Sketch of Proof II

- Next,

$$\frac{\partial^2 \mathrm{MSE}(\vec{\mu}^*(W, (\vec{\lambda}_i)_{i=1}^n), W, (\vec{\lambda}_i)_{i=1}^n)}{\partial \vec{\lambda}_i \partial \vec{\lambda}_i^T} = 2n^{-1} W^T W = 2n^{-1} I_r > O_r. \tag{5}$$

This implies that the solution to $\partial\, \mathrm{MSE}(\vec{\mu}^*(W, (\vec{\lambda}_i)_{i=1}^n), W, (\vec{\lambda}_i)_{i=1}^n)/\partial \vec{\lambda}_i = 0$, $\vec{\lambda}_i^*(W) = W^T(\vec{x}_i - \vec{m}_{\vec{X}})$ is where minimizers are located. Thus,

$$\begin{aligned} \mathrm{MSE}(\vec{\mu}, W, (\vec{\lambda}_i)_{i=1}^n)\|^2 &\geq \mathrm{MSE}(\vec{\mu}^*(W, (\vec{\lambda}_i^*(W))_{i=1}^n), W, (\vec{\lambda}_i^*(W))_{i=1}^n)\|^2 \\ &= \mathrm{tr}(S_{\vec{X},\vec{X}}) - n^{-1} \sum_{i=1}^n \|W^T(\vec{x}_i - \vec{m}_{\vec{X}})\|^2. \end{aligned} \tag{6}$$

- See Section 14.5.1. in Hastie (2009) for further details.

PLS As an Optimal Linear Auto-encoder Pair

- PLS also can be interpreted as finding optimal linear autoencoders while posing constraints on the matching embeddings from $\vec{X}$ and $\vec{Y}$.
- Let's consider the following auto-encoder pairs with orthonormal $W_{\vec{X}}$ and $W_{\vec{Y}}$:
  1. Auto-encoder for $\vec{X}$: $\vec{X} \xrightarrow{\text{Encoding}} W_{\vec{X}}^T(\vec{X} - \vec{m}_{\vec{X}}) \xrightarrow{\text{Decoding}} \vec{m}_{\vec{X}} + W_{\vec{X}}(W_{\vec{X}}^T(\vec{X} - \vec{m}_{\vec{X}}))$.
  2. Auto-encoder for $\vec{Y}$: $\vec{Y} \xrightarrow{\text{Encoding}} W_{\vec{Y}}^T(\vec{Y} - \vec{m}_{\vec{Y}}) \xrightarrow{\text{Decoding}} \vec{m}_{\vec{Y}} + W_{\vec{Y}}(W_{\vec{Y}}^T(\vec{Y} - \vec{m}_{\vec{Y}}))$.
- Then, the summation of reconstruction errors and the matching error between embeddings from two AEs can be expressed as:

$$
\begin{aligned}
& n^{-1}\sum_{i=1}^{n}\|\vec{x}_i - \left(\vec{m}_{\vec{X}} + W_{\vec{X}}\left(W_{\vec{X}}^T(\vec{x}_i - \vec{m}_{\vec{X}})\right)\right)\|^2 \\
& + n^{-1}\sum_{i=1}^{n}\|\vec{y}_i - \left(\vec{m}_{\vec{Y}} + W_{\vec{Y}}\left(W_{\vec{Y}}^T(\vec{y}_i - \vec{m}_{\vec{Y}})\right)\right)\|^2 \\
& + n^{-1}\sum_{i=1}^{n}\| W_{\vec{X}}^T(\vec{x}_i - \vec{m}_{\vec{X}}) - W_{\vec{Y}}^T(\vec{y}_i - \vec{m}_{\vec{Y}})\|^2.
\end{aligned}
\tag{7}
$$

PLS As an Optimal Linear Auto-encoder Pair

- Equation (7) is equal to:

$$S_{\vec{X},\vec{X}} + S_{\vec{Y},\vec{Y}} - 2n^{-1}\sum_{i=1}^{n} < W_{\vec{X}}^T\left(\vec{x}_i - \vec{m}_{\vec{X}}\right), W_{\vec{Y}}^T\left(\vec{y}_i - \vec{m}_{\vec{Y}}\right) > \tag{8}$$

Thus, the $W_{\vec{X}}$ and $W_{\vec{Y}}$ that maximize covariances between linear embeddings of centered $\vec{X}$ and $\vec{Y}$, defining PLS, identify an optimal linear AE pairs for Equation (7).

