

IFT 4031/7031,
Machine Learning for Signal Processing
**Week4: Machine Learning 1,
Decompositions**

Cem Subakan



UNIVERSITÉ
LAVAL



- Did you have a chance to read the project proposal document?
 - ▶ Avez-vous regardé le document sur les proposals de projets?
- The first homework will be released soon!
 - ▶ Le devoir 1 va sortir bientôt!

- Did you have a chance to read the project proposal document?
 - ▶ Avez-vous regardé le document sur les proposals de projets?
- The first homework will be released soon!
 - ▶ Le devoir 1 va sortir bientôt!
- Today: We are starting with machine learning.
 - ▶ Aujourd'hui on commence avec l'apprentissage automatique.

This week

- Today, our aim is to build the foundation for training machine learning models.
 - ▶ Au'jourd'hui le but est de batir le fondation pour l'entrainement des modèles.

This week

- Today, our aim is to build the foundation for training machine learning models.
 - ▶ Au'jourd'hui le but est de batir le fondation pour l'entrainement des modèles.
- More specifically, we will build a framework around learnable decompositions.
 - ▶ Plus spécifiquement on va batir un framework autour des decompositions appris.

Table of Contents

The Decomposition Framework

Fixed Basis Decompositions (Linear Regression)

Learnable Basis Decompositions

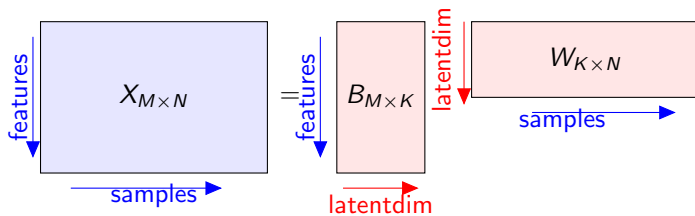
- Principal Component Analysis

- Independent Component Analysis

- Non-Negative Matrix Factorization (NMF)

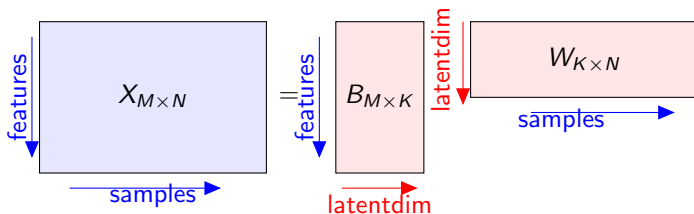
The framework

$$\underbrace{X}_{\text{Data Matrix}} = \underbrace{B}_{\text{Bases}} \times \underbrace{W}_{\text{Activations (Embeddings)}}$$



The framework

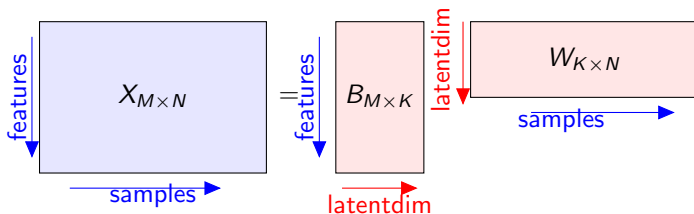
$$\underbrace{X}_{\text{Data Matrix}} = \underbrace{B}_{\text{Bases}} \times \underbrace{W}_{\text{Activations (Embeddings)}}$$



- Note that this framework embeds M dimensional data in K dimensions.
 - Notez qu'on est en train de trouver un embedding de K dimensions pour un data qui a M dimensions.

The framework

$$\underbrace{X}_{\text{Data Matrix}} = \underbrace{B}_{\text{Bases}} \times \underbrace{W}_{\text{Activations (Embeddings)}}$$



- Note that this framework embeds M dimensional data in K dimensions.
 - ▶ Notez qu'on est en train de trouver un embedding de K dimensions pour un data qui a M dimensions.
- We embed X , in the space defined by the columns of B .
 - ▶ On embed X dans une espace definit par les colonnes de B .

