Unsupervised Learning

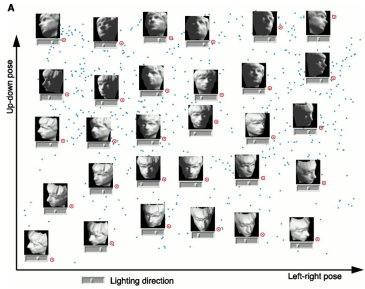
E. Le Pennec

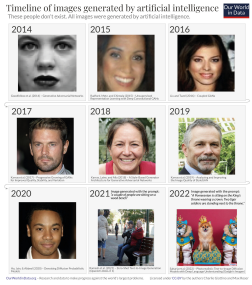
Fall 2023

Outline

Outline

Motivation





What is possible with data without labels?

- To group them?
- To visualize them in a 2 dimensional space?
- To generate more data?

Sources:: Unknown/Tenenbaum et al./World Economic Forum

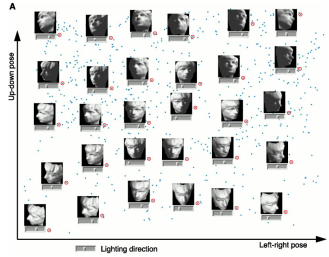
Marketing and Groups

To group them?

- **Data:** Base of customer data containing their properties and past buying records
- **Goal:** Use the customers *similarities* to find groups.
- **Clustering:** propose an explicit *grouping* of the customers
- **Visualization:** propose a representation of the customers so that the groups are *visible*. (Bonus)

Source: Unknown

Image and Visualization

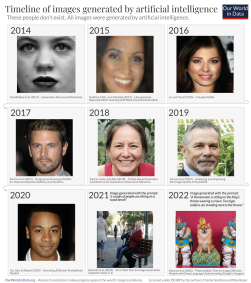


To visualize them?

- **Data:** Images of a single object
- **Goal:** Visualize the *similarities* between images.
- **Visualization:** propose a representation of the images so that similar images are *close*.
- **Clustering:** use this representation to cluster the images. (Bonus)

Source: Tenenbaum et al.

Images and Generation

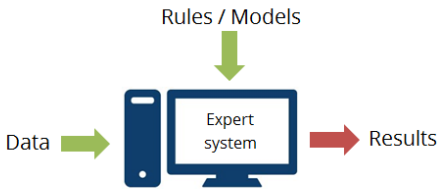


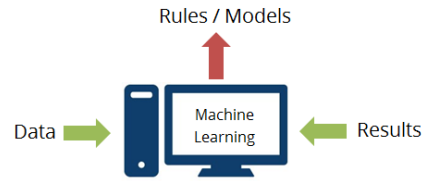
To generate more data?

- **Data:** Images.
- **Goal:** Generate images similar to the ones in the dataset.
- **Generative Modeling:** propose (and train) a generator.

Source: World Economic Forum

Machine Learning





The *classical* definition of Tom Mitchell

A computer program is said to learn from **experience E** with respect to some **class of tasks T** and **performance measure P**, if its performance at tasks in T, as measured by P, improves with experience E.

Source: Council of Europe

Supervised Learning

Experience, Task and Performance measure

- **Training data** : $\mathcal{D} = \{(\underline{X}_1, Y_1), \ldots, (\underline{X}_n, Y_n)\}$ (i.i.d. $\sim \mathbb{P}$)
- **Predictor**: $f : \mathcal{X} \to \mathcal{Y}$ measurable
- **Cost/Loss function**: $\ell(f(\underline{X}), Y)$ measure how well $f(\underline{X})$ *predicts* Y
- **Risk**:

$$\mathcal{R}(f) = \mathbb{E}[\ell(Y, f(\underline{X}))] = \mathbb{E}_X\Big[\mathbb{E}_{Y|\underline{X}}[\ell(Y, f(\underline{X}))]\Big]$$

- Often $\ell(f(\underline{X}), Y) = \|f(\underline{X}) - Y\|^2$ or $\ell(f(\underline{X}), Y) = \mathbf{1}_{Y \neq f(\underline{X})}$

Goal

- Learn a rule to construct a **predictor** $\widehat{f} \in \mathcal{F}$ from the training data $\mathcal{D}_n$ s.t. **the risk** $\mathcal{R}(\widehat{f})$ is **small on average** or with high probability with respect to $\mathcal{D}_n$.

Unsupervised Learning

Experience, Task and Performance measure

- **Training data** : $\mathcal{D} = \{\underline{X}_1, \ldots, \underline{X}_n\}$ (i.i.d. $\sim \mathbb{P}$)
- **Task**: ???
- **Performance measure**: ???

- No obvious task definition!

Tasks for this lecture

- **Dimension reduction:** construct a map of the data in a **low dimensional** space without **distorting** it too much.
- **Clustering (or unsupervised classification):** construct a **grouping** of the data in **homogeneous** classes.
- **Generative modeling: generate** new conditional samples.

Dimension Reduction

- **Training data** : $\mathcal{D} = \{\underline{X}_1, \dots, \underline{X}_n\} \in \mathcal{X}^n \quad (\text{i.i.d. } \sim \mathbb{P})$
- Space $\mathcal{X}$ of possibly high dimension.

Dimension Reduction Map

- Construct a map Φ from the space $\mathcal{X}$ into a space $\mathcal{X}'$ of **smaller dimension**:

$$\begin{aligned} \Phi : \quad & \mathcal{X} \to \mathcal{X}' \\ & \underline{X} \mapsto \Phi(\underline{X}) \end{aligned}$$

- Map can be defined only on the dataset.

Motivations

- Visualization of the data
- Dimension reduction (or embedding) before further processing

Dimension Reduction

- Need to control the **distortion** between $\mathcal{D}$ and $\Phi(\mathcal{D}) = \{\Phi(\underline{X}_1), \ldots, \Phi(\underline{X}_n)\}$

Distortion(s)

- Reconstruction error:
  - Construct $\widetilde{\Phi}$ from $\mathcal{X}'$ to $\mathcal{X}$
  - Control the error between $\underline{X}$ and its reconstruction $\widetilde{\Phi}(\Phi(\underline{X}))$
- Relationship preservation:
  - Compute a *relation* $\underline{X}_i$ and $\underline{X}_j$ and a *relation* between $\Phi(\underline{X}_i)$ and $\Phi(\underline{X}_j)$
  - Control the difference between those two *relations*.

- Lead to different constructions. . . .

Clustering

- **Training data** : $\mathcal{D} = \{\underline{X}_1, \ldots, \underline{X}_n\} \in \mathcal{X}^n \quad (\text{i.i.d.} \sim \mathbb{P})$
- Latent groups?

Clustering

- Construct a map f from $\mathcal{D}$ to $\{1, \ldots, K\}$ where K is a number of classes to be fixed:

$$f : \quad \underline{X}_i \mapsto k_i$$

- Similar to classification except:
  - no ground truth (no given labels)
  - label only elements of the dataset!

Motivations

- Interpretation of the groups
- Use of the groups in further processing

Clustering

- Need to define the **quality** of the cluster.
- No obvious measure!

Clustering quality

- Inner homogeneity: samples in the same group should be similar.
- Outer inhomogeneity: samples in two different groups should be different.

- Several possible definitions of similar and different.
- Often based on the distance between the samples.
- Example based on the Euclidean distance:
  - Inner homogeneity = intra-class variance,
  - Outer inhomogeneity = inter-class variance.
- **Beware:** choice of the number of clusters K often complex!

Generative Modeling

- **Training data** : $\mathcal{D} = \{(\underline{X}_1, \underline{Y}_1), \ldots, (\underline{X}_n, \underline{Y}_n)\} \in (\mathcal{X} \times \mathcal{Y})^n \quad (\text{i.i.d. } \sim \mathbb{P})$
- Same kind of data than for supervised learning if $\mathcal{Y} \neq \emptyset$.

Generative Modeling

- Construct a map G from the product of $\mathcal{Y}$ and a randomness source Ω to $\mathcal{X}$

$$G : \mathcal{Y} \times \Omega \to \mathcal{X}$$
$$(Y, \omega) \mapsto X$$

- Unconditional model if $\mathcal{Y} = \emptyset \ldots$

Motivation

- Generate plausible novel conditional samples based on a given dataset.

Sample Quality

- Related to the proximity between the law of $G(Y, \omega)$ and the law of $X|Y$.
- Most classical choice is the Kullback-Leibler divergence.

Generative Modeling

Ingredients

- Generator $G_\theta(Y, \omega)$ and cond. density prob. $P_\theta(X|Y)$ (Explicit vs implicit link)
- Simple / Complex / Approximate estimation...

Some Possible Choices

	Probabilistic model	Generator	Estimation
Base	Simple (parametric)	Explicit	Simple (ML)
Flow	Image of simple model	Explicit	Simple (ML)
Factorization	Factorization of simple model	Explicit	Simple (ML)
VAE	Simple model with latent var.	Explicit	Approximate (ML)
EBM	Arbitrary	Implicit (MCMC)	Complex (ML/score/discrim.)
Diffusion	Continuous noise	Implicit (MCMC)	Complex (score)
	Discrete Noise with latent var.	Explicit	Approximate (ML)
GAN	Implicit	Explicit	Complex (Discrimination)

- SOTA: Diffusion based approach!

ML: Maximum Likelihood/VAE: Variational AutoEncoder/EBM: Energy Based Model/MCMC: Monte Carlo Markov Chain/GAN: Generative Aversarial Network

Bonus Task: Representation Learning

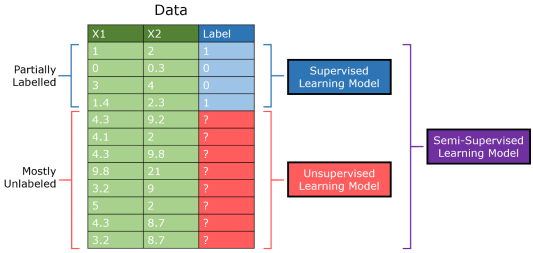
- **General observation:** most data do not have a label !
- **Example:** The number of images on which someone has described the content of the image is a *tiny fraction* of the images online.
- Labeling is very expensive and time consuming
- A lot of information can be extracted from the structure of the data, before seeing any label.

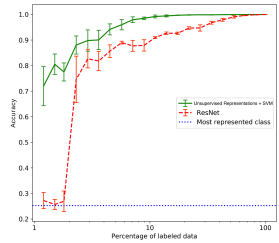
How can we leverage the large quantity of un-labeled data?

- Learn relevant features (= **representations**) in an unsupervised fashion
- Use those features to solve a supervised task with a fraction of labeled data.

- **Semi-supervised framework**
- ↬ Very useful in practice, for images, time series, text.

Semi-supervised Framework





Semu-Supervised Framework

- With representation learned in an unsupervised fashion + a simple linear model, one can achieve the same performance with 10% of data labeled than with a fully annotated dataset.

- Complementary regularization based approaches also exist.

Source: Andre Ye, J.Y. Franceschi et al.

Unsupervised Learning is a Versatile Approach!

Except for generative modeling, the learner is always right

- A subjective measure of performance
- Subjective choices for the algorithmic constraints (e.g., the type of transformation of the data we allow for low-dimensional representation, type of groups in clustering)
- $\Rightarrow$ Very difficult or impossible to tell which is the *best* method.

- Yet:
  - Extremely important in practice:
    - 90-99% of the data is un-labeled!
    - the tasks themselves are fundamental
  - Huge success in various fields (NLP, images...)

Unsupervised Learning is a Versatile Approach!

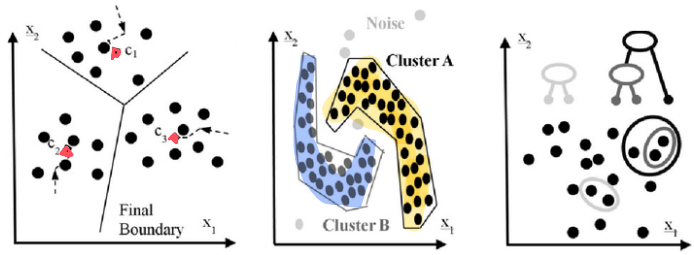
Lecture goals for the three main tasks

- Discussing possible choices of measures of performance and algorithmic constraints
- Understand the correspondences between those choices and a variety of classical algorithms
- For the simplest algorithms (PCA, k-means), get a precise mathematical understanding of the learning process.

Outline

Outline

What's a group?



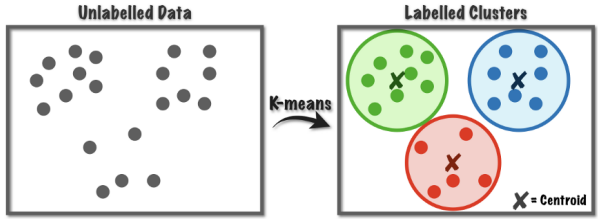
- No simple or unanimous definition!
- Require a notion of similarity/difference...

Three main approaches

- A group is a set of samples similar to a prototype.
- A group is a set of samples that can be linked by contiguity.
- A group can be obtained by fusing some smaller groups...

Source: J. Pitel

Prototype Approach



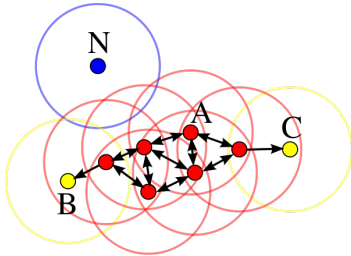
Prototype Approach

- A group is a set of samples similar to a prototype.
- Most classical instance: k-means algorithm.
- Principle: alternate prototype choice for the current groups and group update based on those prototypes.

- Number of groups fixed at the beginning
- No need to compare the samples between them!

Source: A. Jeffares

Contiguity Approach



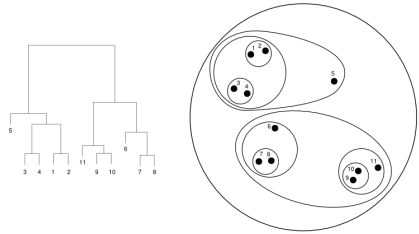
Contiguity Approach

- A group is the set of samples that can be linked by contiguity.
- Most classical instance: DBScan
- Principle: group samples by contiguity if possible (proximity and density)

- Some samples may remain isolated.
- Number of groups controlled by the scale parameter.

Source: Wikipedia

Agglomerative Approach



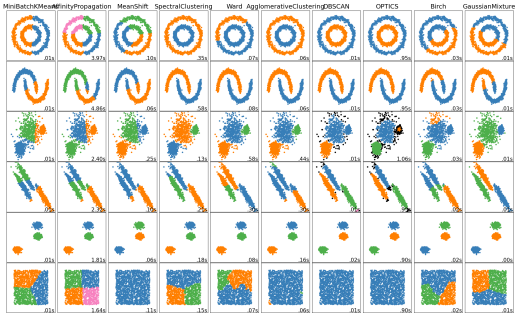
Agglomerative Approach

- A group can be obtained by fusing some smaller groups...
- Hierachical clustering principle: sequential merging of groups according to a *best merge* criterion

- Numerous variations on the merging criterion...
- Number of groups chosen afterward.

Source: upGrad

Choice of the method and of the number of groups



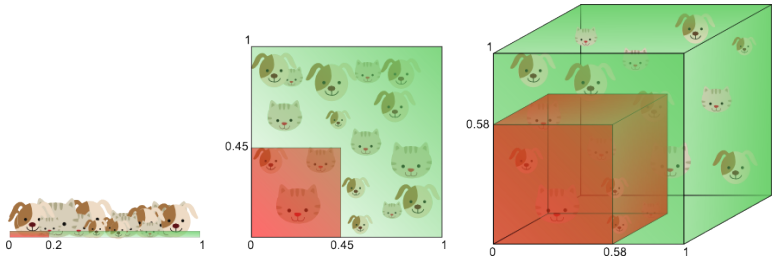
No method or number of groups is better than the others. . .

- Criterion not necessarily explicit!
- No cross validation possible
- Choice of the number of groups: a priori, heuristic, *based on the final usage. . .*

Source: Scikit-Learn

Outline

Dimensionality Curse



- **DISCLAIMER: Even if they are used everywhere beware of the usual distances in high dimension!**

Dimensionality Curse

- Previous approaches based on distances.
- Surprising behavior in high dimension: everything is ((often) as) far away.
- Beware of categories...