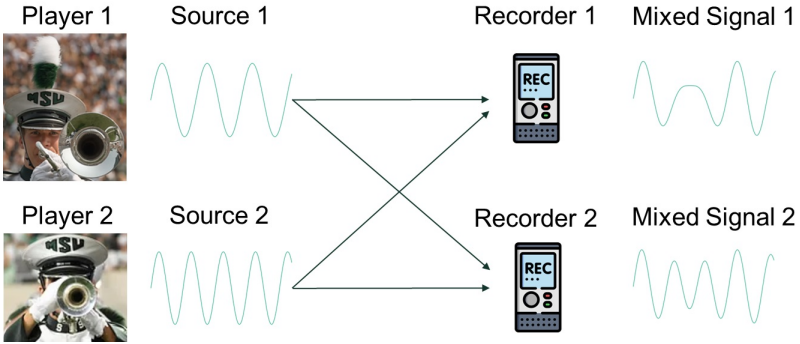
Outline

Motivating Example: Blind Source Separation

- *Independent Component Analysis* (ICA) (Hyvärinen and Oja, 2000) identifies source signals, and its statistical model has inspired recent generative models.
- For a motivating example, consider the following question:
  "How do we recognize specific signals when there are other signals present at the same time?" (Cherry, 1953)

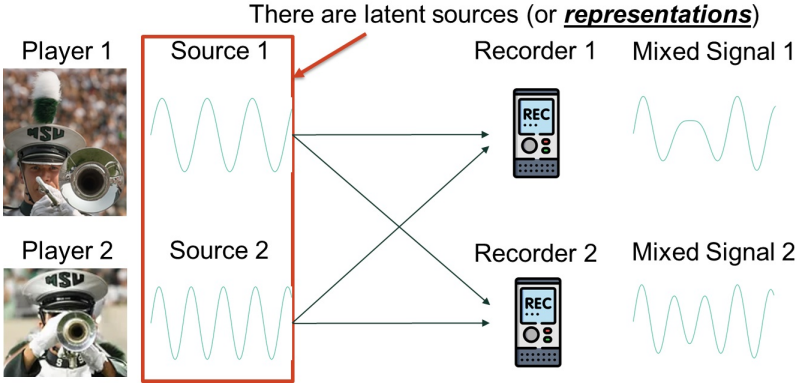
The image is from Flaticon.

Motivating Example: Blind Source Separation



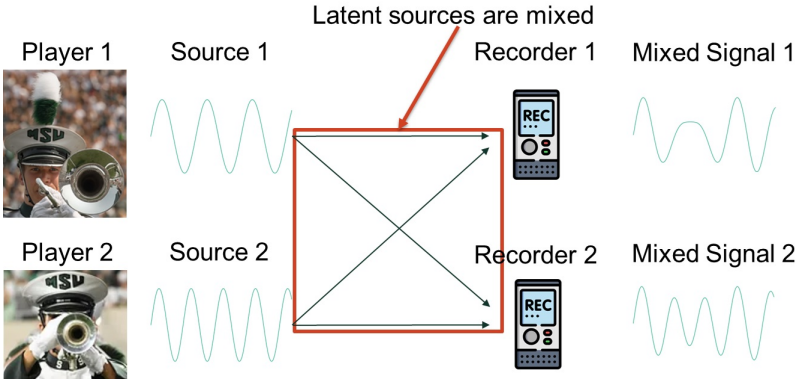
The leftmost images are from *The Spartan Marching Band* website.

Motivating Example: Blind Source Separation



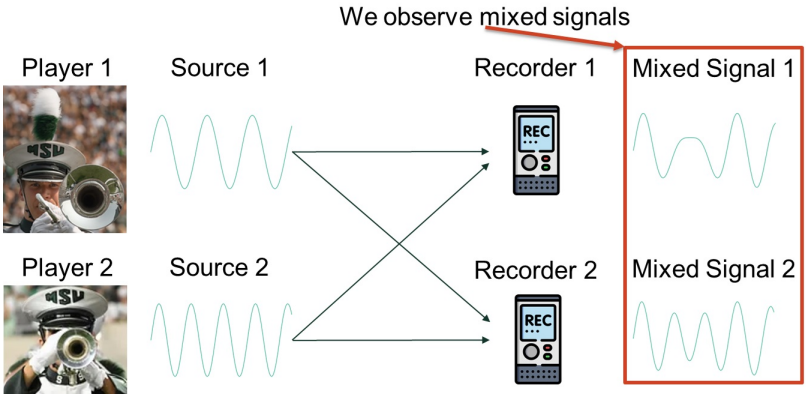
The leftmost images are from *The Spartan Marching Band* website.