Table of Contents

The Decomposition Framework

Fixed Basis Decompositions (Linear Regression)

Learnable Basis Decompositions

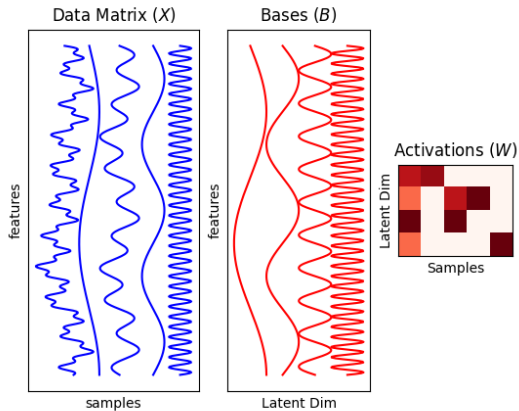
- Principal Component Analysis

- Independent Component Analysis

- Non-Negative Matrix Factorization (NMF)

Example

- Remember this from last week?
 - ▶ Vous-vous en souvenez ça de la semaine passée?



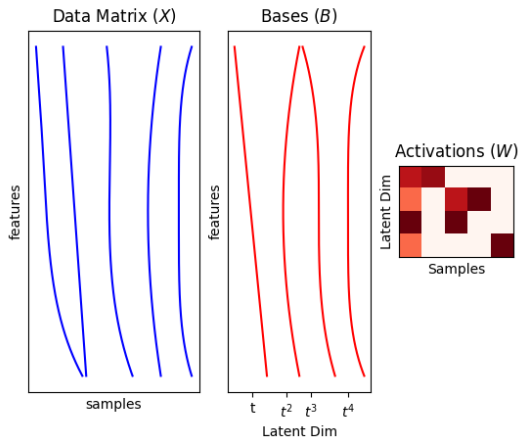
The goal

- We are trying to build a framework that can effectively reduce dimensionality, to explain data in a concise way.
 - ▶ On essaie de bâtir un framework qui peut effectivement réduire la dimensionnalité et expliquer les données de manière parsimonieux.

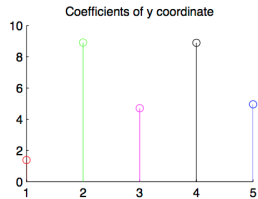
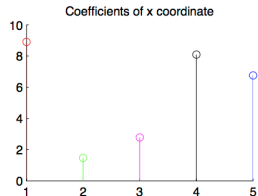
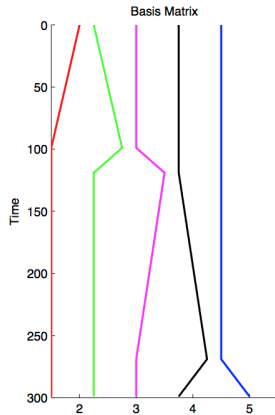
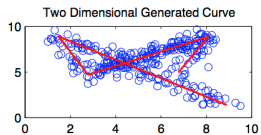
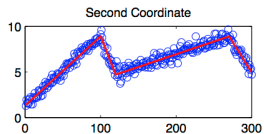
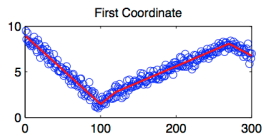
The goal

- We are trying to build a framework that can effectively reduce dimensionality, to explain data in a concise way.
 - ▶ On essaie de bâtir un framework qui peut effectivement réduire la dimensionnalité et expliquer les données de manière parsimonieuse.
- We can use basis functions other than sinusoids!
 - ▶ On peut utiliser des bases autres que les sinusoids!

Non-sinusoids (finally)

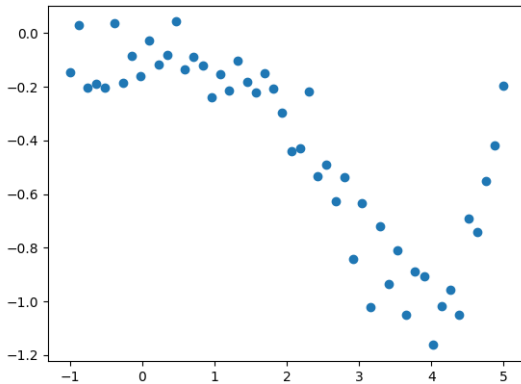


Piece-wise functions!!



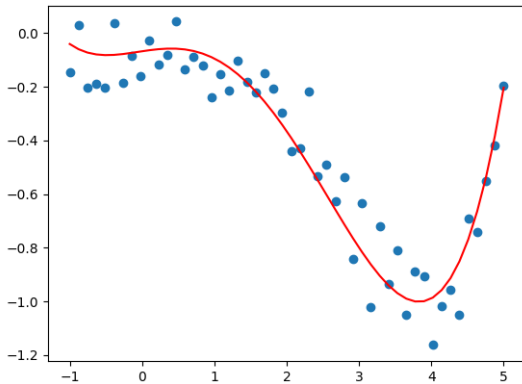
But why, I still don't get it?