Source: Vision Dummy

Dimensionality Curse

- **DISCLAIMER: Even if they are used everywhere beware of the usual distances in high dimension!**

High Dimensional Geometry Curse

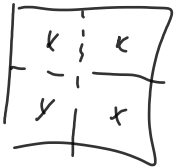
- Folks theorem: In high dimension, everyone is alone.
- Theorem: If $\underline{X}_1, \dots, \underline{X}_n$ in the hypercube of dimension d such that their coordinates are i.i.d then

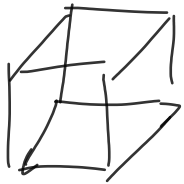
$$d^{-1/p}\left(\max \|\underline{X}_i - \underline{X}_j\|_p - \min \|\underline{X}_i - \underline{X}_j\|_p\right) = 0 + O_P\left(\sqrt{\frac{\log n}{d}}\right)$$

$$\frac{\min \|\underline{X}_i - \underline{X}_j\|_p}{\max \|\underline{X}_i - \underline{X}_j\|_p} = 1 + O_P\left(\sqrt{\frac{\log n}{d}}\right).$$

- When d is large, all the points are almost equidistant...
- Nearest neighbors are meaningless!

$n \leq 2^d$





d	1	2	3	d
n	2	4	8	2^d

Outline

Visualization and Dimension Reduction

- How to view a dataset in high dimension !
- High dimension: dimension larger than 2!
- **Projection** onto a 2D space.

Source: F. Belardi

Visualization and Dimension Reduction

Visualization and Dimension Reduction

- How to view a dataset in high dimension !
- High dimension: dimension larger than 2!
- **Projection** onto a 2D space.

Source: F. Belardi

Visualization and Dimension Reduction

Visualization and Dimension Reduction

- How to view a dataset in high dimension !
- High dimension: dimension larger than 2!
- **Projection** onto a 2D space.

Source: F. Belardi

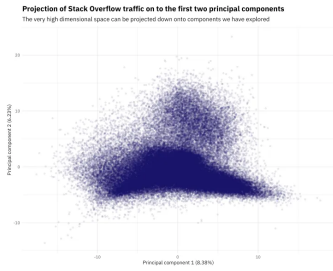
Visualization and Dimension Reduction

Visualization and Dimension Reduction

- How to view a dataset in high dimension !
- High dimension: dimension larger than 2!
- **Projection** onto a 2D space.

Source: F. Belardi

Principal Component Analysis



projection orthogonale

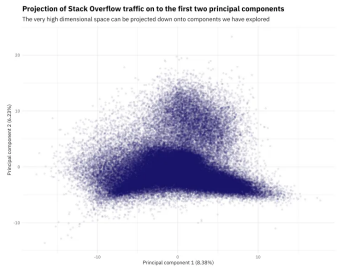
- Simple formula: $\tilde{X} = P(X - m)$

How to chose P?

- Maximising the dispersion of the points?
- Allowing to well reconstruct X from $\tilde{X}$?
- Preserving the relationship between the X through those between the $\tilde{X}$?

Source: J. Silge

Principal Component Analysis



- Simple formula: $\tilde{X} = P(X - m)$

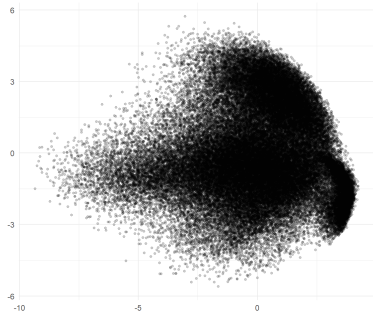
How to chose P?

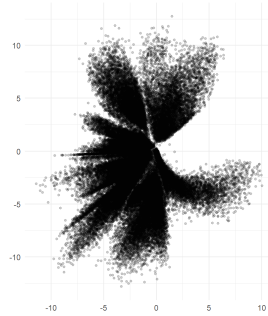
- Maximising the dispersion of the points?
- Allowing to well reconstruct X from $\tilde{X}$?
- Preserving the relationship between the X through those between the $\tilde{X}$?

- The 3 approaches yield the same solution!

Source: J. Silge

Reconstruction Approaches



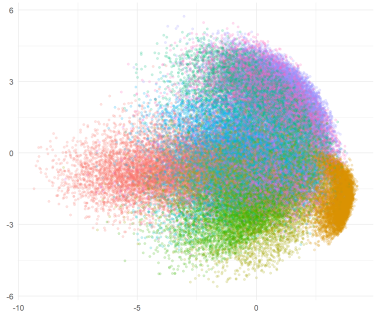


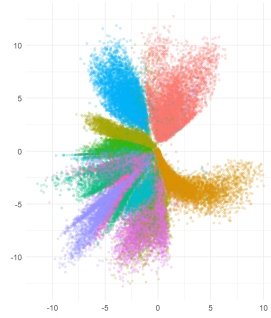
Reconstruction Approaches

- Learn a formula to encode and one formula to decode.
- Auto-encoder structure

- Yields a formula for new points.

Reconstruction Approaches



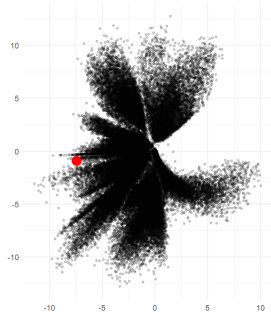


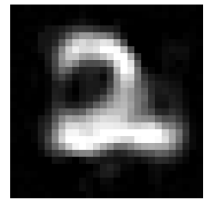
Reconstruction Approaches

- Learn a formula to encode and one formula to decode.
- Auto-encoder structure

- Yields a formula for new points.

Reconstruction Approaches



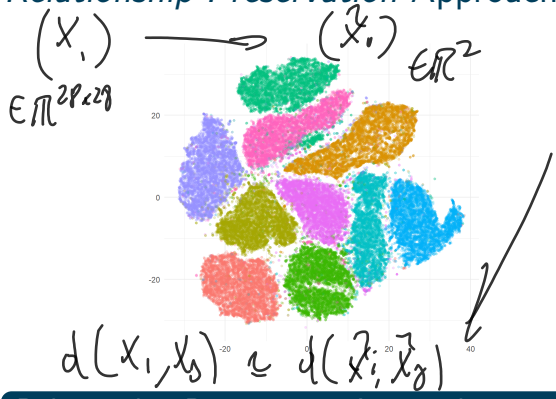


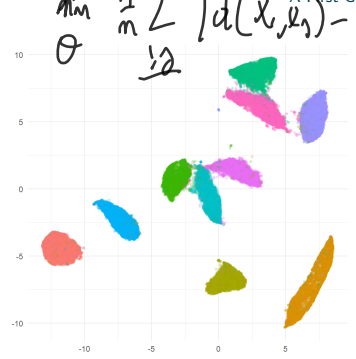
Reconstruction Approaches

- Learn a formula to encode and one formula to decode.
- Auto-encoder structure

- Yields a formula for new points.

Relationship Preservation Approaches



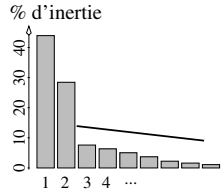


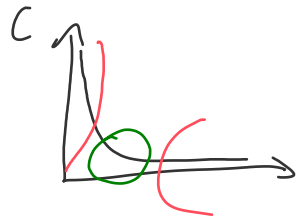
Relationship Preservation Approaches

- Based on the definition of the relationship notion (in both worlds).
- Huge flexibility

- Not always yields a formula for new points.

Choices of Methods and Dimension





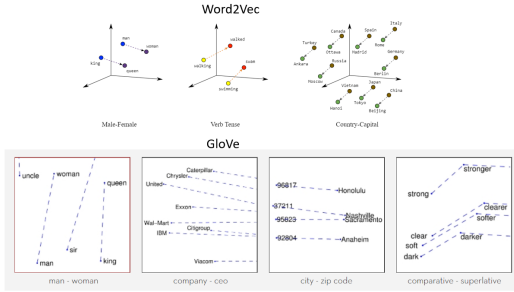
No Better Choice?

- Different criterion for different methods: impossible to use cross-validation.
- The larger the dimension the easier is to be faithful!
- In visualization, dimension 2 is the only choice.
- Heuristic criterion for the dimension choice: elbow criterion (no more gain), stability...

- Dimension Reduction is rarely used standalone but rather as a step in a predictive/prescriptive method.
- The dimension becomes an hyper-parameter of this method.

Source: E. Matzner-Løber

Representation Learning



Representation Learning

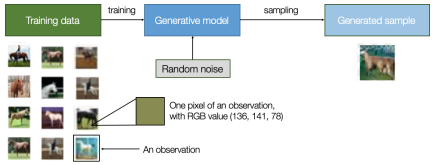
- How to transform arbitrary objects into numerical vectors?
- Objects: Categorical variables, Words, Images/Sounds...

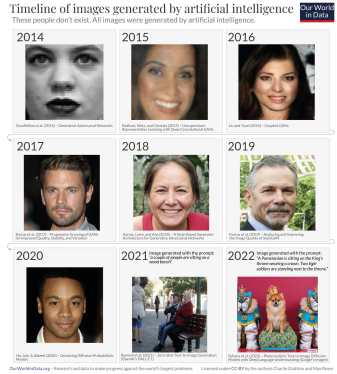
- The two previous dimension reduction approaches can be used (given possibly a first simple high dimensional representation)

Source: R. Khandelwal

Outline

Generative Modeling





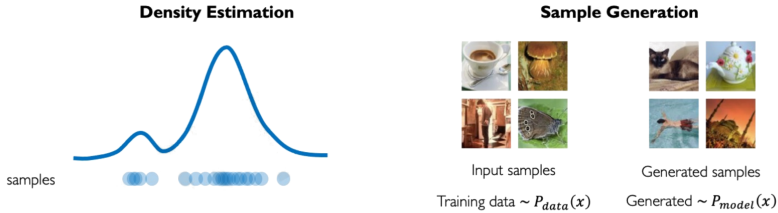
Generative Modeling

- Generate new samples similar to the ones in an original dataset.
- Generation may be conditioned by a input.

- Key for image generation. . . and chatbot!

Sources:: D. Foster/World Economic Forum

Density Estimation and Simulation



How can we learn $P_{model}(x)$ similar to $P_{data}(x)$?

- **Heuristic:** If we can estimate the (conditional) law $\mathbb{P}$ of the data and can simulate it, we can obtain new samples similar to the input ones.

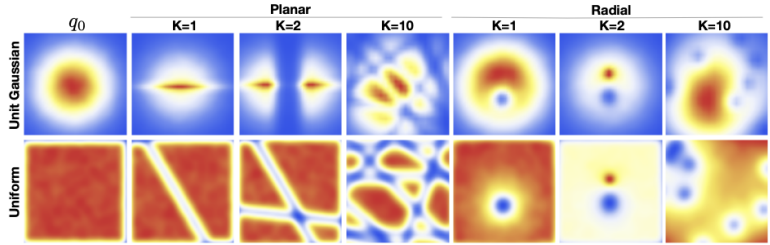
Estimation and Simulation

- How to estimate the density?
- How to simulate the estimate density?

- Other possibilities?

Source: Pipe Galera

Simple Estimation and Simple Simulation



Parametric Model, Image and Factorization

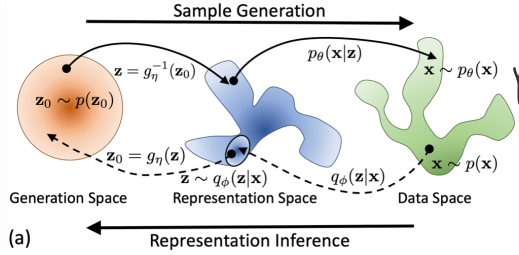
- Use
  - a simple parametric model,...
  - or the image of a parametric model (flow),...
  - or a factorization of a parametric model (recurrent model)

  as they are *simple* to estimate and to simulate.

- Estimation by Maximum Likelihood principle.
- Recurrent models are used in Large Language Models!

Source: Rezende et al.

Complex Estimation and Simple Simulation



X

$P(X,Z) = P(Z)$

$P(X|Z)$

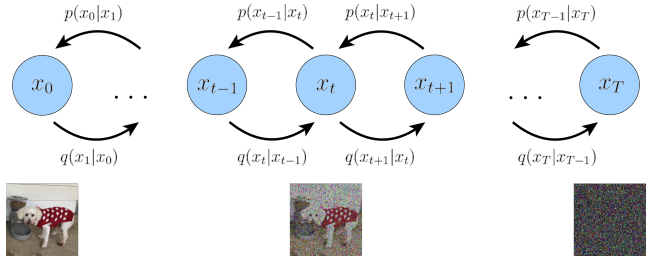
Latent Variable

- Generate first a (low dimensional) latent variable Z from which X is easy to sample.
- Estimation based on approximate Maximum Likelihood (VAE/ELBO)

- The latent variable can be generated by a simple method (or a more complex one. . .).

Source: S. Elhabian

Complex Estimation and Complex Simulation

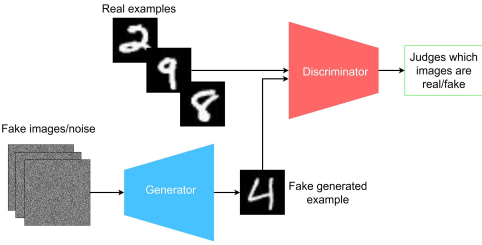


Monte Carlo Markov Chain

- Rely on much more complex probability model...
- which can only be simulated numerically.
- Often combined with noise injection to stabilizes the numerical scheme (Diffusion).

- Much more expensive to simulate than with Latent Variable approaches.

Source: Calvin Huo



Generative Adversarial Network

- Bypass the density estimation problem, by transforming the problem into a competition between the generator and a discriminator.
- The better the generator, the harder it is for the generator to distinguish true samples from synthetic ones.
- No explicit density!

- Fast simulator but unstable training. . .

Source: IBM

Outline

Dimension Reduction

- **Training data** : $\mathcal{D} = \{\underline{X}_1, \ldots, \underline{X}_n\} \in \mathcal{X}^n \quad (\text{i.i.d. } \sim \mathbb{P})$
- Space $\mathcal{X}$ of possibly high dimension.

Dimension Reduction Map

- Construct a map Φ from the space $\mathcal{X}$ into a space $\mathcal{X}'$ of **smaller dimension**:

$$\begin{aligned} \Phi : \quad & \mathcal{X} \to \mathcal{X}' \\ & \underline{X} \mapsto \Phi(\underline{X}) \end{aligned}$$

Criterion

- Reconstruction error
- Relationship preservation

Outline

How to Simplify?

A Projection Based Approach

- Observations: $\underline{X}_1, \ldots, \underline{X}_n \in \mathbf{R}^d$
- Simplified version: $\Phi(\underline{X}_1), \ldots, \Phi(\underline{X}_n) \in \mathbf{R}^d$ with Φ an affine projection preserving the mean $\Phi(\underline{X}) = P(\underline{X} - m) + m$ with $P^\top = P = P^2$ and $m = \frac{1}{n}\sum_i \underline{X}_i$.

How to choose P?

- **Inertia criterion**:
$$\max_P \sum_{i,j} \|\Phi(\underline{X}_i) - \Phi(\underline{X}_j)\|^2 ?$$
- **Reconstruction criterion**:
$$\min_P \sum_i \|\underline{X}_i - \Phi(\underline{X}_i)\|^2 ?$$
- **Relationship criterion**:
$$\min_P \sum_{i,j} |(\underline{X}_i - m)^\top (\underline{X}_j - m) - (\Phi(\underline{X}_i) - m)^\top (\Phi(\underline{X}_j) - m)|^2 ?$$

- **Rk:** Best solution is $P = I$! Need to reduce the rank of the projection to $d' < d \ldots$

Inertia criterion

- **Heuristic:** a good representation is such that the projected points are far apart.