Motivating Example: Blind Source Separation



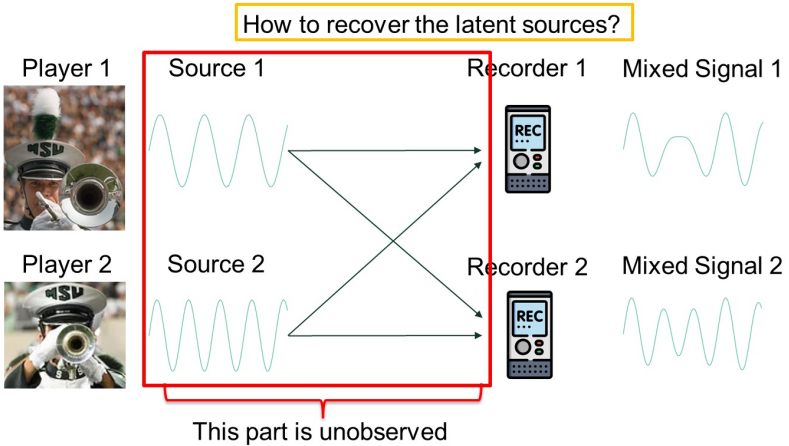
The leftmost images are from *The Spartan Marching Band* website.

Motivating Example: Blind Source Separation



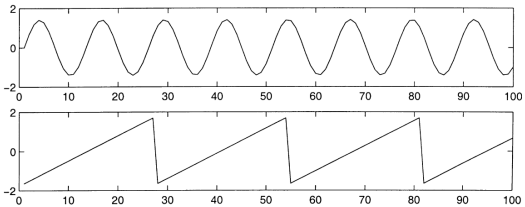
The leftmost images are from *The Spartan Marching Band* website.

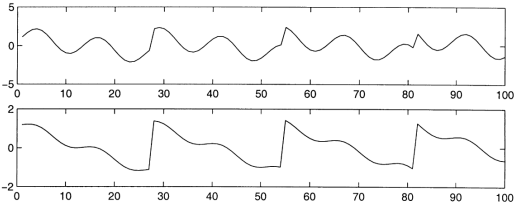
Motivating Example: Blind Source Separation



The leftmost images are from *The Spartan Marching Band* website.

Motivating Example: Blind Source Separation

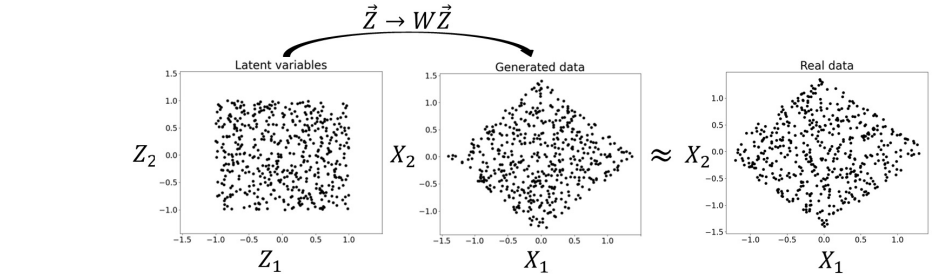




- Let $X_1(t)$ and $X_2(t)$ (right figures) be recorded amplitudes by the two audio meters at a time step t, and $Z_1(t)$ and $Z_2(t)$ (left figures) be signals by the two players.
- When we assume that recorded amplitudes are linear mixtures of signals, this problem can be formulated as $\vec{X}(t) = W\vec{Z}(t)$ where W is a time-invariant invertible 2 by 2 matrix.
- The main interest is to recover signals with observed mixtures of signals. The challenging part is that W is unknown. If W was known, this problem reduces to a linear system.

Figures are from Hyvärinen and Oja (2000).

Independent Component Analysis



- ICA assumes a linear generative structure: $\vec{X} = W\vec{Z}$ where $W^T W = I_r$. Here, $\vec{Z}$ denotes latent variables following a known distribution (e.g., $U([-1,1]^2)$.
- All the sources are assumed to be independent. With W and $\vec{X}$, they can be inferred via $\vec{Z} = W^T\vec{X}$.
- Factor analysis and PCA assume uncorrelated components. In contrast, components from ICA are statistically independent.