- One application might be do to regression.
 - ▶ On peut faire de la regression avec ce framework.



But why, I still don't get it?

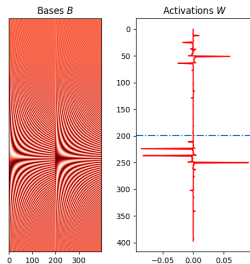
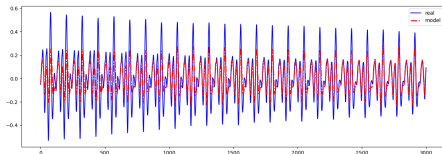
- One application might be do to regression.
 - ▶ On peut faire de la regression avec ce framework.



Something a bit more real

■ Modeling a guitar string Listen Real, Listen the Model

► Modélisons un corde de guitare



Visualizing the model ingredients

$$\underbrace{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_T \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} b_1(1) & b_2(1) & \dots & b_K(1) \\ b_1(2) & b_2(2) & \dots & b_K(2) \\ \vdots & \vdots & \ddots & \vdots \\ b_1(T) & b_2(T) & \dots & b_K(T) \end{bmatrix}}_B \underbrace{\begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_K \end{bmatrix}}_{\mathbf{w}}$$

- $b_k(t)$ is the k 'th basis function in the basis (design) matrix B .
 - $b_k(t)$ est la fonction de base k 'eme dans la matrice de base.
- The output is a linear combination of the basis functions such that
 - La sortie du modèle est la combinaison linéaire des bases:

$$x_t = \sum_{k=1}^K w_k b_k(t) = w_1 b_1(t) + w_2 b_2(t) + \dots + w_K b_K(t).$$

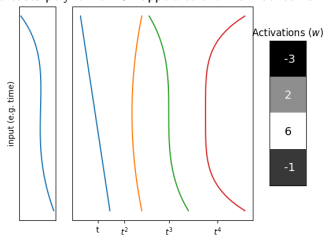
Visualizing the model ingredients

$$\underbrace{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_T \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} b_1(1) & b_2(1) & \dots & b_K(1) \\ b_1(2) & b_2(2) & \dots & b_K(2) \\ \vdots & \vdots & \ddots & \vdots \\ b_1(T) & b_2(T) & \dots & b_K(T) \end{bmatrix}}_{\mathbf{B}} \underbrace{\begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_K \end{bmatrix}}_{\mathbf{w}}$$

- $b_k(t)$ is the k 'th basis function in the basis (design) matrix $\mathbf{B}$.
 - ▶ $b_k(t)$ est la fonction de base k 'eme dans la matrice de base.
- The output is a linear combination of the basis functions such that
 - ▶ La sortie du modèle est la combinaison linéaire des bases:

$$x_t = \sum_{k=1}^K w_k b_k(t) = w_1 b_1(t) + w_2 b_2(t) + \dots + w_K b_K(t).$$

- Here's an example design matrix with polynomial basis functions. This particular choice is also called a Vandermonde matrix.
 - ▶ Voici un matrice de desin exemplaire avec fonctions de bases polynomial. On appelle ce choix la matrice Vandermonde.



Even autoregressive modeling

Autoregressive Modeling

$$\underbrace{\begin{bmatrix} x_{K+1} \\ x_{K+2} \\ \vdots \\ x_T \end{bmatrix}}_y = \underbrace{\begin{bmatrix} x_K & x_{K-1} & \dots & x_2 & x_1 \\ x_{K+1} & x_K & \dots & x_3 & x_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{t-2} & x_{t-3} & \dots & x_{t-K+2} & x_{t-K+1} \\ x_{t-1} & x_{t-2} & \dots & x_{t-K+1} & x_{t-K} \end{bmatrix}}_B \underbrace{\begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_K \end{bmatrix}}_w$$