Two views on inertia

- Inertia:

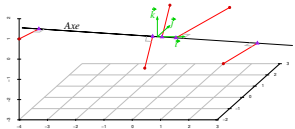
$$I = \frac{1}{2n^2} \sum_{i,j} \|\underline{X}_i - \underline{X}_j\|^2 = \frac{1}{n} \sum_{i=1}^{n} \|\underline{X}_i - m\|^2$$

- 2 times the mean squared distance to the mean = Mean squared distance between individual

Inertia criterion (Principal Component Analysis)

- Criterion: $\max_P \sum_{i,j} \frac{1}{2n^2} \|P\underline{X}_i - P\underline{X}_j\|^2 = \max_P \frac{1}{n} \sum_i \|P\underline{X}_i - m\|^2$
- **Solution:** Choose P as a projection matrix on the space spanned by the d' first eigenvectors of $\Sigma = \frac{1}{n} \sum_i (\underline{X}_i - m)(\underline{X}_i - m)^\top$

First Component of the PCA



- $\widetilde{\underline{X}} = m + a^\top(\underline{X} - m)a$ with $\|a\| = 1$
- Inertia: $\frac{1}{n}\sum_{i=1}^{n} a^\top(\underline{X}_i - m)(\underline{X}_i - m)^\top a$

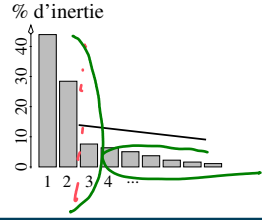
Principal Component Analysis: optimization of the projection

- Maximization of $\widetilde{I} = \frac{1}{n}\sum_{i=1}^{n} a^\top(\underline{X}_i - m)(\underline{X}_i - m)^\top a = a^\top \Sigma a$ with

  $\Sigma = \frac{1}{n}\sum_{i=1}^{n}(\underline{X}_i - m)(\underline{X}_i - m)^\top$ the empirical covariance matrix.
- Explicit optimal choice given by the eigenvector of the largest eigenvalue of Σ.

Source: E. Matzner-Løber

PCA



Principal Component Analysis : sequential optimization of the projection

- Explicit optimal solution obtain by the projection on the eigenvectors of the largest eigenvalues of Σ.
- Projected inertia given by the sum of those eigenvalues.

- Often fast decay of the eigenvalues: some dimensions are much more important than others.
- Not exactly the curse of dimensionality setting...
- Yet a lot of *small* dimension can drive the distance!

Source: E. Matzner-Løber

Reconstruction Criterion

- **Heuristic:** a good representation is such that the projected points are close to the original ones.

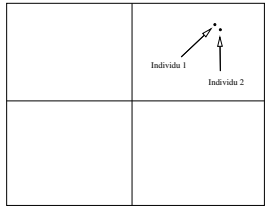
Reconstruction Criterion

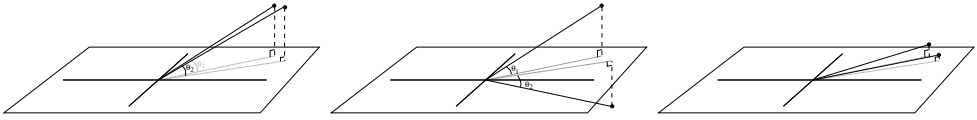
- Criterion: $\min_P \sum_i \frac{1}{n} \|\underline{X}_i - (P(\underline{X}_i - m) + m)\|^2 = \min_P \frac{1}{n} \sum_i \|(I - P)(\underline{X}_i - m)\|^2$
- **Solution:** Choose P as a projection matrix on the space spanned by the d' first eigenvectors of $\Sigma = \frac{1}{n} \sum_i (\underline{X}_i - m)(\underline{X}_i - m)^\top$

- Same solution with a different heuristic!
- Proof (Pythagora):

$$\sum_i \|\underline{X}_i - m\|^2 = \sum_i \left(\|P(\underline{X}_i - m)\|^2 + \|(I - P)(\underline{X}_i - m)\|^2 \right)$$

PCA, Reconstruction and Distances





Close projection doesn't mean close individuals!

- Same projections but different situations.
- Quality of the reconstruction measured by the angle with the projection space!

Source: E. Matzner-Løber

Relationship Criterion

- **Heuristic:** a good representation is such that the projected points scalar products are similar to the original ones.

Relationship Criterion (Multi Dimensional Scaling)

- Criterion: $\min_P \sum_{i,j} |(\underline{X}_i - m)^\top (\underline{X}_j - m) - (\Phi(\underline{X}_i) - m)^\top (\Phi(\underline{X}_j) - m)|^2$
- **Solution:** Choose P as a projection matrix on the space spanned by the d' first eigenvectors of $\Sigma = \frac{1}{n} \sum_i (\underline{X}_i - m)(\underline{X}_i - m)^\top$

- Same solution with a different heuristic!
- Much more involved justification!

Link with SVD

- PCA model: $\underline{X} - m \simeq P(\underline{X} - m)$
- **Prop:** $P = VV^\top$ with V an orthormal family in dimension d of size d'.
- PCA model with V: $\underline{X} - m \simeq VV^\top(\underline{X} - m)$ where $\tilde{\underline{X}} = V^\top(\underline{X} - m) \in \mathbb{R}^{d'}$
- Row vector rewriting: $\underline{X}^\top - m^\top \simeq \tilde{\underline{X}}^\top V^\top$

Matrix Rewriting and Low Rank Factorization

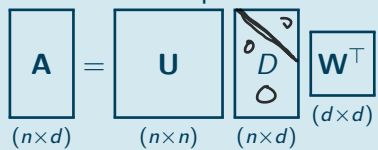
- Matrix rewriting

$$\begin{bmatrix} \underline{X}_1^\top - m^\top \\ \vdots \\ \vdots \\ \underline{X}_n^\top - m^\top \end{bmatrix}_{(n\times d)} \simeq \begin{bmatrix} \tilde{\underline{X}}_1^\top \\ \vdots \\ \vdots \\ \tilde{\underline{X}}_n^\top \end{bmatrix}_{(n\times d')} \boxed{\mathbf{V}^\top}_{(d'\times d)}$$

- Low rank matrix factorization! (Truncated SVD solution...)

SVD Decomposition

- Any matrix $n \times d$ matrix A can be decomposed as



with U and W two orthonormal matrices and D a *diagonal* matrix with decreasing values.

Low Rank Approximation

- The best low rank approximation or rank r is obtained by restriction of the matrices to the first r dimensions:

$$\underset{(n\times d)}{\mathbf{A}} \simeq \underset{(n\times r)}{\mathbf{U_r}} \underset{(r\times r)}{D_{r,r}} \underset{(r\times d)}{\mathbf{W_r}^\top}$$

  for both the operator norm and the Frobenius norm!

- PCA: Low rank approximation with Frobenius norm, $d' = r$ and

$$\begin{pmatrix} \underline{X}_1^\top - m^\top \\ \vdots \\ \vdots \\ \underline{X}_n^\top - m^\top \end{pmatrix} \leftrightarrow A, \quad \begin{pmatrix} \underline{\tilde{X}}_1^\top \\ \vdots \\ \vdots \\ \underline{\tilde{X}}_n^\top \end{pmatrix} \leftrightarrow \mathbf{U_r} D_{r,r}, \quad \mathbf{V}^\top \leftrightarrow \mathbf{W_r}^\top$$

SVD Decompositions

- Recentered data:
$$\mathbf{R} = \begin{pmatrix} \underline{X_1}^\top - m^\top \\ \vdots \\ \underline{X_n}^\top - m^\top \end{pmatrix} = UDW^\top$$
- Covariance matrix:
$$\Sigma = \mathbf{R}^\top \mathbf{R} = WD^\top DW$$
with $D^\top D$ diagonal.
- Gram matrix (matrix of scalar products):
$$G = \mathbf{R}\mathbf{R}^\top = UDD^\top U$$
with $DD^\top$ diagonal.

$\Sigma_{\ell\ell'} = \frac{1}{n}\sum_i X_i^{(\ell)} X_i^{(\ell')}$

$G_{ij} = \frac{1}{\ell}\sum_\ell X_i^{(\ell)} X_j^{(\ell)}$

- Those are the same U, W and D, hence the link between all the approaches.

Outline

Reconstruction Error Approach

Goal

- Construct a map Φ from the space $\mathcal{X}$ into a space $\mathcal{X}'$ of **smaller dimension**:

$$\begin{aligned} \Phi : \quad & \mathcal{X} \to \mathcal{X}' \\ & \underline{X} \mapsto \Phi(\underline{X}) \end{aligned}$$

- Construct $\widetilde{\Phi}$ from $\mathcal{X}'$ to $\mathcal{X}$
- Control the error between $\underline{X}$ and its reconstruction $\widetilde{\Phi}(\Phi(\underline{X}))$

- Canonical example for $\underline{X} \in \mathbb{R}^d$: find Φ and $\widetilde{\Phi}$ in a parametric family that minimize

$$\frac{1}{n} \sum_{i=1}^{n} \|\underline{X}_i - \widetilde{\Phi}(\Phi(\underline{X}_i))\|^2$$

Principal Component Analysis

- $\mathcal{X} \in \mathbb{R}^d$ and $\mathcal{X}' = \mathbb{R}^{d'}$
- Affine model $\underline{X} \sim m + \sum_{l=1}^{d'} \underline{X}'^{(l)} V^{(l)}$ with $(V^{(l)})$ an orthonormal family.
- Equivalent to:
$$\Phi(\underline{X}) = V^\top(\underline{X} - m) \quad \text{and} \quad \widetilde{\Phi}(\underline{X}') = m + V\underline{X}'$$
- Reconstruction error criterion:
$$\frac{1}{n}\sum_{i=1}^{n} \|\underline{X}_i - (m + VV^\top(\underline{X}_i - m)\|^2$$
- **Explicit solution:** m is the empirical mean and V is any orthonormal basis of the space spanned by the d' first eigenvectors (the one with largest eigenvalues) of the empirical covariance matrix $\frac{1}{n}\sum_{i=1}^{n}(\underline{X}_i - m)(\underline{X}_i - m)^\top$.

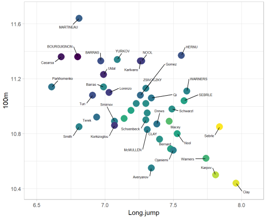
Principal Component Analysis

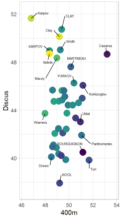
PCA Algorithm

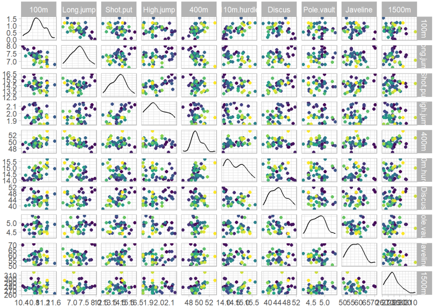
- Compute the empirical mean $m = \frac{1}{n}\sum_{i=1}^{n} \underline{X}_i$
- Compute the empirical covariance matrix $\frac{1}{n}\sum_{i=1}^{n}(\underline{X}_i - m)(\underline{X}_i - m)^\top$.
- Compute the d' first eigenvectors of this matrix: $V^{(1)}, \dots, V^{(d')}$
- Set $\Phi(\underline{X}) = V^\top(\underline{X} - m)$

- Complexity: $O(n(d + d^2) + d'd^2)$
- Interpretation:
  - $\Phi(\underline{X}) = V^\top(\underline{X} - m)$: coordinates in the restricted space.
  - $V^{(i)}$: influence of each original coordinates in the ith new one.
- **Scaling:** This method is not invariant to a scaling of the variables! It is custom to normalize the variables (at least within groups) before applying PCA.

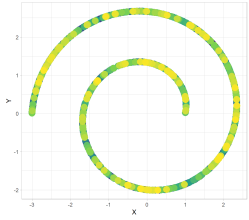
Decathlon

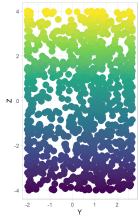

100m
Long.jump
11.6
11.2
10.8
10.4
6.5
7.0
7.5
8.0

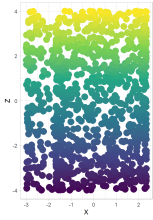

Discus
400m
52
48
44
40
46
48
50
52
54

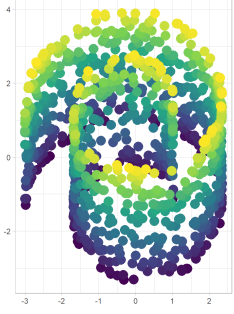

100m
Long.jump
Shot.put
High.jump
400m
Discus
Pole.vault
Javeline
1500m

Swiss Roll

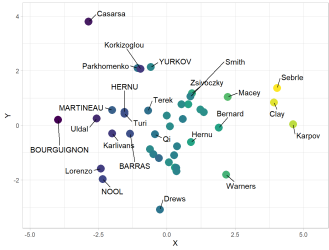




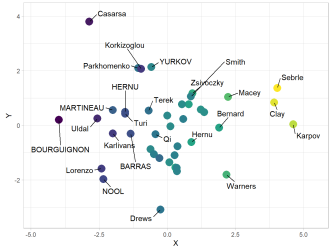




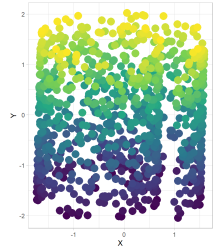
Principal Component Analysis



Decathlon



Decathlon
Renormalized



Swiss Roll

Normalisation automatique

Multiple Factor Analysis

- PCA assumes $\mathcal{X} = \mathbb{R}^d$!
- How to deal with categorical values?
- MFA = PCA with clever coding strategy for categorical values.

Categorical value code for a single variable

- Classical redundant dummy coding:
$$\underline{X} \in \{1, \dots, V\} \mapsto P(\underline{X}) = (\mathbf{1}_{\underline{X}=1}, \dots, \mathbf{1}_{\underline{X}=V})^\top$$
- Compute the mean (i.e. the empirical proportions): $\overline{P} = \frac{1}{n} \sum_{i=1}^{n} P(\underline{X}_i)$
- *Renormalize* $P(\underline{X})$ by $1/\sqrt{(V-1)\overline{P}}$:
$$P(\underline{X}) = (\mathbf{1}_{\underline{X}=1}, \dots \mathbf{1}_{\underline{X}=V}) \mapsto \left(\frac{\mathbf{1}_{\underline{X}=1}}{\sqrt{(V-1)\overline{P}_1}}, \dots, \frac{\mathbf{1}_{\underline{X}=V}}{\sqrt{(V-1)\overline{P}_V}} = P^r(\underline{X}) \right)$$

- χ^2 type distance!

Multiple Factor Analysis

- PCA becomes the minimization of

$$\frac{1}{n}\sum_{i=1}^{n}\|P^r(\underline{X}_i) - (m + VV^\top(P^r(\underline{X}_i) - m))\|^2$$

$$= \frac{1}{n}\sum_{i=1}^{n}\sum_{v=1}^{V}\frac{\left|\mathbf{1}_{\underline{X}_i=v} - (m' + \sum_{l=1}^{d'} V^{(l)\top}(P(\underline{X}_i) - m')V^{(l,v)})\right|^2}{(V-1)\overline{P}_v}$$

- Interpretation:
  - $m' = \overline{P}$
  - $\Phi(\underline{X}) = V^\top(P^r(\underline{X}) - m)$: coordinates in the restricted space.
  - $V^{(l)}$ can be interpreted s as a probability profile.
- Complexity: $O(n(V + V^2) + d'V^2)$
- Link with Correspondence Analysis (CA)

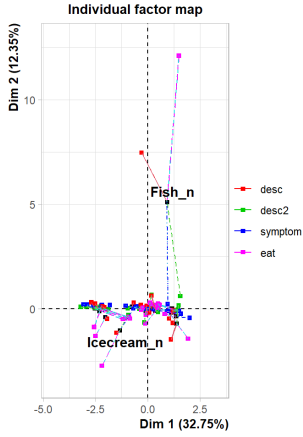
Multiple Factor Analysis

MFA Algorithm

- Redundant dummy coding of each categorical variable.
- Renormalization of each block of dummy variable.
- Classical PCA algorithm on the resulting variables

- Interpretation as a reconstruction error with a rescaled/χ^2 metric.
- Interpretation:
  - $\Phi(\underline{X}) = V^\top(P^r(\underline{X}) - m)$: coordinates in the restricted space.
  - $V^{(l)}$: influence of each modality/variable in the ith new coordinates.
- **Scaling:** This method is not invariant to a scaling of the continuous variables! It is custom to normalize the variables (at least within groups) before applying PCA.

Multiple Factor Analysis



Non Linear PCA

PCA Model

- PCA: Linear model assumption

$$\underline{X} \simeq m + \sum_{l=1}^{d'} \underline{X}^{\prime,(l)} V^{(l)} = m + V\underline{X}'$$

- with
  - $V^{(l)}$ orthonormal
  - $\underline{X}^{\prime,(l)}$ without constraints.

- Two directions of extension:
  - Other constraints on V (or the coordinates in the restricted space): ICA, NMF, Dictionary approach
  - PCA on a non-linear image of $\underline{X}$: kernel-PCA
- Much more complex algorithm!