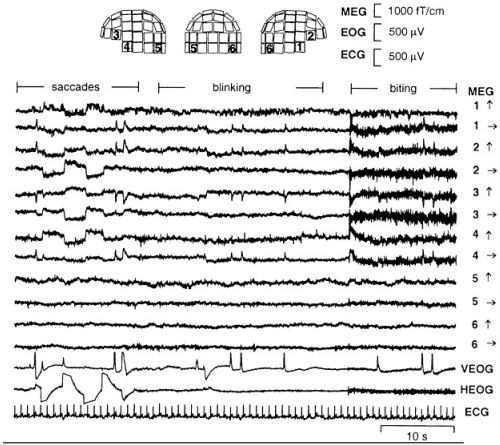
Estimation

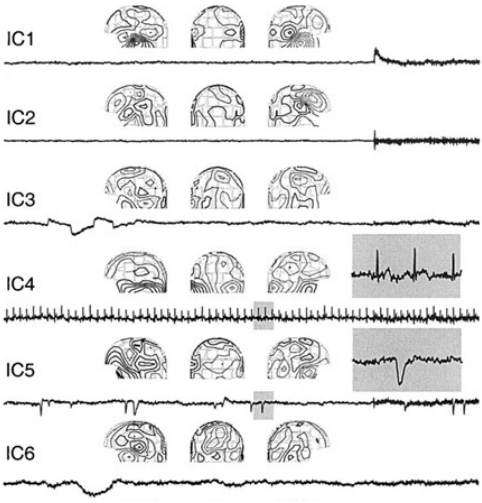
- There are several criteria for learning the linear generation structure, including kurtosis, mutual information, and likelihood.
- Here, we focus on likelihood-maximization approach.[2] The change of variable formula gives $p_\theta(\vec{x}) = p(\vec{z} = W^T\vec{x})|W^T|$, implying:

$$\begin{aligned} l_n(\theta) &= \sum_{i=1}^{n} \log p(\vec{z} = W^T\vec{x}_i) + \log|W^T| \\ &= \sum_{i=1}^{n}\sum_{j=1}^{r} \log p(z_j = (W^T\vec{x}_i)_j) + \log|W^T|. \end{aligned} \tag{9}$$

[2] Although ICA is generally applicable when $r < p$, for simplicity, we assume $r = p$ and $W^T = W^{-1}$.

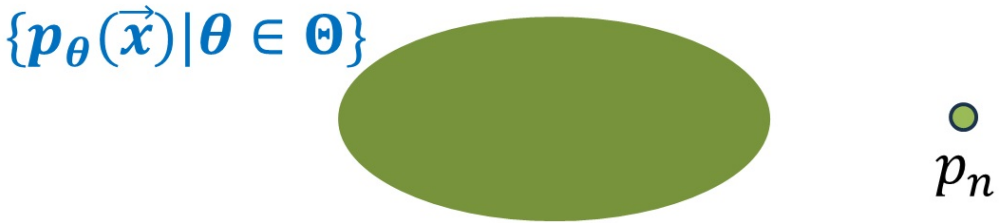
Example Application



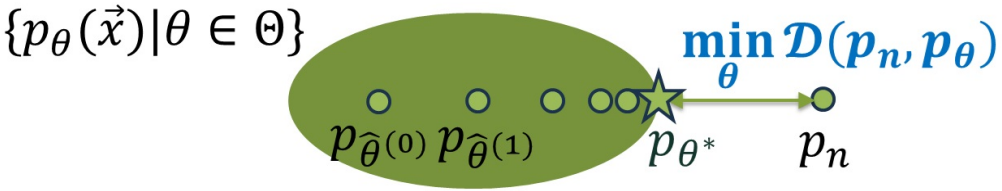


Figures are edited from Hyvärinen and Oja (2000).

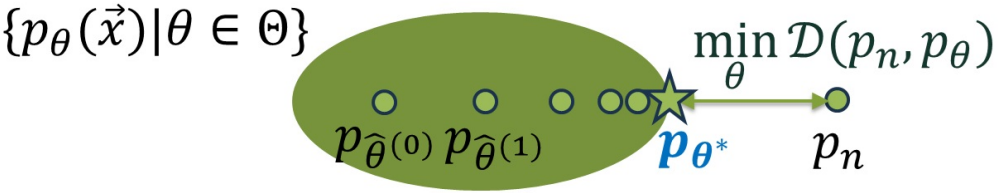
Statistical Reasoning with Independent Component Analysis



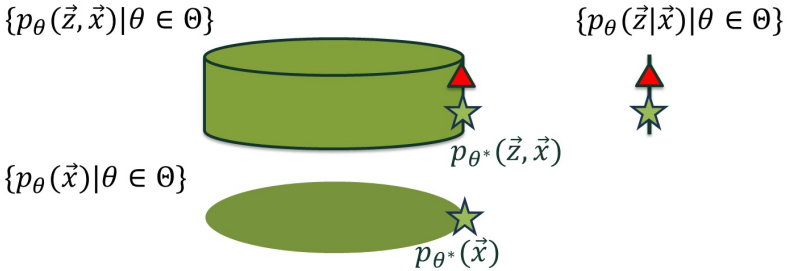
Statistical Reasoning with Independent Component Analysis