Even autoregressive modeling

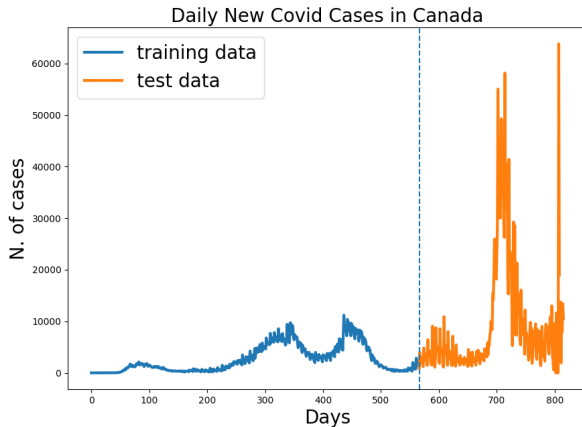
Autoregressive Modeling **LLM: Linear Language Model** (I am joking)

$$\underbrace{\begin{bmatrix} x_{K+1} \\ x_{K+2} \\ \vdots \\ x_T \end{bmatrix}}_y = \underbrace{\begin{bmatrix} x_K & x_{K-1} & \dots & x_2 & x_1 \\ x_{K+1} & x_K & \dots & x_3 & x_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{t-2} & x_{t-3} & \dots & x_{t-K+2} & x_{t-K+1} \\ x_{t-1} & x_{t-2} & \dots & x_{t-K+1} & x_{t-K} \end{bmatrix}}_B \underbrace{\begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_K \end{bmatrix}}_w$$

Btw, do see that we have a series of convolutions? / En passant, vous voyez vous qu'on fait des convolutions?

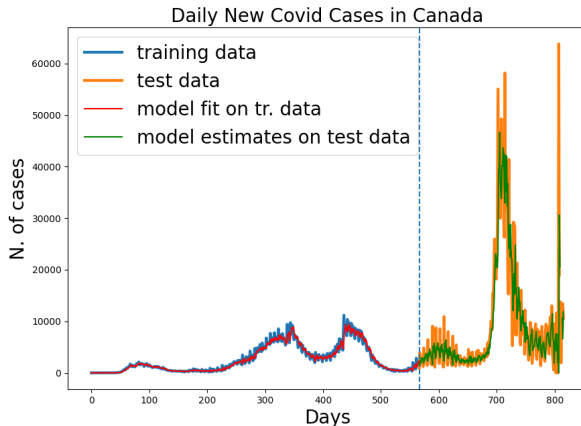
Real Real-life data

- We try to fit a regression model with autoregressive design matrix on nbr. of cases data with $K = 3$. / On utilise un matrice de design qui est autoregressive. On utilise un filtre de $K = 3$.



Real Real-life data

- We try to fit a regression model with autoregressive design matrix on nbr. of cases data with $K = 3$. / On utilise un matrice de design qui est autoregressive. On utilise un filtre de $K = 3$.



How to learn W ?

- Consider the following model (Linear Regression):
 - Considérons la modèle suivante (Regression Linéaire):

$$w_n \sim \mathcal{N}(\mathbf{w}_n; 0, \sigma_0^2 I)$$

$$x_{t,n} | w_n \sim \mathcal{N}(x_t; B(t)\mathbf{w}_n, \sigma^2 I)$$

n is the signal index, t is the time index / n est l'indice du temps, t est l'indice du data.

How to learn W ?

- Consider the following model (Linear Regression):

- Considérons la modèle suivante (Regression Linéaire):

$$\begin{aligned}w_n &\sim \mathcal{N}(\mathbf{w}_n; 0, \sigma_0^2 I) \\x_{t,n} | w_n &\sim \mathcal{N}(x_t; B(t)\mathbf{w}_n, \sigma^2 I)\end{aligned}$$

n is the signal index, t is the time index / n est l'indice du temps, t est l'indice du data.

- Let's write the model likelihood / Écrivons le likelihood du modèle,

$$\begin{aligned}\mathcal{L} &:= \log p(x_{1:T,n}, \mathbf{w}_n) = \sum_t \log p(x_{t,n} | \mathbf{w}_n) + \log p(\mathbf{w}_n) \\&= \sum_t \log \mathcal{N}(x_t; B(t)\mathbf{w}_n, \sigma^2 I) + \log \mathcal{N}(\mathbf{w}_n; 0, \sigma_0^2 I)\end{aligned}$$

How to learn W ?