Non Linear PCA

ICA (Independent Component Analysis)

- Linear model assumption

$$\underline{X} \simeq m + \sum_{l=1}^{d'} \underline{X}'^{,(l)} V^{(l)} = m + V\underline{X}'$$

- with
  - $V^{(l)}$ without constraints.
  - $\underline{X}'^{,(l)}$ *independent*

NMF (Non Negative Matrix Factorization)

- (Linear) Model assumption

$$\underline{X} \simeq \sum_{l=1}^{d'} \underline{X}'^{,(l)} V^{(l)} = V\underline{X}'$$

- with
  - $V^{(l)}$ non-negative
  - $\underline{X}'^{,(l)}$ non-negative.

Non Linear PCA

Dictionary

- (Linear) Model assumption

$$\underline{X} \simeq m + \sum_{l=1}^{d'} \underline{X}^{\prime,(l)} V^{(l)} = m + V\underline{X}'$$

- with
  - $V^{(l)}$ without constraints
  - $\underline{X}'$ sparse (with a lot of 0)

kernel PCA

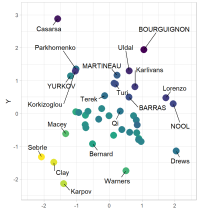
- Linear model assumption

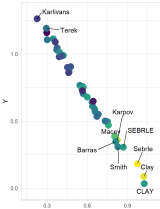
$$\Psi(\underline{X} - m) \simeq \sum_{l=1}^{d'} \underline{X}^{\prime,(l)} V^{(l)} = V\underline{X}'$$

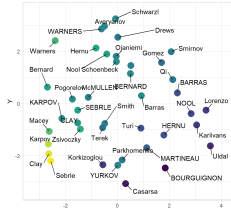
- with
  - $V^{(l)}$ orthonormal
  - $\underline{X}'_l$ without constraints.

Non Linear PCA

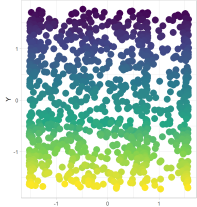
Decathlon

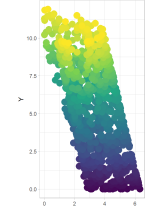


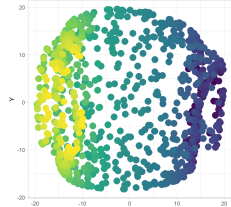




Swiss Roll







ICA NMF Kernel PCA

Deep Auto Encoder

- Construct a map Φ with a **NN** from the space $\mathcal{X}$ into a space $\mathcal{X}'$ of smaller dimension:

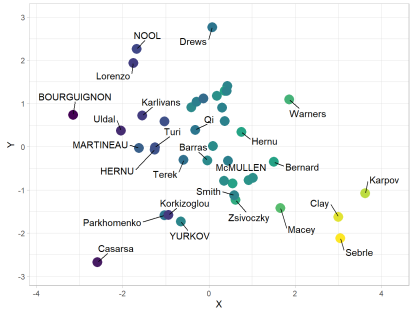
$$\begin{aligned} \Phi : \quad & \mathcal{X} \to \mathcal{X}' \\ & \underline{X} \mapsto \Phi(\underline{X}) \end{aligned}$$

- Construct $\widetilde{\Phi}$ with a **NN** from $\mathcal{X}'$ to $\mathcal{X}$
- Control the error between $\underline{X}$ and its reconstruction $\widetilde{\Phi}(\Phi(\underline{X}))$:

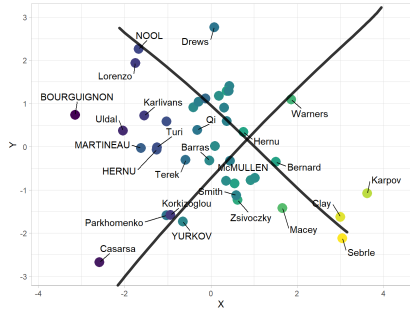
$$\frac{1}{n}\sum_{i=1}^{n} \|\underline{X}_i - \widetilde{\Phi}(\Phi(\underline{X}_i))\|^2$$

- Optimization by gradient descent.
- NN can be replaced by another parametric function...

Deep Auto Encoder



Shallow Auto Encoder (PCA)



Deep Auto Encoder

Outline

Pairwise Relation

- Different point of view!
- Focus on pairwise relation $\mathcal{R}(\underline{X}_i, \underline{X}_j)$.

Distance Preservation

- Construct a map Φ from the space $\mathcal{X}$ into a space $\mathcal{X}'$ of **smaller dimension**:

$$\begin{aligned} \Phi : \quad & \mathcal{X} \to \mathcal{X}' \\ & \underline{X} \mapsto \Phi(\underline{X}) = \underline{X}' \end{aligned}$$

- such that

$$\mathcal{R}(\underline{X}_i, \underline{X}_j) \sim \mathcal{R}'(\underline{X}'_i, \underline{X}'_j)$$

- Most classical version (MDS):
  - Scalar product relation: $\mathcal{R}(\underline{X}_i, \underline{X}_j) = (\underline{X}_i - m)^\top (\underline{X}_j - m)$
  - Linear mapping $\underline{X}' = \Phi(\underline{X}) = V^\top (\underline{X} - m)$.
  - Euclidean scalar product matching:

$$\frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \left| (\underline{X}_i - m)^\top (\underline{X}_j - m) - \underline{X}'^\top_i \underline{X}'_j \right|^2$$

- Φ often defined only on $\mathcal{D}\ldots$

MultiDimensional Scaling

MDS Heuristic

- Match the *scalar* products:
$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left|(\underline{X}_i - m)^\top(\underline{X}_j - m) - \underline{X}_i{}'^\top \underline{X}'_j\right|^2$$
- Linear method: $\underline{X}' = U^\top(\underline{X} - m)$ with U orthonormal

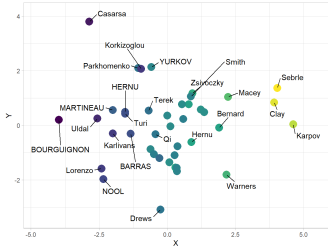
- **Beware:** $\underline{X}$ can be unknown, only the scalar products are required!
- Resulting criterion: minimization in $U^\top(\underline{X}_i - m)$ of
$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left|(\underline{X}_i - m)^\top(\underline{X}_j - m) - (\underline{X}_i - m)^\top UU^\top(\underline{X}_j - m)\right|^2$$
without using explicitly $\underline{X}$ in the algorithm...
- Explicit solution obtained through the eigendecomposition of the know Gram matrix $(\underline{X}_i - m)^\top(\underline{X}_j - m)$ by keeping only the d' largest eigenvalues.

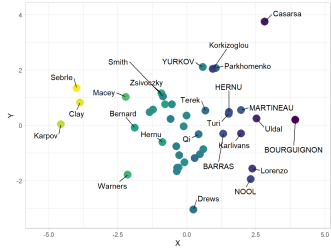
MultiDimensional Scaling

- In this case, MDS yields the same result as the PCA (but with different inputs, distance between observation vs correlations)!
- **Explanation:** Same SVD problem up to a transposition:
  - MDS
  $$\overline{X}_{(n)}{}^{\top}\overline{X}_{(n)} \sim \overline{X}_{(n)}{}^{\top} U U^{\top} \overline{X}_{(n)}$$
  - PCA
  $$\overline{X}_{(n)}\overline{X}_{(n)}{}^{\top} \sim U^{\top}\overline{X}_{(n)}\overline{X}_{(n)}{}^{\top} U$$
- Complexity: PCA $O((n+d')d^2)$ vs MDS $O((d+d')n^2)\ldots$

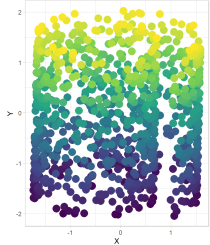
MultiDimensional Scaling

Decathlon

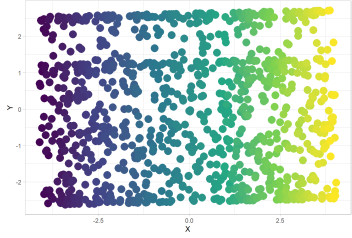




Swiss Roll



PCA



MDS

Generalized MDS

- Preserving the scalar products amounts to preserve the Euclidean distance.
- Easier **generalization** if we work in terms of distance!

Generalized MDS

- Generalized MDS:
  - Distance relation: $\mathcal{R}(\underline{X}_i, \underline{X}_j) = d(\underline{X}_i, \underline{X}_j)$
  - Linear mapping $\underline{X}' = \Phi(\underline{X}) = V^\top(\underline{X} - m)$.
  - Euclidean matching:

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left|d(\underline{X}_i, \underline{X}_j) - d'(\underline{X}'_i, \underline{X}'_j)\right|^2$$

- Strong connection (but no equivalence) with MDS when $d(x, y) = \|x - y\|^2$!
- **Minimization:** Simple gradient descent can be used (can be stuck in local minima).

ISOMAP

- MDS: equivalent to PCA (but more expensive) if $d(x, y) = \|x - y\|^2$!
- ISOMAP: use a *localized* distance instead to limit the influence of very far point.

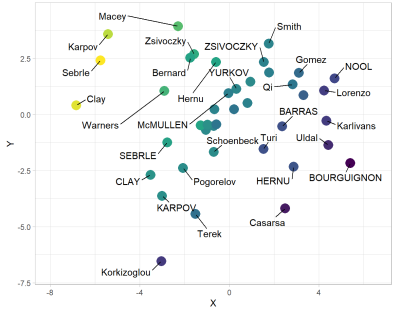
ISOMAP

- For each point $\underline{X}_i$, define a neighborhood $\mathcal{N}_i$ (either by a distance or a number of points) and let

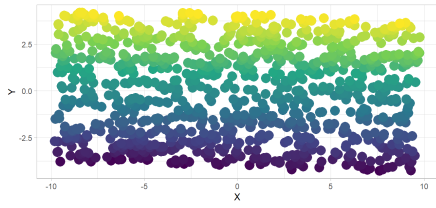
$$d_0(\underline{X}_i, \underline{X}_j) = \begin{cases} +\infty & \text{if } \underline{X}_j \notin \mathcal{N}_i \\ \|\underline{X}_i - \underline{X}_j\|^2 & \text{otherwise} \end{cases}$$

- Compute the shortest path distance for each pair.
- Use the MDS algorithm with this distance

ISOMAP



Decathlon



Swiss Roll

Random Projection

Random Projection Heuristic

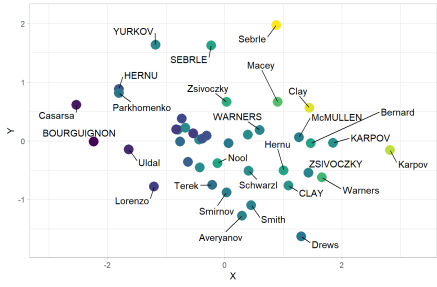
- Draw at random d' unit vector (direction) U_i.
- Use $\underline{X}' = U^\top(\underline{X} - m)$ with $m = \frac{1}{n}\sum_{i=1}^{n} \underline{X}_i$

- **Property:** If $\underline{X}$ lives in a space of dimension d'', then, as soon as, $d' \sim d'' \log(d'')$,

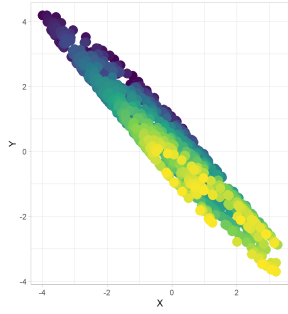
$$\|\underline{X}_i - \underline{X}_j\|^2 \sim \frac{d}{d'}\|\underline{X}'_i - \underline{X}'_j\|^2$$

- Do not really use the data!

Random Projection



Decathlon



Swiss Roll

t-Stochastic Neighbor Embedding

SNE heuristic

- From $\underline{X}_i \in \mathcal{X}$, construct a set of conditional probability:

$$P_{j|i} = \frac{e^{-\|\underline{X}_i - \underline{X}_j\|^2/2\sigma_i^2}}{\sum_{k \neq i} e^{-\|\underline{X}_i - \underline{X}_k\|^2/2\sigma_i^2}} \qquad P_{i|i} = 0$$

- Find $\underline{X}'_i$ in $\mathbb{R}^{d'}$ such that the set of conditional probability:

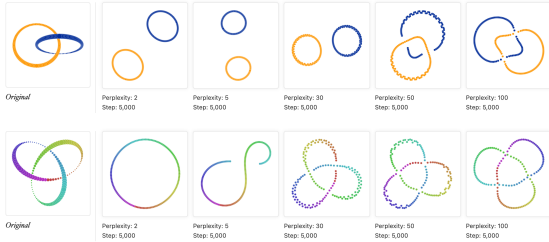
$$Q_{j|i} = \frac{e^{-\|\underline{X}'_i - \underline{X}'_j\|^2/2\sigma_i^2}}{\sum_{k \neq i} e^{-\|\underline{X}'_i - \underline{X}'_k\|^2/2\sigma_i^2}} \qquad Q_{i|i} = 0$$

  is close from P.

- **t-SNE:** use a Student-t term $(1 + \|\underline{X}'_i - \underline{X}'_j\|^2)^{-1}$ for $\underline{X}'_i$
- Minimize the Kullback-Leibler divergence $(\sum_{i,j} P_{j|i} \log \frac{P_{j|i}}{Q_{j|i}})$ by a simple gradient descent (can be stuck in local minima).
- Parameters σ_i such that $H(P_i) = -\sum_{j=1}^{n} P_{j|i} \log P_{j|i} = \text{cst}$.

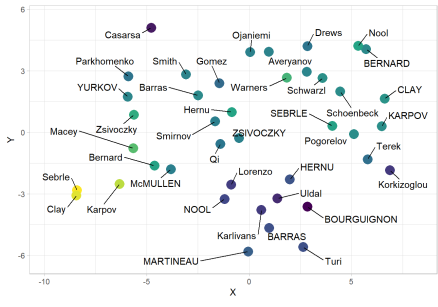
t-Stochastic Neighbor Embedding

- Very successful/ powerful technique in practice
- Convergence may be long, unstable, or strongly depending on parameters.
- See this distill post for many impressive examples

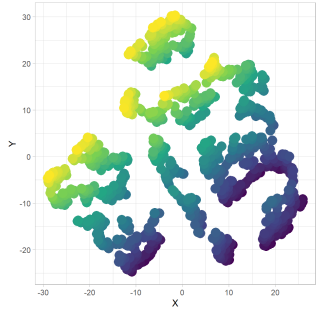


Representation depending on t-SNE parameters

t-SNE



Decathlon



Swiss Roll

UMAP

- Topological Data Analysis inspired.

Uniform Manifold Approximation and Projection

- Define a notion of asymmetric scaled local proximity between neighbors:
  - Compute the k-neighborhood of $\underline{X}_i$, its diameter σ_i and the distance ρ_i between $\underline{X}_i$ and its nearest neighbor.
  - Define

$$w_i(\underline{X}_i, \underline{X}_j) = \begin{cases} e^{-(d(\underline{X}_i, \underline{X}_j) - \rho_i)/\sigma_i} & \text{for } \underline{X}_j \text{ in the } k\text{-neighborhood} \\ 0 & \text{otherwise} \end{cases}$$

- Symmetrize into a *fuzzy* nearest neighbor criterion

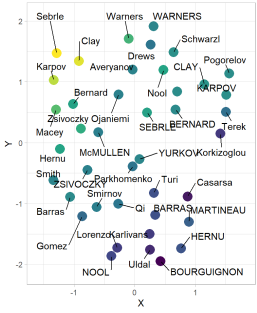
$$w(\underline{X}_i, \underline{X}_j) = w_i(\underline{X}_i, \underline{X}_j) + w_j(\underline{X}_j, \underline{X}_i) - w_i(\underline{X}_i, \underline{X}_j) w_j(\underline{X}_j, \underline{X}_i)$$

- Determine the points $\underline{X}'_i$ in a low dimensional space such that

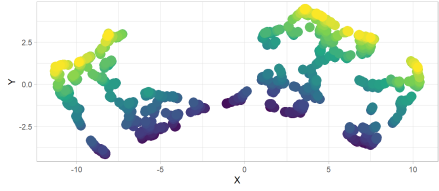
$$\sum_{i \neq j} w(\underline{X}_i, \underline{X}_j) \log \left(\frac{w(\underline{X}_i, \underline{X}_j)}{w'(\underline{X}'_i, \underline{X}'_j)} \right) + (1 - w(\underline{X}_i, \underline{X}_j)) \log \left(\frac{(1 - w(\underline{X}_i, \underline{X}_j))}{(1 - w'(\underline{X}'_i, \underline{X}'_j))} \right)$$

- Can be performed by local gradient descent.