Statistical Reasoning with Independent Component Analysis



Identifiability Issue



- Any invertible matrix R satisfies $\vec{X} = W\vec{Z} = (WR^{-1})(R\vec{Z})$. That is, ICA cannot distinguish the generation process with $(W, \vec{Z})$ from $(WR^{-1}, R\vec{Z})$.
- In general, this issue is referred to as the *identifiability issue* of generative models:

$$p_{\theta_1}(\vec{x}) = p_{\theta_2}(\vec{x}) \text{ does not imply } \theta_1 = \theta_2. \tag{10}$$

Consequently, $p_{\theta_1}(\vec{z}|\vec{x}) = p_{\theta_2}(\vec{z}|\vec{x})$ is not guaranteed.

Identifiability Issue

- A critical consequence of the identifiability issue is that, even with $p_{\theta^*}(\vec{x}) \approx p_n(\vec{x})$, $p_{\theta^*}(\vec{z}|\vec{x})$ and $p_n(\vec{z}|\vec{x})$ can differ.
- Representations of ICA are identifiable when we have at least two non-Gaussian components in $\vec{Z}$ (Comon, 1994):[3]

$$p_{\theta_1}(\vec{x}) = p_{\theta_2}(\vec{x}) \text{ implies } p_{\theta_1}(\vec{z}|\vec{x}) = p_{\theta_2}(PD\vec{z}|\vec{x}) \tag{11}$$

  for some permutation and diagonal matrices P and D, respectively. That is, representations can be identified up to permutation and scaling operations.
- Note that when there are two or more Gaussian components in $\vec{Z}$, Equation (11) does not hold.

  Hint: Consider a rotation transformation to the two Gaussian components.
- However, non-linear generative models are not identifiable without further assumptions (Locatello et al., 2019).

[3] See Theorem 11 in Comon (1994) for details.

Outline

Model Class: Recap of the Motivating Example

- Let $\vec{X} := (X_1, \ldots, X_p)^T \in \mathcal{X}^p$ represent a p-dimensional random vector for the 4K-resolution color images where $p \approx 24M$.
- Each color channel value is discrete, ranging from 0 to 255 ($|\mathcal{X}| = 256$).

  Q: How to model $p(X_1 = x_1, \ldots, X_m = x_m)$?

Model Class: Two Extreme Joint Models

1. **Multivariate Categorical Distribution**: Without any domain knowledge, we can introduce parameters for each realization, e.g., $p(X_1 = 0, \ldots, X_p = 0)$.

- It can express all the distributions defined on the data domain. However, the number of parameters is huge, about $|\mathcal{X}|^p$ (approximately 10^{60M}).

2. **Degenerated Model**: When $(X_2, \ldots, X_p)$ are (known) deterministic functions of X_1, introducing parameters for the marginal distribution $p(X_1)$ is sufficient.

- The number of parameters is small, about $|\mathcal{X}|$, and invariant to the data dimension. However, the model class is significantly reduced.

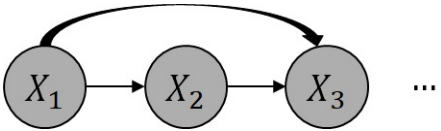
Building appropriate models that reflect domain knowledge is important

Model Class: Bayesian Network

- *Bayesian networks* are probabilistic graphical models that address conditional dependencies using directed acyclic graph structures.
- We can express $p(X_1, \ldots, X_p)$ as a product of conditional distributions:

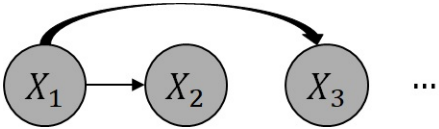
$$p(X_1)p(X_2|X_1)\ldots p(X_p|X_1, X_2, \cdots, X_{p-1}). \tag{12}$$

Many dependency structures can be represented by a directed graph with p nodes $X_1, \ldots, X_p$ and $(p-1)p/2$ directed edges, such as $X_1 \to X_2, \ldots, X_{p-1} \to X_p$.

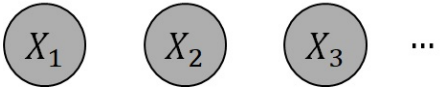


- For example, **1. Multivariate Categorical Distribution** corresponds to the graph using the all nodes and directed edges.