- Consider the following model (Linear Regression):

- ▶ Considérons la modèle suivante (Regression Linéaire):

$$\begin{aligned}w_n &\sim \mathcal{N}(\mathbf{w}_n; 0, \sigma_0^2 I) \\x_{t,n} | w_n &\sim \mathcal{N}(x_t; B(t)\mathbf{w}_n, \sigma^2 I)\end{aligned}$$

n is the signal index, t is the time index / n est l'indice du temps, t est l'indice du data.

- Let's write the model likelihood / Écrivons le likelihood du modèle,

$$\begin{aligned}\mathcal{L} &:= \log p(x_{1:T,n}, \mathbf{w}_n) = \sum_t \log p(x_{t,n} | \mathbf{w}_n) + \log p(\mathbf{w}_n) \\&= \sum_t \log \mathcal{N}(x_t; B(t)\mathbf{w}_n, \sigma^2 I) + \log \mathcal{N}(\mathbf{w}_n; 0, \sigma_0^2 I)\end{aligned}$$

- What should we do next to estimate $\mathbf{w}_{1:N}$?

- ▶ Qu'est qu'on fait maintenant pour estimer $\mathbf{w}_{1:N}$?

Finding the best $\mathbf{w}_n$

- Now, we will take the gradient of $\mathcal{L}$ with respect to $\mathbf{w}_n$, set it equal to zero and solve for w_n . We switch to matrix-vector notation, and drop the n index to reduce clutter.

$$\begin{aligned}\mathcal{L} &\propto (\mathbf{B}\mathbf{w} - \mathbf{x})^\top (\mathbf{B}\mathbf{w} - \mathbf{x}) \\ &= -\frac{1}{2\sigma^2} \left(\mathbf{w}^\top \mathbf{B}^\top \mathbf{B} \mathbf{w} - 2\mathbf{w}^\top \mathbf{B}^\top \mathbf{x} + \mathbf{x}^\top \mathbf{x} \right) - \frac{1}{2\sigma_0^2} \left(\mathbf{w}^\top \mathbf{w} \right)\end{aligned}$$

Finding the best $\mathbf{w}_n$

- Now, we will take the gradient of $\mathcal{L}$ with respect to $\mathbf{w}_n$, set it equal to zero and solve for w_n . We switch to matrix-vector notation, and drop the n index to reduce clutter.

$$\begin{aligned}\mathcal{L} &\propto (\mathbf{B}\mathbf{w} - \mathbf{x})^\top (\mathbf{B}\mathbf{w} - \mathbf{x}) \\ &= -\frac{1}{2\sigma^2} \left(\mathbf{w}^\top \mathbf{B}^\top \mathbf{B} \mathbf{w} - 2\mathbf{w}^\top \mathbf{B}^\top \mathbf{x} + \mathbf{x}^\top \mathbf{x} \right) - \frac{1}{2\sigma_0^2} \left(\mathbf{w}^\top \mathbf{w} \right)\end{aligned}$$

- And the gradient,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = -\frac{1}{\sigma^2} \left(\mathbf{B}^\top \mathbf{B} \mathbf{w} - \mathbf{B}^\top \mathbf{x} \right) - \frac{1}{\sigma_0^2} \mathbf{w} \quad (1)$$

Finding the best $\mathbf{w}_n$

- Now, we will take the gradient of $\mathcal{L}$ with respect to $\mathbf{w}_n$, set it equal to zero and solve for $\mathbf{w}_n$. We switch to matrix-vector notation, and drop the n index to reduce clutter.

$$\begin{aligned}\mathcal{L} &\propto (\mathbf{B}\mathbf{w} - \mathbf{x})^\top (\mathbf{B}\mathbf{w} - \mathbf{x}) \\ &= -\frac{1}{2\sigma^2} \left(\mathbf{w}^\top \mathbf{B}^\top \mathbf{B} \mathbf{w} - 2\mathbf{w}^\top \mathbf{B}^\top \mathbf{x} + \mathbf{x}^\top \mathbf{x} \right) - \frac{1}{2\sigma_0^2} \left(\mathbf{w}^\top \mathbf{w} \right)\end{aligned}$$

- And the gradient,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = -\frac{1}{\sigma^2} \left(\mathbf{B}^\top \mathbf{B} \mathbf{w} - \mathbf{B}^\top \mathbf{x} \right) - \frac{1}{\sigma_0^2} \mathbf{w} \quad (1)$$