UMAP



Decathlon



Swiss Roll

Graph based

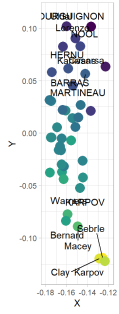
Graph heuristic

- Construct a graph with weighted edges $w_{i,j}$ measuring the *proximity* of $\underline{X}_i$ and $\underline{X}_j$ ($w_{i,j}$ large if close and 0 if there is no information).
- Find the points $\underline{X}'_i \in \mathbb{R}^{d'}$ minimizing

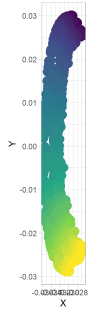
$$\frac{1}{n}\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{i,j}\|\underline{X}'_i - \underline{X}'_j\|^2$$

- Need of a constraint on the size of $\underline{X}'_i \ldots$
- Explicit solution through linear algebra: d' eigenvectors with smallest eigenvalues of the Laplacian of the graph $D - W$, where D is a diagonal matrix with $D_{i,i} = \sum_j w_{i,j}$.
- Variation on the definition of the Laplacian...

Graph



Decathlon



Swiss Roll

Outline

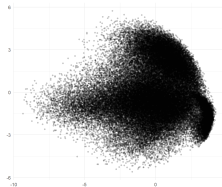
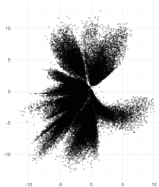
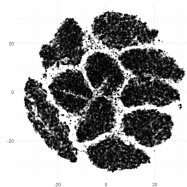
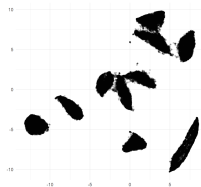
How to Compare Different Dimensionality Reduction Methods ?

- **Difficult!** Once again, the metric is very subjective.

However, a few possible attempts

- Did we preserve a lot of inertia with only a few directions?
- Do those directions *make sense* from an expert point of view?
- Do the low dimension representation *preserve* some important information?
- Are we better on **subsequent task**?

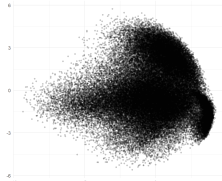
A Challenging Example: MNIST

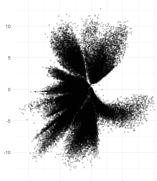
MNIST Dataset

- Images of 28×28 pixels.
- No label used!
- 4 different embeddings.

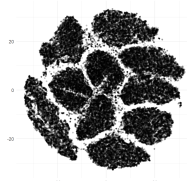
A Challenging Example: MNIST



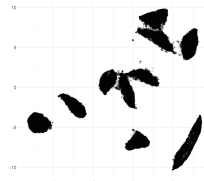
PCA



autoencoder



t-SNE

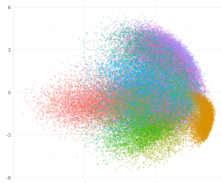


UMAP

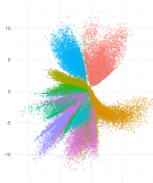
MNIST Dataset

- Images of 28×28 pixels.
- No label used!
- 4 different embeddings.

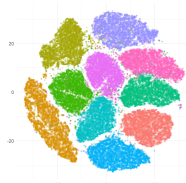
A Challenging Example: MNIST



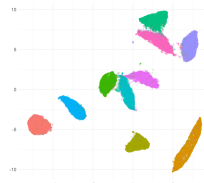
PCA



autoencoder



t-SNE



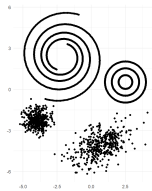
UMAP

MNIST Dataset

- Images of 28×28 pixels.
- No label used!
- 4 different embeddings.

- Quality evaluated by visualizing the true labels **not used to obtain the embeddings.**
- Only a few labels could have been used.

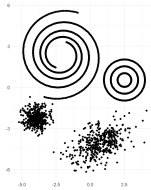
A Simpler Example: A 2D Set



Cluster Dataset

- Set of points in 2D.
- No label used!
- 3 different embeddings.

A Simpler Example: A 2D Set



PCA

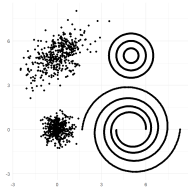
t-SNE

UMAP

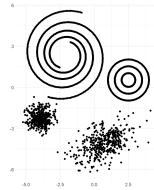
Cluster Dataset

- Set of points in 2D.
- No label used!
- 3 different embeddings.

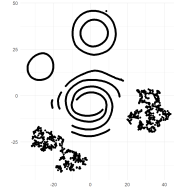
A Simpler Example: A 2D Set



Original



PCA



t-SNE

UMAP

Cluster Dataset

- Set of points in 2D.
- No label used!
- 3 different embeddings.

- Quality evaluated by stability...

Outline

Clustering

- **Training data** : $\mathcal{D} = \{\underline{X}_1, \ldots, \underline{X}_n\} \in \mathcal{X}^n \quad (\text{i.i.d. } \sim \mathbb{P})$
- Latent groups?

Clustering

- Construct a map f from $\mathcal{D}$ to $\{1, \ldots, K\}$ where K is a number of classes to be fixed:

$$f : \quad \underline{X}_i \mapsto k_i$$

Motivations

- Interpretation of the groups
- Use of the groups in further processing

- Several strategies possible!
- Can use dimension reduction as a preprocessing.

Outline

Partition Based

Partition Heuristic

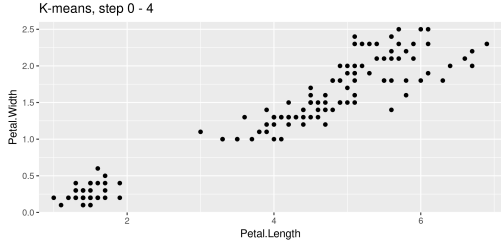
- Clustering is defined by a partition in K classes...
- that minimizes a homogeneity criterion.

K- Means

- Cluster k defined by a *center* μ_k.
- Each sample is associated to the closest center.
- Centers defined as the minimizer of $\sum_{i=1}^{n} \min_k \|\underline{X}_i - \mu_k\|^2$

- Iterative scheme (Loyd):
  - Start by a (pseudo) random choice for the centers μ_k
  - Assign each samples to its nearby center
  - Replace the center of a cluster by the mean of its assigned samples.
  - Repeat the last two steps until convergence.

Partition Based



Partition based

- Other schemes:
  - McQueen: modify the mean each time a sample is assigned to a new cluster.
  - Hartigan: modify the mean by removing the considered sample, assign it to the nearby center and recompute the new mean after assignment.

A good initialization is crucial!

- Initialize by samples.
- k-Mean++: try to take them as separated as possible.
- No guarantee to converge to a global optimum: repeat and keep the best result!

- Complexity : $O(n \times K \times T)$ where T is the number of steps in the algorithm.

Partition based

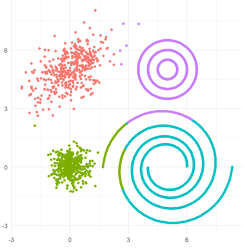
- k-Medoid: use a sample as a center
  - PAM: for a given cluster, use the sample that minimizes the intra distance (sum of the squared distance to the other points)
  - Approximate medoid: for a given cluster, assign the point that is the closest to the mean.

Complexity

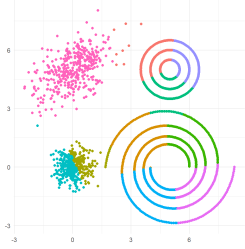
- PAM: $O(n^2 \times T)$ in the worst case!
- Approximate medoid: $O(n \times K \times T)$ where T is the number of steps in the algorithm.

- **Remark:** Any distance can be used... but the complexity of computing the centers can be very different.

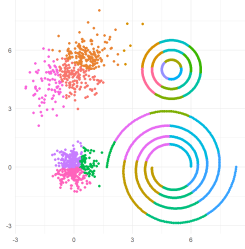
K-Means



$k = 4$



$k = 10$



$k = 10$

Model Based

Model Heuristic

- Use a generative model of the data:

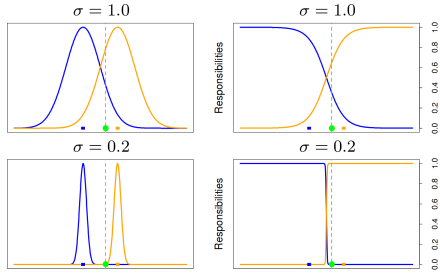
$$\mathbb{P}(\underline{X}) = \sum_{k=1}^{K} \pi_k \mathbb{P}_{\theta_k}(\underline{X}|k)$$

  where π_k are proportions and $\mathbb{P}_\theta(\underline{X}|k)$ are parametric probability models.
- Estimate those parameters (often by a ML principle).
- Assign each observation to the class maximizing the a posteriori probability (obtained by Bayes formula)

$$\frac{\widehat{\pi_k}\mathbb{P}_{\widehat{\theta_k}}(\underline{X}|k)}{\sum_{k'=1}^{K}\widehat{\pi_{k'}}\mathbb{P}_{\widehat{\theta_{k'}}}(\underline{X}|k')}$$

- Link with Generative model in supervised classification!

Model Based



A two class example

- A mixture $\pi_1 f_1(\underline{X}) + \pi_2 f_2(\underline{X})$
- and the posterior probability $\pi_i f_i(\underline{X})/(\pi_1 f_1(\underline{X}) + \pi_2 f_2(\underline{X}))$
- Natural class assignment!

Model Based

Sub-population estimation

- A mixture $\pi_1 f_1(\underline{X}) + \pi_2 f_2(\underline{X})$
- Two populations with a parametric distribution f_i.
- Most classical choice: Gaussian distribution

Gaussian Setting

- $\underline{X}_1, \ldots, \underline{X}_n$ independent
- $\underline{X}_i \sim \mathrm{N}(\mu_1, \sigma_1^2)$ with probability π_1 or $\underline{X}_i \sim \mathrm{N}(\mu_2, \sigma_2^2)$ with probability π_2
- We don't know the parameters μ_i, σ_i, π_i.
- We don't know from which distribution each $\underline{X}_i$ has been drawn.

Model Based

Maximum Likelihood

- Density: $\pi_1 \Phi(\underline{X}, \mu_1, \sigma_1^2) + \pi_2 \Phi(\underline{X}, \mu_2, \sigma_2^2)$
- log-likelihood: $\mathcal{L}(\theta) = \sum_{i=1}^{n} \log \left(\pi_1 \Phi(\underline{X}_i, \mu_1, \sigma_1^2) + \pi_2 \Phi(\underline{X}_i, \mu_2, \sigma_2^2) \right)$
- No straightforward way to optimize the parameters!

What if algorithm

- Assume we know from which distribution each sample has been sampled: $Z_i = 1$ if from f_1 and $Z_i = 0$ otherwise.
- log-likelihood: $\sum_{i=1}^{n} Z_i \log \ \Phi(\underline{X}_i, \mu_1, \sigma_1^2) + (1 - Z_i) \log \Phi(\underline{X}_i, \mu_2, \sigma_2^2)$
- Easy optimization... but the Z_i are unknown!

Model Based

What if algorithm

- Assume we know from which distribution each sample has been sampled: $Z_i = 1$ if from f_1 and $Z_i = 0$ otherwise.
- log-likelihood: $\sum_{i=1}^{n} Z_i \log \Phi(\underline{X}_i, \mu_1, \sigma_1^2) + (1 - Z_i) \log \Phi(\underline{X}_i, \mu_2, \sigma_2^2)$
- Easy optimization... but the Z_i are unknown!

Bootstrapping Idea

- Replace Z_i by its expectation given the current estimate.
- $\mathbb{E}[Z_i] = \mathbb{P}(Z_i = 1|\theta)$ (A posteriori probability)
- and iterate...
- Can be proved to be good idea!

EM Algorithm

- (Random) initialization: μ_i^0, σ_i^0, π_i^0.
- Repeat:
  - Expectation (Current a posteriori probability):

$$\mathbb{E}_t[Z_i] = \mathbb{P}\left(Z_i = 1 | \theta^t\right) = \frac{\pi_1^t \Phi(\underline{X}_i, \mu_1^t, (\sigma_1^t)^2)}{\pi_1^t \Phi(\underline{X}_i, \mu_1^t, (\sigma_1^t)^2) + \pi_2^t \Phi(\underline{X}_i, \mu_2^t, (\sigma_2^t)^2)}$$

  - Maximization of

$$\sum_{i=1}^{n} \mathbb{E}_t[Z_i] \log \ \Phi(\underline{X}_i, \mu_1, \sigma_1^2) + \mathbb{E}_t[1 - Z_i] \log \Phi(\underline{X}_i, \mu_2, \sigma_2^2)$$

    to obtain μ_i^{t+1}, σ_i^{t+1}, π_i^{t+1}.

Model Based

- Large choice of parametric models.

Gaussian Mixture Model

- Use
$$\mathbb{P}_{\theta_k}\left(\vec{\underline{X}}|k\right) \sim \mathrm{N}(\mu_k, \Sigma_k)$$
with $\mathrm{N}(\mu, \Sigma)$ the Gaussian law of mean μ and covariance matrix Σ.

- Efficient optimization algorithm available (EM)
- Often some constraint on the covariance matrices: identical, with a similar structure...
- Strong connection with K-means when the covariance matrices are assumed to be the same multiple of the identity.

Model Based

Probabilistic latent semantic analysis (PLSA)

- Documents described by their word counts w
- Model:

$$\mathbb{P}(w) = \sum_{k=1}^{K} \pi_k \mathbb{P}_{\theta_k}(w|k)$$

with k the (hidden) topic, π_k a topic probability and $\mathbb{P}_{\theta_k}(w|k)$ a multinomial law for a given topic.

- Clustering according to

$$\mathbb{P}(k|w) = \frac{\widehat{\pi_k}\mathbb{P}_{\widehat{\theta_k}}(w|k)}{\sum_{k'} \widehat{\pi_{k'}}\mathbb{P}_{\widehat{\theta_{k'}}}(w|k')}$$

- Same idea than GMM!
- Bayesian variant called LDA.

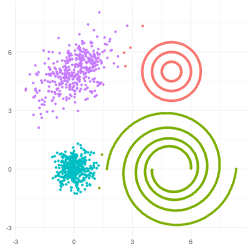
Model Based

Parametric Density Estimation Principle

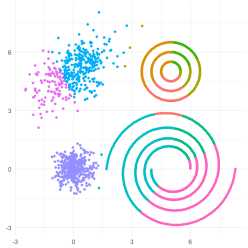
- Assign a probability of membership.
- Lots of theoretical studies...
- Model selection principle can be used to select K the number of classes (or rather to avoid using a nonsensical K...):
  - AIC / BIC / MDL penalization
  - Cross Validation is also possible!

- Complexity: $O(n \times K \times T)$

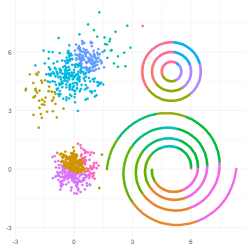
Gaussian Mixture Models



$k = 4$



$k = 10$



$k = 10$

Outline

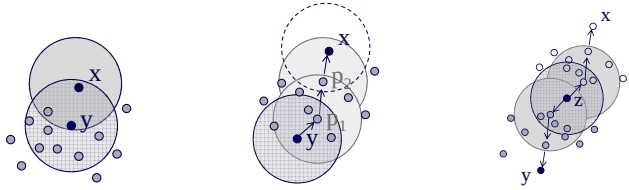
(Non Parametric) Density Based

Density Heuristic

- Cluster are connected dense zone separated by low density zone.
- Not all points belong to a cluster.

- Basic bricks:
  - Estimate the density.
  - Find points with high densities.
  - Gather those points according to the density.
- Density estimation:
  - Classical kernel density estimators...
- Gathering:
  - Link points of high density and use the resulted component.
  - Move them toward top of density *hill* by following the gradient and gather all the points arriving at the same *summit*.