Model Class: Bayesian Network



- **2. Degenerated Model**: This model is a special case of the above graph where $p_\theta(X_1, \ldots, X_p) = p_\theta(X_1)p_\theta(X_2|X_1) \ldots p_\theta(X_p|X_1)$.



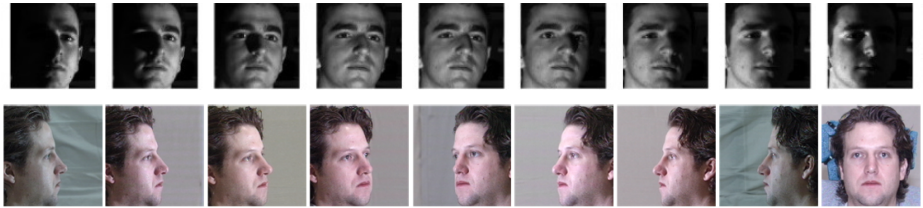
3. **Multivariate Independent Categorical Distribution**: This model assumes the (mutual) independency among variables, using $p_\theta(X_1, \ldots, X_p) = p_\theta(X_1) \ldots p_\theta(X_p)$. It requires $|\mathcal{X}|\, p$ parameters, but its assumption is strong.

Discriminative Model Example: Naive Bayes Classifier

- *Naive Bayes classifier* assumes that $p_\theta(Y, \vec{X}) = p(Y) \prod_{j=1}^{p} p_\theta(X_j|Y)$.
- With this assumption and Bayes' theorem, the logit has a tractable form:

$$\begin{aligned} \log \frac{\mathbb{P}_\theta(Y=1|\vec{X})}{\mathbb{P}_\theta(Y=0|\vec{X})} &= \log \frac{\mathbb{P}_\theta(\vec{X}|Y=1)\mathbb{P}(Y=1)}{\mathbb{P}_\theta(\vec{X}|Y=0)\mathbb{P}(Y=0)} \\ &= \log \frac{\mathbb{P}(Y=1)}{\mathbb{P}(Y=0)} + \sum_{j=1}^{p} \log \frac{\mathbb{P}_\theta(X_j|Y=1)}{\mathbb{P}_\theta(X_j|Y=0)}. \end{aligned} \tag{13}$$

Model Class: Bayesian Network

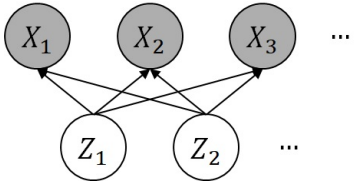


4. **Latent Variable Model**: There are latent factors $\vec{Z} := (Z_1, \ldots, Z_r)^T$, typically consisting of independent components, that are mixed to generate data, e.g., $\vec{X} = W\vec{Z}$ in Independent Component Analysis.

- The age, size, and location of eyes, light source location, and camera angle are examples of latent factors.

The top and bottom images are from the Extended Yale-B (Georghiades et al., 2001) and Multi-pie (Gross et al., 2010) datasets, respectively.

Model Class: Bayesian Network

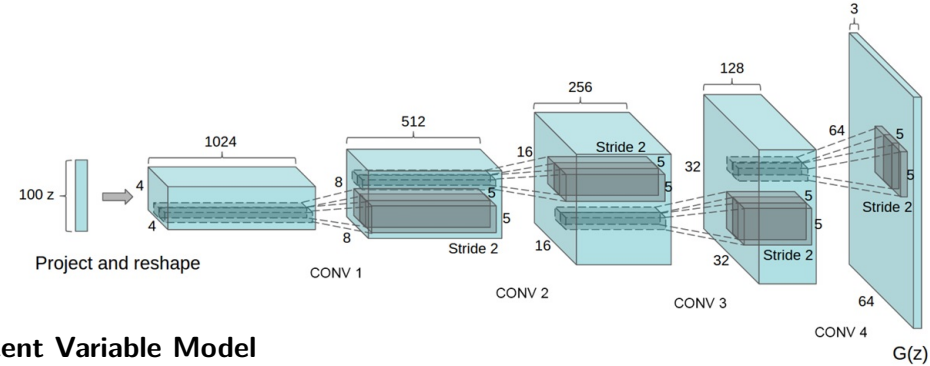


4. Latent Variable Model

$$
\begin{aligned}
p_\theta(X_1, \ldots, X_p) &= \int \Big(p_\theta(X_1, \ldots, X_p | \vec{Z} = \vec{z}) \Big) p(\vec{Z} = \vec{z})\, d\vec{z} \\
&= \int \Big(p_\theta(X_1 | \vec{Z} = \vec{z}) \cdots p_\theta(X_p | \vec{Z} = \vec{z}) \Big) \prod_{i=1}^{r} p(Z_i = z_i)\, d\vec{z}
\end{aligned} \tag{14}
$$

- When we have discrete $\vec{Z}$, the number of parameters is approximately $\Big(|\mathcal{X}| \cdot |\mathcal{Z}|^r \Big) p$.