- Solve for $\mathbf{w}$,

$$\begin{aligned}\frac{1}{\sigma^2} \left(\mathbf{B}^\top \mathbf{B} \mathbf{w} - \mathbf{B}^\top \mathbf{x} \right) - \frac{1}{\sigma_0^2} \mathbf{w} &= 0 \\ \left(\mathbf{B}^\top \mathbf{B} + \frac{\sigma^2}{\sigma_0^2} \mathbf{I} \right) \mathbf{w} &= \mathbf{B}^\top \mathbf{x} \\ \rightarrow \hat{\mathbf{w}} &= \left(\mathbf{B}^\top \mathbf{B} + \frac{\sigma^2}{\sigma_0^2} \mathbf{I} \right)^{-1} \mathbf{B}^\top \mathbf{x}\end{aligned}$$

The MAP solution for $\mathbf{w}_n$

- Note this is the MAP solution for $\mathbf{w}_n$ for $n \in \{1, \dots, N\}$ that we saw before:
 - ▶ Notez que c'est la solution MAP pour $\mathbf{w}_n$, $n \in \{1, \dots, N\}$:

$$\hat{\mathbf{w}}_n = \left(B^\top B + \frac{\sigma^2}{\sigma_0^2} I \right)^{-1} B^\top \mathbf{x}$$

The MAP solution for $\mathbf{w}_n$

- Note this is the MAP solution for $\mathbf{w}_n$ for $n \in \{1, \dots, N\}$ that we saw before:

- ▶ Notez que c'est la solution MAP pour $\mathbf{w}_n$, $n \in \{1, \dots, N\}$:

$$\hat{\mathbf{w}}_n = \left(B^\top B + \frac{\sigma^2}{\sigma_0^2} I \right)^{-1} B^\top \mathbf{x}$$

- You can also easily show that the Maximum-Likelihood solution is very similar (Uniform Prior):

- ▶ On peut aussi très facilement montrer que la solution Maximum-Likelihood est très similaire (si on utilise un prior uniforme):

$$\hat{\mathbf{w}}_n = \left(B^\top B \right)^{-1} B^\top$$

The MAP solution for $\mathbf{w}_n$

- Note this is the MAP solution for $\mathbf{w}_n$ for $n \in \{1, \dots, N\}$ that we saw before:

- ▶ Notez que c'est la solution MAP pour $\mathbf{w}_n$, $n \in \{1, \dots, N\}$:

$$\hat{\mathbf{w}}_n = \left(B^\top B + \frac{\sigma^2}{\sigma_0^2} I \right)^{-1} B^\top \mathbf{x}$$

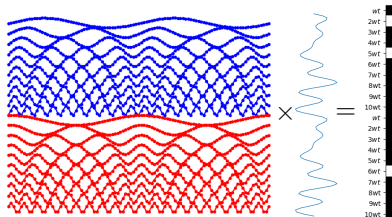
- You can also easily show that the Maximum-Likelihood solution is very similar (Uniform Prior):

- ▶ On peut aussi très facilement montrer que la solution Maximum-Likelihood est très similaire (si on utilise un prior uniforme):

$$\hat{\mathbf{w}}_n = \underbrace{\left(B^\top B \right)^{-1}}_{B^\dagger} B^\top \mathbf{x}$$

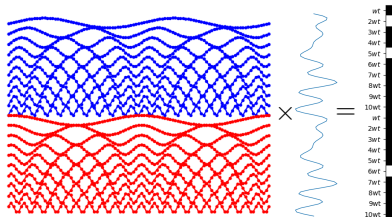
Ok, but how about the DFT stuff we talked about last week?

- Ok, but were we doing this last week? Is this optimal?
- D'accord, on faisait ça la semaine dernière? Est-ce optimale?



Ok, but how about the DFT stuff we talked about last week?

- Ok, but were we doing this last week? Is this optimal?
- D'accord, on faisait ça la semaine dernière? Est-ce optimale?



- Yes!/Oui!