(Non Parametric) Density Based

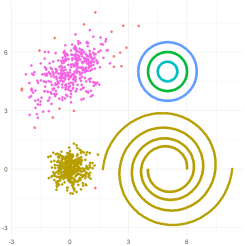


Examples

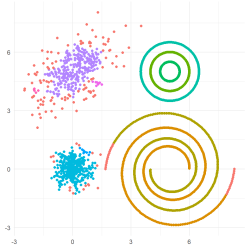
- DBSCAN: link point of high densities using a very simple kernel.
- PdfCLuster: find connected zone of high density.
- Mean-shift: move points toward top of density *hill* following an evolving kernel density estimate.

- Complexity: $O(n^2 \times T)$ in the worst case.
- Can be reduced to $O(n \log(n) T)$ if samples can be encoded in a tree structure (n-body problem type approximation).

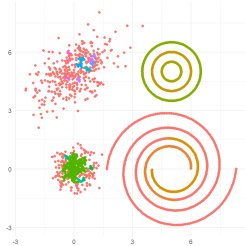
DBSCAN



$\epsilon = .45$



$\epsilon = .2$



$\epsilon = .1$

Outline

Agglomerative Clustering

Agglomerative Clustering Heuristic

- Start with very small clusters (a sample by cluster?)
- Sequential merging of the most similar clusters...
- according to some *greedy* criterion Δ.

- Generates a hierarchy of clustering instead of a single one.
- Need to select the number of cluster afterwards.
- Several choices for the merging criterion...
- Examples:
  - Minimum Linkage: merge the closest cluster in term of the usual distance
  - Ward's criterion: merge the two clusters yielding the less inner inertia loss (k-means criterion)

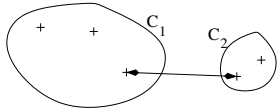
Agglomerative Clustering

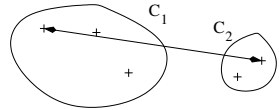
Algorithm

- Start with $(\mathcal{C}_i^{(0)}) = (\{X_i\})$ the collection of all singletons.
- At step s, we have $n - s$ clusters $(\mathcal{C}_i^{(s)})$:
  - Find the two most similar clusters according to a criterion Δ:
    $$(i, i') = \underset{(j,j')}{\operatorname{argmin}}\, \Delta(\mathcal{C}_j^{(s)}, \mathcal{C}_{j'}^{(s)})$$
  - Merge $\mathcal{C}_i^{(s)}$ and $\mathcal{C}_{i'}^{(s)}$ into $\mathcal{C}_i^{(s+1)}$
  - Keep the $n - s - 2$ other clusters $\mathcal{C}_{i''}^{(s+1)} = \mathcal{C}_{i''}^{(s)}$
- Repeat until there is only one cluster.

- Complexity: $O(n^3)$ in general.
- Can be reduced to $O(n^2)$
  - if only a bounded number of merging is possible for a given cluster,
  - for the most classical distances by maintaining a nearest neighbors list.

Agglomerative Clustering





Merging criterion based on the distance between points

- Minimum linkage:

$$\Delta(\mathcal{C}_i, \mathcal{C}_j) = \min_{\underline{X}_i \in \mathcal{C}_i} \min_{\underline{X}_\in \mathcal{C}_j} d(\underline{X}_i, \underline{X}_j)$$

- Maximum linkage:

$$\Delta(\mathcal{C}_i, \mathcal{C}_j) = \max_{\underline{X}_i \in \mathcal{C}_i} \max_{\underline{X}_\in \mathcal{C}_j} d(\underline{X}_i, \underline{X}_j)$$

- Average linkage:

$$\Delta(\mathcal{C}_i, \mathcal{C}_j) = \frac{1}{|\mathcal{C}_i||\mathcal{C}_j|} \sum_{\underline{X}_i \in \mathcal{C}_i} \sum_{\underline{X}_\in \mathcal{C}_j} d(\underline{X}_i, \underline{X}_j)$$

- Clustering based on the proximity...

Source: E. Matzner-Löber

Agglomerative Clustering

Merging criterion based on the inertia (distance to the mean)

- Ward's criterion:

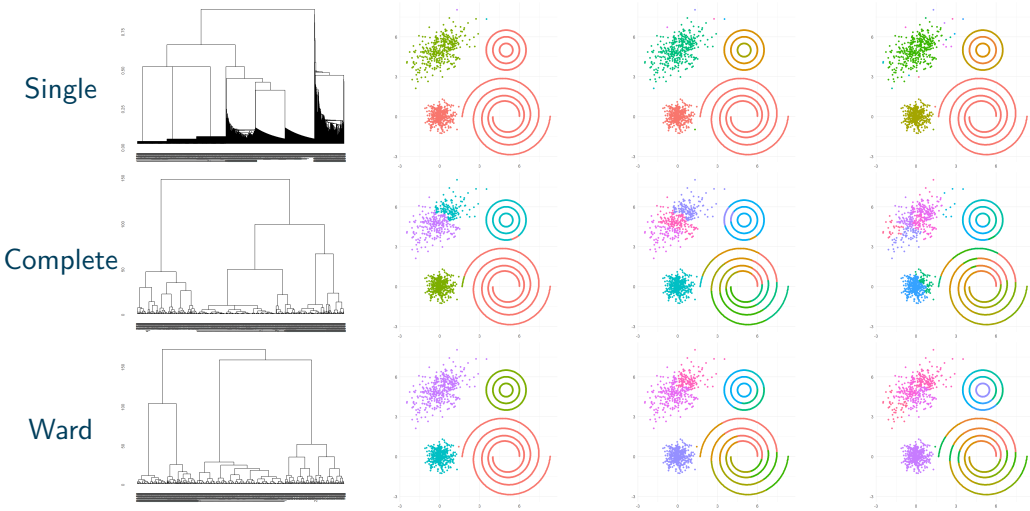
$$\Delta(\mathcal{C}_i, \mathcal{C}_j) = \sum_{\underline{X}_i \in \mathcal{C}_i} \left(d^2(\underline{X}_i, \mu_{\mathcal{C}_i \cup \mathcal{C}_j}) - d^2(\underline{X}_i, \mu_{\mathcal{C}_i}) \right) + \sum_{\underline{X}_j \in \mathcal{C}_j} \left(d^2(\underline{X}_j, \mu_{\mathcal{C}_i \cup \mathcal{C}_j}) - d^2(\underline{X}_j, \mu_{\mathcal{C}_j}) \right)$$

- If d is the Euclidean distance:

$$\Delta(\mathcal{C}_i, \mathcal{C}_j) = \frac{2|\mathcal{C}_i||\mathcal{C}_j|}{|\mathcal{C}_i| + |\mathcal{C}_j|} d^2(\mu_{\mathcal{C}_i}, \mu_{\mathcal{C}_j})$$

- Same criterion than in the k-means algorithm but greedy optimization.

Agglomerative Clustering



Outline

Grid based

Grid heuristic

- Split the space in pieces
- Group those of high density according to their proximity

- Similar to density based estimate (with partition based initial clustering)
- Space splitting can be fixed or adaptive to the data.
- Examples:
  - STING (Statistical Information Grid): Hierarchical tree construction plus DBSCAN type algorithm
  - AMR (Adaptive Mesh Refinement): Adaptive tree refinement plus k-means type assignment from high density leaves.
  - CLIQUE: Tensorial grid and 1D detection.
- Linked to Divisive clustering (DIANA)

Others

Graph based

- Spectral clustering: dimension reduction + k-means.
- Message passing: iterative local algorithm.
- Graph cut: min/max flow.

- Kohonen Map,
- ...

Outline

Scalability

Large dataset issue

- When n is large, a $O(n^\alpha \log n)$ with $\alpha > 1$ is not acceptable!
- How to deal with such a situation?

- **Beware:** Computing all the pairwise distance requires $O(n^2)$ operations!

Ideas

- Sampling
- Online processing
- Simplification
- Parallelization

Sampling

Sampling heuristic

- Use only a subsample to construct the clustering.
- Assign the other points to the constructed clusters afterwards.

- Requires a clustering method that can assign new points (partition, model...)
- Often repetition and choice of the best clustering
- Example:
  - CLARA: K-medoid with sampling and repetition
- Two-steps algorithm:
  - Generate a large number n' of clusters using a fast algorithm (with $n' \ll n$)
  - Cluster the clusters with a more accurate algorithm.

Online

Online heuristic

- Modify the current clusters according to the value of a single observation.

- Requires compactly described clusters.
- Examples:
  - Add to an existing cluster (and modify it) if it is close enough and create a new cluster otherwise (k-means without reassignment)
  - Stochastic descent gradient (GMM)
- May leads to far from optimal clustering.

Simplification

Simplification heuristic

- Simplify the algorithm to be more efficient at the cost of some precision.

- Algorithm dependent!
- Examples:
  - Replace groups of observation (preliminary cluster) by the (approximate) statistics.
  - Approximate the distances by cheaper ones.
  - Use n-body type techniques.

Parallelization

Parallelization heuristic

- Split the computation on several computers.

- Algorithm dependent!
- Examples:
  - Distance computation in k-means, parameter gradient in model based clustering
  - Grid density estimation, Space splitting strategies
- Classical batch sampling not easy to perform as partitions are not easily merged...

Outline

Generative Modeling

- **Training data** : $\mathcal{D} = \{(\underline{X}_1, \underline{Y}_1), \ldots, (\underline{X}_n, \underline{Y}_n)\} \in (\mathcal{X} \times \mathcal{Y})^n \quad (\text{i.i.d. } \sim \mathbb{P})$
- Same kind of data than for supervised learning if $\mathcal{Y} \neq \emptyset$.

Generative Modeling

- Construct a map G from the product of $\mathcal{Y}$ and a randomness source Ω to $\mathcal{X}$

$$G : \mathcal{Y} \times \Omega \to \mathcal{X}$$
$$(Y, \omega) \mapsto X$$

- Unconditional model if $\mathcal{Y} = \emptyset \ldots$

Motivation

- Generate plausible novel conditional samples based on a given dataset.

Sample Quality

- Related to the proximity between the law of $G(Y, \omega)$ and the law of $X|Y$.
- Most classical choice is the Kullback-Leibler divergence.

Generative Modeling

Ingredients

- Generator $G_\theta(Y, \omega)$ and cond. density prob. $P_\theta(X|Y)$ (Explicit vs implicit link)
- Simple / Complex / Approximate estimation...

Some Possible Choices

	Probabilistic model	Generator	Estimation
Base	Simple (parametric)	Explicit	Simple (ML)
Flow	Image of simple model	Explicit	Simple (ML)
Factorization	Factorization of simple model	Explicit	Simple (ML)
VAE	Simple model with latent var.	Explicit	Approximate (ML)
EBM	Arbitrary	Implicit (MCMC)	Complex (ML/score/discrim.)
Diffusion	Continuous noise	Implicit (MCMC)	Complex (score)
	Discrete Noise with latent var.	Explicit	Approximate (ML)
GAN	Implicit	Explicit	Complex (Discrimination)

- SOTA: Diffusion based approach!

ML: Maximum Likelihood/VAE: Variational AutoEncoder/EBM: Energy Based Model/MCMC: Monte Carlo Markov Chain/GAN: Generative Aversarial Network

Generators

$$\widetilde{X} = G(Y, \omega) \quad ?$$

- Small abuse of notations...
- More an algorithm than a map!

Generators

- One step: $\omega \sim \widetilde{Q}(\cdot|Y)$ and $\widetilde{X} = G(Y, \omega)$.
- Several steps:
  - $\omega_0 \sim \widetilde{Q}_0(\cdot|Y)$ and $\widetilde{X}_0 = G_0(\omega_0)$
  - $\omega_{t+1} \sim \widetilde{Q}_{t+1}(\cdot|Y, \widetilde{X}_t)$ and $\widetilde{X}_{t+1} = G_{t+1}(Y, \widetilde{X}_k, \omega_{t+1})$
- Fixed or variable number of steps.
- Fixed or variable dimension for $\widetilde{X}_t$ and ω_t...

- $\widetilde{Q}$ (or $\widetilde{Q}_t$) should be easy to sample.
- Most of the time, parametric representations for $\widetilde{Q}$ (or $\widetilde{Q}_t$) and G (or G_t).

Outline

Warmup: Density Estimation and Generative Modeling

$$X \sim P \text{ with } dP(x) = p(x)d\lambda \longrightarrow \widetilde{X} \sim \widetilde{P} \text{ with } d\widetilde{P}(x) = \widetilde{p}(x)d\lambda$$

Heuristic

- Estimate p by $\widetilde{p}$ from an i.i.d. sample $X_1, \ldots, X_n$.
- Simulate $\widetilde{X}$ having a law $\widetilde{P}$.

- By construction, if $\widetilde{p}$ is *close* from p, the law of $\widetilde{X}$ will be close from the law of X.

Issue: How to do it?

- How to estimate $\widetilde{p}$? Parametric, non-parametric? Maximum likelihood? Other criteria?
- How to simulate $\widetilde{P}$? Parametric? One-step? Multi-step? Iterative?

Warmup: Parametric Density Estimation

$$X \sim P(\cdot) \text{ with } dP(x) = p(x)d\lambda \longrightarrow \widetilde{X} \sim \widetilde{P}_{\widetilde{\theta}} \text{ with } d\widetilde{P}_{\widetilde{\theta}}(x) = \widetilde{p}_{\widetilde{\theta}}(x)d\lambda$$

Maximum Likelihood Approach

- Select a family $\widetilde{P}$ and estimate p by $\widetilde{p}_{\widetilde{\theta}}$ from an i.i.d. sample $X_1, \ldots, X_n$.
- Simulate $\widetilde{X}$ having a law $\widetilde{P}_{\widetilde{\theta}}$.

- By construction, if $\widetilde{p}_{\widetilde{\theta}}$ is *close* from p, the law of $\widetilde{X}$ will be close from the law of X.

Issue: How to do it?

- Which family $\widetilde{P}$?
- How to simulate $\widetilde{P}_{\widetilde{\theta}}$? Parametric? Iterative?

- Corresponds to $\omega \sim \widetilde{P}_{\widetilde{\theta}}$ and $\widetilde{X} = G(\omega) = \omega$

Conditional Density Est. and Generative Modeling

$$X|Y \sim P(\cdot|Y) \text{ with } dP(x|Y) = p(x|Y)d\lambda$$
$$\longrightarrow \widetilde{X}|Y \sim \widetilde{P}(\cdot|Y) \text{ with } d\widetilde{P}(x|Y) = \widetilde{p}(x|Y)d\lambda$$

Heuristic

- Estimate p by $\widetilde{p}$ from an i.i.d. sample $(X_1, Y_1), \ldots, (X_n, Y_n)$.
- Simulate $\widetilde{X}|Y$ having a law $\widetilde{P}(\cdot|Y)$.

- By construction, if $\widetilde{p}$ is *close* from p, the law of $\widetilde{X}|Y$ will be close from the law of $X|Y$.

Issue: How to do it?

- How to estimate $\widetilde{p}$? Parametric, non-parametric? Maximum likelihood? Other criteria?
- How to simulate $\widetilde{P}$? Parametric? One-step? Multi-step? Iterative?

Parametric Conditional Density Estimation

$$X|Y \sim P(\cdot|Y) \text{ with } dP(x|Y) = p(x|Y)d\lambda$$
$$\longrightarrow \widetilde{X}|Y \sim \widetilde{P}_{\widetilde{\theta}(Y)} \text{ with } d\widetilde{P}_{\theta(Y)}(x) = \widetilde{p}_{\theta(Y)}(x)d\lambda$$

Maximum Likelihood Approach

- Select a family $\widetilde{P}$ and estimate p by $\widetilde{p}_{\widetilde{\theta}}$ from an i.i.d. sample $(X_1, Y_1), \ldots, (X_n, Y_n)$ where $\widetilde{\theta}$ is now a function of Y.
- Simulate $\widetilde{X}|Y$ having a law $\widetilde{P}_{\widetilde{\theta}(Y)}$

- If $\widetilde{p}_{\widetilde{\theta}}$ is *close* from p, the law of $\widetilde{X}|Y$ will be close from the law of $X|Y$.

Issue: How to do it?

- Which family $\widetilde{P}$? Which function family for $\widetilde{\theta}$?
- How to simulate $\widetilde{P}_{\widetilde{\theta}(Y)}$? Parametric? Iterative?