Model Class: Bayesian Network



4. Latent Variable Model

- Deep generative models are predominantly based on latent variable models.
- The conditional distributions $p_\theta(\vec{X}|\vec{Z})$ are usually modeled as parametric family distributions, e.g., $N(\boldsymbol{\mu}_{\vec{X}|\vec{Z}}(\vec{Z}), \boldsymbol{\Sigma}_{\vec{X}|\vec{Z}}(\vec{Z}))$, which further reduce the number of parameters.

An example neural network for deep generative model from Radford (2015).

Model Class: Cyclic Model

- Some generative models are cyclic graph models.
- **Spatial Model**: The value of the center pixel (e.g., X_{3842}) depends only on adjacent pixels (e.g., $X_1, \ldots, X_{7683}$), making it conditionally independent of all other pixels.
- The number of parameters is about $|\mathcal{X}|^{\#\text{ of adjacent pixels} + 1} \times p$. Assuming translation invariance reduces it to approximately $|\mathcal{X}|^{\#\text{ of adjacent pixels} + 1}$.

Auto-regressive Model

- The *auto-regressive model* is an approach in generative modeling that predicts a subset of explanatory variables using their preceding explanatory variables, e.g., $p(X_3 \mid X_1, X_2)$.
- Again, the joint density of $\vec{X}$ can be expressed as:

$$p(X_1)p(X_2 \mid X_1) \cdots p(X_p \mid X_1, X_2, \ldots, X_{p-1}) \tag{15}$$

- For simplicity, we will assume each feature X_j is binary, e.g., in MNIST images, where 0 and 1 represent black and white pixels, respectively. In this case, we can model $\mathbb{P}(X_j \mid X_1, \ldots, X_{j-1})$ to capture their full joint density.

Auto-regressive Model: Fully Visible Sigmoid Belief Network

- The Fully Visible Sigmoid Belief Network (Neal, 1992)[4] is a logistic auto-regressive model:

$$\mathbb{P}_\theta(X_j = 1 \mid X_1, X_2, \ldots, X_{j-1}) = \sigma(\alpha^{(j)} + \vec{\beta}^{(j)\,T}(X_1, \ldots, X_{j-1})) \tag{16}$$

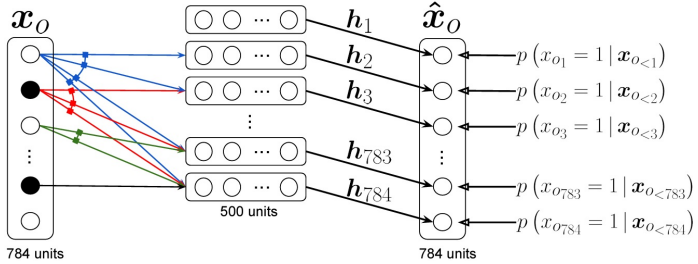
  where $\sigma(x) := 1/(1 + \exp(-x))$ represents the sigmoid function and $\theta := (\alpha^{(1)}, \ldots, \alpha^{(p)}, \vec{\beta}^{(2)}, \ldots, \vec{\beta}^{(p)})^T$.

- The data can be generated in a sequential manner:
  1. $\tilde{x}_1$ is sampled from $\text{Ber}(\sigma(\alpha^{(1)}))$
  2. $\tilde{x}_2$ is sampled from $\text{Ber}(\sigma(\alpha^{(2)} + \beta^{(2)}\tilde{x}_2))$
  3. $\tilde{x}_3$ is sampled from $\text{Ber}(\sigma(\alpha^{(3)} + \vec{\beta}^{(3)\,T}(\tilde{x}_2, \tilde{x}_3))$

  $\vdots$

[4] The name is derived from the Boltzmann machine literature and will be discussed in detail in the next lecture on energy-based models.

Auto-regressive Model: Neural Auto-regressive Density Estimation



- The Neural Auto-regressive Density Estimation (Uria et al., 2016) introduces a neural network:

$$\mathbb{P}_\theta(X_j = 1 \mid X_1 = x_1, \ldots, X_{j-1} = x_{j-1}) = \sigma(\vec{\alpha}_{\vec{H}}^{(j)} + W_{\vec{H}}^{(j)}(h_1^{(j)}, \ldots, h_d^{(j)})) \tag{17}$$

where

$$(h_1^{(j)}, \ldots, h_H^{(j)})^T = \sigma(\vec{\alpha}_{\vec{X}}^{(j)} + W_{\vec{X}}^{(j)}(X_1, \ldots, X_{j-1})^T). \tag{18}$$

The figure is from Uria et al. (2016).

Auto-regressive Model: Neural Auto-regressive Density Estimation

Figure 4: **Left**: samples from NADE trained on binarized MNIST. **Right**: probabilities from which each pixel was sampled. Ancestral sampling was used with the same fixed ordering used during training.

The figure is from Uria et al. (2016).

Auto-regressive Model: Pixel Convolutional Neural Network

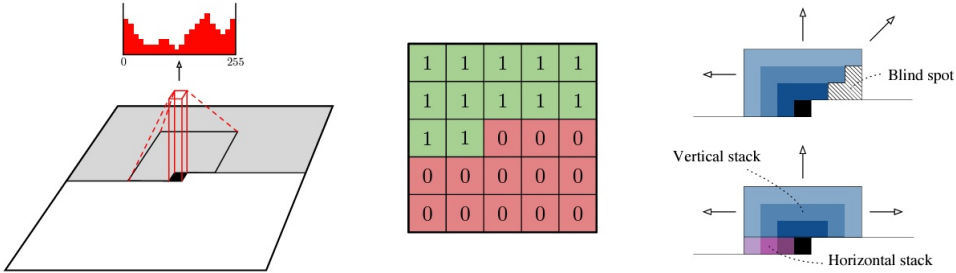


Figure 1: **Left**: A visualization of the PixelCNN that maps a neighborhood of pixels to prediction for the next pixel. To generate pixel x_i the model can only condition on the previously generated pixels $x_1, \ldots x_{i-1}$. **Middle**: an example matrix that is used to mask the 5x5 filters to make sure the model cannot read pixels below (or strictly to the right) of the current pixel to make its predictions. **Right**: Top: PixelCNNs have a *blind spot* in the receptive field that can not be used to make predictions. Bottom: Two convolutional stacks (blue and purple) allow to capture the whole receptive field.

The figure is from Van den Oord et al. (2016).

ICA vs. Auto-regressive Model

- Independent Component Analysis: $\mathbb{P}_\theta(\vec{x}) = \int \delta_{\vec{x}}(W\vec{z}) d\mathbb{P}(\vec{z})$
- Auto-regressive Model: $p_\theta(\vec{x}) = p_\theta(x_1)p_\theta(x_2|x_1)\cdots p_\theta(x_p|x_1,\ldots,x_{p-1})$
  Q: What would be their strong and weak points?

References I

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. (2006). Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19.

Cherry, E. C. (1953). Some experiments on the recognition of speech, with one and with two ears. *The Journal of the acoustical society of America*, 25(5):975–979.

Comon, P. (1994). Independent component analysis, a new concept? *Signal processing*, 36(3):287–314.

Georghiades, A. S., Belhumeur, P. N., and Kriegman, D. J. (2001). From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE transactions on pattern analysis and machine intelligence*, 23(6):643–660.

Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*.

References II

Gross, R., Matthews, I., Cohn, J., Kanade, T., and Baker, S. (2010). Multi-pie. *Image and vision computing*, 28(5):807–813.

Hastie, T. (2009). The elements of statistical learning: data mining, inference, and prediction.

Hyvärinen, A. and Oja, E. (2000). Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430.

Kim, J.-H., Zhang, Y., Han, K., Wen, Z., Choi, M., and Liu, Z. (2021). Representation learning of resting state fmri with variational autoencoder. *NeuroImage*, 241:118423.

Kramer, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2):233–243.

Locatello, F., Bauer, S., Lucic, M., Raetsch, G., Gelly, S., Schölkopf, B., and Bachem, O. (2019). Challenging common assumptions in the unsupervised learning of disentangled representations. In *international conference on machine learning*, pages 4114–4124. PMLR.

References III

Neal, R. M. (1992). Connectionist learning of belief networks. *Artificial intelligence*, 56(1):71–113.

Plaut, E. (2018). From principal subspaces to principal components with linear autoencoders. *arXiv preprint arXiv:1804.10253*.

Radford, A. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.

Uria, B., Côté, M.-A., Gregor, K., Murray, I., and Larochelle, H. (2016). Neural autoregressive distribution estimation. *Journal of Machine Learning Research*, 17(205):1–37.

Van den Oord, A., Kalchbrenner, N., Espeholt, L., Vinyals, O., Graves, A., et al. (2016). Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29.