- Corresponds to $\omega \sim \widetilde{Q}(\cdot|Y) = \widetilde{P}_{\widetilde{\theta}(Y)}$ and $\widetilde{X} = G(Y, \omega) = \omega$

Direct Parametric Conditional Density Estimation

$$\omega \sim \widetilde{Q}_{\widetilde{\theta}(Y)} \sim \tilde{q}_{\widetilde{\theta}(Y)}(x)d\lambda \quad \text{and} \quad \widetilde{X}|Y = G(Y, \omega) = \omega$$

Estimation

- By construction,
$$dP(\widetilde{X}|Y) = \tilde{q}_{\tilde{\theta}(y)}(x)d\lambda$$
- Maximum Likelihood approach:
$$\widetilde{\theta} = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} \log \tilde{q}_{\tilde{\theta}(Y_i)}(X_i)$$

Simulation

- $\widetilde{P}$ has been chosen so that this distribution is easy to sample...

- Possible families: Gaussian, Multinomial, Exponential model...
- Possible parametrizations for $\widetilde{\theta}$: linear, neural network...
- Limited expressivity!

Invertible Transform

$$\omega \sim \widetilde{Q}_{\widetilde{\theta}(Y)} \sim \widetilde{q}_{\widetilde{\theta}(Y)}(x)d\lambda \quad \text{and} \quad \widetilde{X}|Y = G(\omega) \text{ with a given } G \text{ invertible.}$$

Estimation

- By construction,
$$d\widetilde{P}\Big(G^{-1}(\widetilde{X})|Y\Big) = \tilde{q}_{\tilde{\theta}(y)}(G^{-1}(x))d\lambda$$
- Maximum Likelihood approach:
$$\widetilde{\theta} = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} \log \tilde{q}_{\tilde{\theta}(Y_i)}(G^{-1}(X_i))$$

Simulation

- $\widetilde{Q}$ has been chosen so that this distribution is easy to sample...
- Possible transform G: Change of basis, known transform...

Flow

$$\omega \sim \widetilde{Q}_{\widetilde{\theta}(Y)} = \widetilde{q}_{\widetilde{\theta}(Y)}(x)d\lambda \quad \text{and} \quad \widetilde{X}|Y = G_{\widetilde{\theta}_G(Y)}(\omega) \text{ with } G_\theta \text{ invertible.}$$

Estimation

- By construction,

$$d\widetilde{P}\Big(\widetilde{X}|Y\Big) = |\mathrm{Jac}G^{-1}_{\widetilde{\theta}_G(Y)}(x)|\tilde{q}_{\tilde{\theta}(Y)}(G^{-1}_{\widetilde{\theta}_G(Y)}(x))d\lambda$$

  where $\mathrm{Jac}G^{-1}_{\theta_G(Y)}(x)$ is the Jacobian of $G^{-1}_{\theta_G(Y)}$ at x
- Maximum Likelihood approach:

$$\widetilde{\theta}, \widetilde{\theta}_G = \underset{\theta,\theta_G}{\mathrm{argmax}} \sum_{i=1}^{n} \Big(\log|\mathrm{Jac}G^{-1}_{\theta_G(Y)}(x)| + \log \tilde{q}_{\theta(Y_i)}(G^{-1}_{\theta_G(Y_i)}(X_i))\Big)$$

Simulation

- $\widetilde{Q}$ has been chosen so that this distribution is easy to sample...

- Often, in practice, $\tilde{\theta}(Y)$ is independent of Y...
- Main issue: G_θ, its inverse and its Jacobian should be easy to compute.

Possible Flows

$$G_\theta?$$

- Main issue: G_θ, its inverse and its Jacobian should be easy to compute.

Flow Models

- Composition

$$G_\theta = G_{\theta_T} \circ G_{\theta_{T-1}} \circ G_{\theta_1} \circ G_{\theta_0}$$

$$|\mathrm{Jac} G_\theta^{-1}| = \prod |\mathrm{Jac} G_{\theta_i}^{-1}|$$

- Real NVP

$$G_\theta(x) = \begin{pmatrix} x_1 \\ \vdots \\ x_{d'} \\ x_{d'+1} e^{s_{d'+1}(x_{1,\dots,d'})} + t_d(x_{1,\dots,d'}) \\ \vdots \\ x_d e^{s_d(x_{1,\dots,d'}} + t_d(x_{1,\dots,d'}) \end{pmatrix} \to G_\theta^{-1}(x) = \begin{pmatrix} x_1 \\ \vdots \\ x_{d'} \\ (x_{d'+1} - t_d(x_{1,\dots,d'})) e^{-s_{d'+1}(x_{1,\dots,d'})} + \\ \vdots \\ (x_d - t_d(x_{1,\dots,d'})) e^{-s_d(x_{1,\dots,d'})} \end{pmatrix} \to |\mathrm{Jac} G(x)^{-1}| = \prod_{d''=d'+1}^{d} e^{-s_{d''}(x_{1,\dots,d'})}$$

- Combined with permutation along dimension or invertible transform across dimension.

- Not that much flexibility...

Factorization

$$\omega_0 \sim \widetilde{Q}_0(\cdot|Y) \text{ and } \widetilde{X}_0 = G_0(\omega_0)$$
$$\omega_{t+1} \sim \widetilde{Q}_{t+1}\Big(\cdot|Y,(\widetilde{X}_l)_{l\leq t}\Big) \text{ and } \widetilde{X}_{t+1} = G_{t+1}(Y,(\widetilde{X}_l)_{l\leq t},\omega_{t+1})$$
$$\widetilde{X} = (\widetilde{X}_0,\ldots,\widetilde{X}_{d-1})$$

Factorization

- Amounts to use a factorized representation
$$\widetilde{P}\Big(\widetilde{X}|Y\Big) = \prod_{0\leq k<d} \widetilde{P}\Big(\widetilde{X}_k|Y,(\widetilde{X}_l)_{l<t}\Big)$$
- $\widetilde{Q}_t$ and G_k can be chosen as in the plain density estimation case as the $X_i^{(t)}$ are observed.

Estimation

- d generative models to estimate instead of one.
- Simple generator by construction.

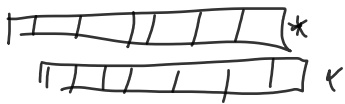
Sequence and Markov Model

$$\omega_{t+1} \sim \widetilde{Q}\Big(\cdot | Y, (\widetilde{X}_l)_{t \geq l \geq t-o}\Big) \text{ and } \widetilde{X}_{t+1} = G(Y, (\widetilde{X}_l)_{t \geq l \geq t-o}, \omega_{t+1})$$
$$\widetilde{X} = (\widetilde{X}_0, \dots, \widetilde{X}_{d-1})$$

Sequence and Markov Models

- Sequence: sequence of *similar* objects with a translation invariant structure.
- Translation invariant probability model of finite order (memory) o.
- Requires an initial padding of the sequence.

- Faster training as the parameters are shared for all t.
- Model used in Text Generation!



Outline

Latent Variable

$$\omega_0 \sim \widetilde{Q}_0(\cdot|Y) \text{ and } \widetilde{X}_0 = G_0(\omega_0)$$
$$\omega_1 \sim \widetilde{Q}_1(\cdot|Y, X_0) \text{ and } \widetilde{X}_1 = G_1(Y, \omega_0)$$
$$\widetilde{X} = \widetilde{X}_1$$

- Most classical example:
  - Gaussian Mixture Model with $\widetilde{X}_0 = \omega_0 \sim \mathcal{M}(\pi)$ and $\widetilde{X} = \omega_1 \sim \mathrm{N}(\mu_{\widetilde{X}_0}, \Sigma_{\widetilde{X}_0})$.

Estimation

- Still a factorized representation

$$\widetilde{P}\left(\widetilde{X}_1, \widetilde{X}_0 | Y\right) = \widetilde{P}_0\left(\widetilde{X}_0 | Y\right) \widetilde{P}_1\left(\widetilde{X}_1 | Y, \widetilde{X}_0\right)$$

  but only $\widetilde{X}_1$ is observed.
- **Much more complex estimation!**

- Simple generator by construction provided that the $\widetilde{Q}_t$ are easy to simulate.

Log Likelihood and ELBO

$$\begin{aligned}\log \widetilde{p}(\widetilde{X}|Y) &= \log \mathbb{E}_{\widetilde{P}(\widetilde{X}_0|Y,\widetilde{X})}\Big[\widetilde{p}(\widetilde{X},\widetilde{X}_0|Y)\Big] \\ &= \sup_{R(\cdot|Y,\widetilde{X}])} \mathbb{E}_{R(\cdot|Y,\widetilde{X})}\Big[\log \widetilde{p}(\widetilde{X},\widetilde{X}_0|Y) - \log r(\widetilde{X}_0|Y,\widetilde{X})\Big]\end{aligned}$$

- Need to integrate over $\widetilde{X}_0$ using the conditional law $\widetilde{P}\Big(\widetilde{X}_0|Y,\widetilde{X}\Big)$, which may be hard to compute.

Evidence Lower BOund

- Using $\log \widetilde{p}(\widetilde{X}|Y) = \mathbb{E}_{R(\cdot|Y,\widetilde{X})}\Big[\log\Big(\widetilde{p}(\widetilde{X},\widetilde{X}_0|Y)/\widetilde{p}(\widetilde{X}_0|Y,\widetilde{X})\Big)\Big]$,

$$\begin{aligned}\log \widetilde{p}(\widetilde{X}|Y) = \mathbb{E}_{R(\cdot|Y,\widetilde{X})}\Big[&\log \widetilde{p}(\widetilde{X},\widetilde{X}_0|Y) - \log r(\widetilde{X}_0|Y,\widetilde{X})\Big] \\ &- \mathrm{KL}_{\widetilde{X}_0}(R(\widetilde{X}_0|Y,\widetilde{X}),\widetilde{P}\Big(\widetilde{X}_0|Y,\widetilde{X}\Big))\end{aligned}$$

- Equality is obtained for $R(\cdot|Y,\widetilde{X}) = \widetilde{P}\Big(\widetilde{X}_0|Y,\widetilde{X}\Big)$.

- Maximization over $\widetilde{P}$ and R instead of only over $\widetilde{P}$...

ELBO and Stochastic Gradient Descent

$$\sup_{\widetilde{P}} \mathbb{E}_{\widetilde{X},Y}\Big[\log \widetilde{p}(\widetilde{X}|Y)\Big] = \sup_{\widetilde{P},R} \mathbb{E}_{\widetilde{X},Y,\widetilde{X}_0\sim R(\cdot|Y,\widetilde{X})}\Big[\log \widetilde{p}(\widetilde{X},\widetilde{X}_0|Y) - \log r(\widetilde{X}_0|Y,\widetilde{X})\Big]$$

$$= \sup_{\widetilde{P},R} \mathbb{E}_{\widetilde{X},Y,\widetilde{X}_0\sim R(\cdot|Y,\widetilde{X})}\Big[\log \widetilde{p}(\widetilde{X}|Y,\widetilde{X}_0)\Big] + \underbrace{\mathbb{E}_{\widetilde{X},Y,\widetilde{X}_0\sim R(\cdot|Y,\widetilde{X})}\Big[\log \widetilde{p}(\widetilde{X}_0|Y) - \log r(\widetilde{X}_0|Y,\widetilde{X})\Big]}_{\mathbb{E}_{\widetilde{X},Y}\left[\mathrm{KL}(R(\cdot|Y,\widetilde{X}),\widetilde{P}(\widetilde{X}_0|Y))\right]}$$

- Parametric models for $\widetilde{P}(X_0|Y)$, $\widetilde{P}(X|Y,X_0)$ and $R(\widetilde{X}_0|\widetilde{X},Y)$.

Stochastic Gradient Descent

- Sampling on $(X,Y,\widetilde{X}_0 \sim R)$ for $\mathbb{E}_{\widetilde{X},Y,\widetilde{X}_0\sim R(\cdot|Y,\widetilde{X})}\Big[\nabla \log \widetilde{p}(\widetilde{X}|Y,\widetilde{X}_0)\Big]$
- Sampling on (X,Y) for $\mathbb{E}_{\widetilde{X},Y}\Big[\nabla \mathrm{KL}(R(\cdot|Y,\widetilde{X}),\widetilde{P}(\widetilde{X}_0|Y))\Big]$ if closed formula.
- Reparametrization trick for the second term otherwise...

Reparametrization Trick

$Z = G(\omega)$ with $\omega \sim Q(\cdot)$ fixed $\longrightarrow \nabla \mathbb{E}_Z[F(Z)] = \nabla \mathbb{E}_\omega[F(G(\omega))] = \mathbb{E}_\omega[\nabla(F \circ G)(\omega)]$

Reparametrization Trick

- Define a random variable Z as the image by a parametric map G of a random variable ω of fixed distribution Q.
- Most classical case: Gaussian...
- Allow to compute the derivative the expectation of a function of Z through a sampling of ω.

- Application for ELBO:
  - $\widetilde{X}_0 = G_R(Y, \widetilde{X}, \omega_R)$ with $\omega_R \sim Q(\cdot|Y, \widetilde{X})$ a fixed probability law.
  - Sampling on ω to approximate:

$$\nabla \mathbb{E}_{\widetilde{X},Y,\widetilde{X}_0 \sim R(\cdot|Y,\widetilde{X})} \left[\log \widetilde{p}(\widetilde{X}_0|Y) - \log r(\widetilde{X}_0|Y,\widetilde{X}) \right]$$

$$= \mathbb{E}_{\widetilde{X},Y,\omega_0 \sim Q(\cdot|Y,\widetilde{X})} \left[\nabla \log \widetilde{p}(G_R(Y,\widetilde{X},\omega_0)|Y) - \nabla \log r(G_R(Y,\widetilde{X},\omega_0)|Y,\widetilde{X}) \right]$$

$$z \sim \mathcal{N}(\mu, \sigma^2) \qquad z = \mu + \sigma^2 \tilde{z} \qquad \tilde{z} \sim \mathcal{N}(0, 1)$$

$$E_z[\ell(z)] = E_{\tilde{z}}[\ell(\mu + \sigma^2 \tilde{z})] \qquad E[g(w)] \simeq \frac{1}{m} \sum g(w_i)$$

$$\nabla_\mu E_z[\ell(z)] = \nabla_\mu E_{\tilde{z}}[\ell(\mu + \sigma^2 \tilde{z})]$$

$$\nabla_\mu \left[\int \ell(x) \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx \right] = E_{\tilde{z}}\left[\nabla_\mu \ell(\mu + \sigma^2 \tilde{z}) \right]$$

$$\nabla_\mu = \int +2(x-\mu)\, \ell(x) \frac{1}{\sqrt{2\pi}} e^{\frac{(x-\mu)^2}{2\sigma^2}} dx \qquad \simeq \frac{1}{m} \sum \nabla_\mu \ell(\mu + \sigma^2 \tilde{z}_i)$$

Variational Auto Encoder

Generation: $\widetilde{X}_0 \sim \widetilde{P}(\cdot|Y) \xrightarrow{\text{decoder}} \widetilde{X} \sim \widetilde{P}(\cdot|Y, X_0))$

Training: $X \sim P(\cdot|Y) \xrightarrow{\text{encoder}} X_0 \sim R(\cdot|Y, X) \xrightarrow{\text{decoder}} \widetilde{X} \sim \widetilde{P}(\cdot|Y, X_0)$

Variational Auto Encoder

- Training structure similar to classical autoencoder... but matching on distributions rather than samples.
- Encoder interpretation of the approximate posterior $R(\cdot|Y, X)$.
- Implicit *low* dimension for X_0.

Latent Variables

$$\omega_0 \sim \widetilde{Q}_0(\cdot|Y) \text{ and } \widetilde{X}_0 = G_0(\omega_0)$$
$$\omega_{t+1} \sim \widetilde{Q}_{t+1}(\cdot|Y, X_k) \text{ and } \widetilde{X}_{t+1} = G_{t+1}(Y, \widetilde{X}_t, \omega_{t+1})$$
$$\widetilde{X} = \widetilde{X}_T$$

Latent Variables

- Deeper hierachy is possible...
- ELBO scheme still applicable using *decoders* R_i

$$R_i(\widetilde{X}_i|Y, \widetilde{X}_{i+1}) \simeq \widetilde{P}\Big(\widetilde{X}_i|Y, \widetilde{X}_{i+1}\Big)$$

Outline

Energy Based Model and MCMC Simulator

$$d\widetilde{P}\big(\widetilde{X}|Y\big) \propto e^{u(\widetilde{X},Y)} d\lambda$$

$$\longrightarrow \omega_{t+1} \sim \widetilde{Q}_u\big(\cdot|Y,\widetilde{X}_t\big) \text{ and } \widetilde{X}_{t+1} = G_u(Y, X_t, \omega_{t+1})$$

$$\widetilde{X} \simeq \lim \widetilde{X}_t$$

- Explicit conditional density model up to normalizing constant

$$Z(Y) = \int e^{u(x,Y)} d\lambda(x)$$

Simulation

- Several MCMC schemes to simulate the law without knowing $Z(Y)$

Estimation

- Not so easy as $Z(Y)$ depends a lot on u itself.

MCMC Simulation - Metropolis-Hastings

$$X_{t+1/2} \sim \widetilde{Q}_u\left(\cdot|Y, \widetilde{X}_t\right)$$

$$X_{t+1} = \begin{cases} X_{t+1/2} & \text{with proba } \alpha_t \\ X_t & \text{with proba } 1-\alpha_t \end{cases} \quad \text{with } \alpha_t = \min\left(1, \frac{e^{u(X_{t+1/2},Y)}\widetilde{Q}_u\left(\widetilde{X}_t|Y, \widetilde{X}_{t+1/2}\right)}{e^{u(X_t,Y)}\widetilde{Q}_u\left(\widetilde{X}_{t+1/2}|Y, \widetilde{X}_t\right)}\right)$$

Metropolis Hastings

- Most classical algorithm.
- Convergence guarantee under reversibility of the proposal.
- Main issue is the choice of this proposal $\widetilde{Q}$.

- Many enhanced versions exist!

Monte Carlo Markov Chain

MCMC Simulation - Langevin

$$\widetilde{X}_{t+1/2} = X_t + \gamma_t \nabla_{\widetilde{X}} u(\widetilde{X}_t, Y) + \sqrt{2\gamma_t}\omega_t \text{ with } \omega_t \sim \mathrm{N}(0, \mathrm{Id})$$

$$\widetilde{X}_{t+1} = \begin{cases} \widetilde{X}_{t+1/2} & \text{with proba } \alpha_t \\ \widetilde{X}_t & \text{with proba } 1-\alpha_t \end{cases} \quad \text{with } \alpha_t = \min\left(1, \frac{e^{u(\widetilde{X}_{t+1/2}, Y)}\widetilde{Q}_u\left(\widetilde{X}_t | Y, \widetilde{X}_{t+1/2}\right)}{e^{u(\widetilde{X}_t, Y)}\widetilde{Q}_u\left(\widetilde{X}_{t+1/2} | Y, \widetilde{X}_t\right)}\right)$$

- Connection with a SDE:

$$\frac{d\widetilde{X}}{dt} = \nabla_{\widetilde{X}} u(\widetilde{X}, Y) + \sqrt{2} dB_t$$

where B_t is a Brownian Motion.

Langevin

- If $\gamma_t = \gamma$, Metropolis-Hasting algorithm.
- With $\widetilde{X}_{t+1} = \widetilde{X}_{t+1/2}$, convergence toward an approximation of the law.
- Connection with SGD with decaying α_t

EBM Estimation

$$X|Y \sim P(\cdot|Y) \longrightarrow \widetilde{X}|Y \sim \widetilde{P}(\cdot|Y) \text{ with } d\widetilde{P}(x|Y) = \widetilde{p}(x|Y)d\lambda \propto e^{u(x,Y)}d\lambda$$

- Intractable log-likelihood:

$$\log \widetilde{p}(\widetilde{x}|Y) = u(\widetilde{x}, Y) - \log Z(u)$$

Estimation

- Contrastive: simulate some $\widetilde{P}$ at each step and use

$$\nabla \log \widetilde{p}(\widetilde{x}|Y) = \nabla u(\widetilde{x}, Y) - \nabla \log Z(u) = \nabla u(\widetilde{x}, Y) - \mathbb{E}_{\widetilde{P}}\left[\nabla u(\widetilde{X}, Y)\right]$$

- Noise contrastive: learn to discriminate $W = X$ from $W = X' \sim R(\cdot|Y) \sim e^{r(x,Y)d\lambda}$ with the parametric approximation

$$\mathbb{P}(W = X|Y) \simeq \frac{e^{u(x,Y)}}{e^{u(x,Y)} + Z(Y)e^{r(x,Y)}}$$

- Score based: learn directly $s(\cdot|Y) = \nabla_{\widetilde{X}} u(\cdot|Y) \simeq \nabla_X \log p(\cdot|Y)$.

Score Based Method

$$\mathbb{E}\Big[\|\nabla_X \log p(X|Y) - s(X|Y)\|^2\Big] = \mathbb{E}\Big[\frac{1}{2}\|s(x|Y)\|^2 + \mathrm{tr}\,\nabla_X s(X,Y)\Big] + \mathrm{cst}.$$

Score Based Method

- Non trivial formula based on partial integration.
- Hard to use in high dimension

$$X_\sigma = X + \sigma\epsilon \longrightarrow \mathbb{E}\Big[\|\nabla_X \log p_\sigma(X_\sigma|Y) - s_\sigma(W,Y)\|^2\Big]$$
$$= \mathbb{E}\Big[\|\|\nabla_X \log p_\sigma(X_\sigma|X,Y) - s_\sigma(X_\sigma,Y)\|^2\Big] + \mathrm{cst}.$$

Noisy Score

- Connection to denoising through Tweedie formula

$$\mathbb{E}[X|X_\sigma] = X_\sigma + \sigma^2\nabla_X \log p_\sigma(X_\sigma|X,Y) \text{ and thus } s_\sigma(X_\sigma,Y) \simeq \frac{\mathbb{E}[X|X_\sigma] - X_\sigma}{\sigma^2}$$

Better Exploration with Annealing and Noisy Score

$$\widetilde{X} \sim e^{u(X,Y)} d\lambda \longrightarrow \widetilde{X}_T \sim e^{\frac{1}{T} u(X,Y)}$$

Annealing

- Simulate a sequence of $\widetilde{X}_T$ starting with T large and decaying to 1.

$$\begin{aligned} X_\sigma = X + \sigma\epsilon \longrightarrow \mathbb{E}\Big[\|\nabla_X \log p_\sigma(X_\sigma|Y) - s_\sigma(W, Y)\|^2\Big] \\ = \mathbb{E}\Big[\|\|\nabla_X \log p_\sigma(X_\sigma|X, Y) - s_\sigma(X_\sigma, Y)\|^2\Big] + \text{cst}. \end{aligned}$$

Noisy Score

- Simulate a noiy sequence of $\widetilde{X}_\sigma$ with σ decaying to 0.

Outline

Noisy Model: Forward and Reverse Views

Forward: $X_{t+1} = X_t + \sigma_t \omega_t \to X_{t+1}|X_0 \sim \mathrm{N}(X_0, s_t^2 = \sum_{t' \leq t} \sigma_{t'}^2)$

Reverse: $\widetilde{X}_T \sim \mathrm{N}(0, s_T^2) \to \widetilde{X}_{t-1} = \widetilde{X}_t + \gamma_t u_{s_t^2}(X_t) + \sqrt{2\gamma_t}\omega_t$

Noisy Model

- Construct a sequence of noisy version X_t.
- Each X_t is easily sampled from X_0 so that the scores $u_{s_t^2}$ can be estimated.
- Use a simplistic sequential Langevin approach to obtain $\widetilde{X}_0 \sim \widetilde{P}(X)$ from $\widetilde{X}_T \sim \mathrm{N}(0, s_T^2)$.

- Lot of approximations everywhere.
- Dependency on Y has been removed for sake of simplicity.

Diffusion with a Forward Point of View

Forward: $$X_{t+\delta_t} = (1 + \alpha_t \delta_t) X_t + \sqrt{2\beta_t \delta_t}\omega_t$$

$$\longrightarrow dX(t) = \alpha(t)X(t)dt + \sqrt{2\beta(t)}dB(t)$$

Forward diffusion

- Generalization of noisy model:

$$X(t)|X(0) = \mathrm{N}\left(X(0)\exp\int_0^t \alpha(u)du, \int_0^t 2\beta(u)\exp\left(\int_u^t \alpha(v)dvdu\right)\right)$$

Reverse: $$dX(t) = (-2\beta(t)\nabla_X \log P(X,t) - \alpha(t)X(t))\,\overline{dt} + \sqrt{2\beta(t)}\overline{dB}(t)$$

$$\longrightarrow X_{t-\delta_t} = (1 - \alpha_t\delta_t)X_t + 2\beta_t \nabla_X \log p(X,t)\delta_t + \sqrt{2\beta_t\delta_t}\omega_t$$

Reverse diffusion

- Allow to sample back in time $X_t|X_T$.
- Quite involved derivation... but Langevin type scheme in the end.

Noise Conditioned Score and Denoising Diffusion

$$\alpha_t = 0 \rightarrow \alpha(t) = 0 \rightarrow X(t)|X(0) = \mathrm{N}\left(X(0), 2\int_0^t \beta(u)du\right)$$

Noise Conditioned Score

- Direct extension of noisy model.
- Better numerical scheme but numerical explosion for $X(t)$.

$$(1 + \alpha_t\delta_t) = \sqrt{1 - 2\beta_t\delta_t} \simeq 1 - \beta_t\delta_t$$
$$\longrightarrow X(t)|X(0) = \mathrm{N}\left(X(0)\int_0^t e^{-\beta(u)}du, 2\left(1 - \int_0^t e^{-\beta(u)}\right)\right)$$

Denoising Diffusion Probalilistic Model

- Explicit decay of the dependency on $P(X)$ and control on the variance.
- Better numerical results.

- Scores $\nabla_X \log p(X, t)$ estimated using the denoising trick as $X(t)|X(0)$ is explicit.
- Choice of $\beta(t)$ has a numerical impact.

A Possible Shortcut ?

Forward (SDE): $dX(t) = \alpha(t)X(t)dt + \sqrt{2\beta(t)}dB_t$

Backward (ODE): $dX(t) = (-2\beta(t)\nabla_X \log P(X,t) - \alpha(t)X(t))\,\overline{dt}$

Deterministic Reverse Equation

- If $X(T)$ is initialized with the law resulting from the forward distribution, the marginal of the reverse diffusion are the right ones.
- No claim on the trajectories... but irrelevant in the generative setting.
- Much faster numerical scheme...

- Stability results on the score estimation error and the numerical scheme exist for both the stochastic and deterministic case.

Connection between Diffusion and VAE

$$X \sim P \underset{P(X|X_1)}{\overset{Q(X_1|X)}{\rightleftharpoons}} X_1 \underset{P(X_1|X_2)}{\overset{Q(X_2|X_1)}{\rightleftharpoons}} X_1 \dots \underset{P(X_t|X_{t+1})}{\overset{Q(X_{t+1}|X_t)}{\rightleftharpoons}} \dots X_{T-1} \underset{P(X_{T-1}|X_T)}{\overset{Q(X_T|X_{T-1})}{\rightleftharpoons}} X_T \sim P_T$$

- Generation of X from X_T using $P(X_t|X_{t+1})$.
- Encoder/Forward diffusion: $Q(X_{t+1}|X_t)$.

Variationnal Auto-Encoder

- P_T is chosen as Gaussian.
- Both generative $P(X_t|X_{t+1})$ and *encoder* $Q(X_{t+1}|X_t)$ have to be learned.

Approximated Diffusion Model

- $Q(X_{t+1}|X_t)$ is known and P_T is approximately Gaussian.
- Generative $P(X_t|X_{t+1})$ has to be learned.

- Denoising tricks can be obtained as an ELBO starting from $Q(X_{t+1|X_t}) = Q(X_{t+1}|X_t, X)\dots$

Outline

Generative Adversarial Network

$$\omega \sim \widetilde{Q}(\cdot|Y) \text{ and } \widetilde{X} = G(Y, \omega)$$

Non density based approach

- Can we optimize G without thinking in term of density (or score)?

$$(\overline{X}, Z, Y) = \begin{cases} (X, 1, Y) & \text{with proba } 1/2 \\ (G(Y, \omega), 0, Y) & \text{otherwise} \end{cases}$$

GAN Approach

- Can we guess Z with a discriminator $D(\overline{X}, Y)$?
- No if G is perfect!

GAN Program

$$\max_G \min_D \mathbb{E}\Big[\overline{X}, Y\Big] \ell(D(\overline{X}, Y)$$
$$= \max_G \min_D \left(\frac{1}{2}\mathbb{E}_{X,Y}[\ell(D(X,Y),1)] + \frac{1}{2}\mathbb{E}_{\omega}[\ell(D(G(Y,\omega),Y),0)]\right)$$

Discrimination

- Similar idea than the *noise* contrastive approach in EBM.
- If ℓ is a convexification of the $\ell^{0/1}$ loss then the optimal classifier is given by
$$D(\overline{X}|Y) = \begin{cases} 1 & \text{if } p(\overline{X}|Y) > \tilde{p}(\overline{X}|Y) \\ 0 & \text{otherwise.} \end{cases}$$
- If ℓ is the log-likelihood
$$\max_G \min_D \mathbb{E}\Big[\overline{X}, Y\Big] \ell(D(\overline{X}, Y) = \max_G \log_2 -\mathbb{E}_Y\Big[JKL_{1/2}(p(\cdot|Y), \tilde{p}(\dot{|}Y))\Big]$$
- Direct (approximate) optimization using only samples (with the reparametrization trick).

Extensions to f Divergences

$$D_f(P,Q) = \int f\left(\frac{p(x)}{q(x)}\right) q(x)$$
$$= sup_T \mathbb{E}_{\underline{X}\sim P}[T(\underline{X})] - \mathbb{E}_{G\sim Q}[f^\star(T(G))]$$

f-GAN

- Optimization of
$$\min_G \sup_T \left(\mathbb{E}_{X,Y}[T(\underline{X})] - \mathbb{E}_{\omega,Y}[f^\star(T(G(Y,\omega)))]\right)$$
- Direct (approximate) optimization using only samples (with the reparametrization trick).

- Direct extension of the previous scheme.
- T is not a discriminator but there is an explicit link when $f(u) = \log(u)$.

Wasserstein GAN

$$W(P,Q) = \inf_{\xi \in \pi(P,Q)} \mathbb{E}_{(p,q)\sim\xi}[\|p-q\|]$$
$$= \frac{1}{K} sup_{\|f\|_L \leq K} \mathbb{E}_{\underline{X}\sim P}[f(\underline{X})] - \mathbb{E}_{G\sim Q}[f(G))]$$

Wasserstein GAN

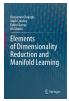
- Optimization of
$$\min_G \sup_{\|f\|_L \leq 1} \mathbb{E}_{\underline{X}\sim P}[f(\underline{X})] - \mathbb{E}_Z[f(G(Z))]$$
- Direct (approximate) optimization using only samples (with the reparametrization trick).

- More stability but hard to optimize on all the 1-Lipschitz functions.

Outline

References

F. Husson, S. Le, and J. Pagès.
Exploratory Multivariate Analysis by Example Using R (2nd ed.)
Chapman and Hall/CRC, 2017



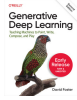
B. Ghojogh, M. Crowley, F. Karray, and A. Ghodsi.
Elements of Dimensionality Reduction and Manifold Learning.
Springer, 2023

Ch. Aggarwal and Ch. Reddy.
Data Clustering: Algorithms and Applications.
Chapman and Hall/CRC, 2013

Ch. Hennig, M. Meila, F. Murtagh, and R. Rocci.
Handbook of Cluster Analysis.
Chapman and Hall/CRC, 2015

Ch. Bouveyron, G. Celeux, B. Murphy, and A. Raftery.
Model-Based Clustering and Classification for Data Science.
Cambridge University Press, 2019

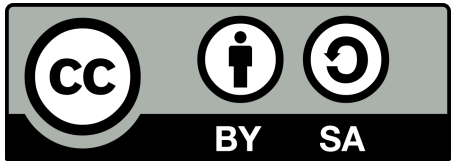
J. Tomczak.
Deep Generative Modeling.
Springer, 2021



D. Foster.
Generative Deep Learning (2nd ed.)
O'Reilly, 2023

A. Géron.
Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow (3rd ed.)
O'Reilly, 2022

Licence and Contributors



Contributors

- Main contributor: E. Le Pennec
- Contributors: S. Boucheron, A. Dieuleveut, A.K. Fermin, S. Gadat, S. Gaiffas, A. Guilloux, Ch. Keribin, E. Matzner, M. Sangnier, E. Scornet.