Fourier Transform Maximizes Gaussian Likelihood

■ Let's see / Voyons:

$$\hat{\mathbf{w}}_n = (B^\top B)^{-1} B^\top$$

Fourier Transform Maximizes Gaussian Likelihood

- Let's see / Voyons:

$$\hat{\mathbf{w}}_n = (B^\top B)^{-1} B^\top$$

Fourier Transform Maximizes Gaussian Likelihood

- Let's see / Voyons:

$$\hat{\mathbf{w}}_n = \underbrace{(B^\top B)^{-1} B^\top}_{B^\dagger} \mathbf{x}$$

- Note that in this case $B = F$, and $F^*F = FF^* = I$. Let's substitute.
 - ▶ F^* , is the Hermitian adjoint (transpose and flip the imaginary part)
- Notez que dans ce cas-ci $B = F$, et $F^*F = FF^* = I$. Substitutions:.
 - ▶ F^* , est le Hermitian adjoint (on transpose et on flippe la partie imaginaire)
- DFT Matrix is orthogonal so,

$$\hat{\mathbf{w}}_n = F\mathbf{x}$$

Fourier Transform Maximizes Gaussian Likelihood

- Let's see / Voyons:

$$\hat{\mathbf{w}}_n = \underbrace{(B^\top B)^{-1} B^\top}_{B^\dagger} \mathbf{x}$$

- Note that in this case $B = F$, and $F^*F = FF^* = I$. Let's substitute.

- ▶ F^* , is the Hermitian adjoint (transpose and flip the imaginary part)

- Notez que dans ce cas-ci $B = F$, et $F^*F = FF^* = I$. Substitutions:

- ▶ F^* , est le Hermitian adjoint (on transpose et on flippe la partie imaginaire)

- DFT Matrix is orthogonal so,

$$\hat{\mathbf{w}}_n = F\mathbf{x}$$

- We can then deduce that DFT maximizes the Gaussian Likelihood under this linear regression / decomposition model! (or minimizes l_2 error)

- ▶ On peut alors déduire que DFT maximise le likelihood Gaussian sous ce modèle. (ou il minimise l'erreur l_2 .)

Table of Contents

The Decomposition Framework

Fixed Basis Decompositions (Linear Regression)

Learnable Basis Decompositions

Principal Component Analysis

Independent Component Analysis

Non-Negative Matrix Factorization (NMF)

Let's learn B too!

- Note that earlier we were only learning the activations W for a fixed basis matrix B :

- ▶ On apprenait juste les activations W pour des bases fixes B :

$$\min_W \|X - BW\|$$

- ▶ But, we can learn B too!
- ▶ Mais, on peut apprendre B aussi!

$$\min_{B,W} \|X - BW\|$$

But how?

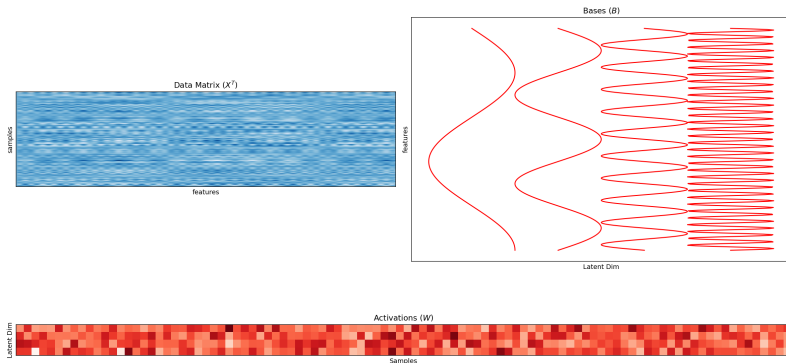
- We can alternate the least squares solution such that,
 - ▶ On peut juste alterner entres les solutions least-squares,

Algorithm 1 Alternating Least Squares

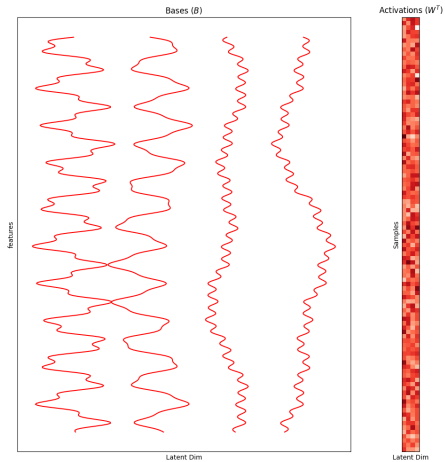
```
1: procedure ALTERNATING LEAST SQUARES
   Input: Input Data Matrix  $X$ . Threshold value  $\epsilon$ .
   Output: Estimated Basis and Activation Matrices  $\hat{B}$ ,  $\hat{W}$ .
2:   Initialize  $\hat{B}$ ,  $\hat{W}$ .
3:   while  $\|X - \hat{B}\hat{W}\| \geq \epsilon$  do
4:      $\hat{W} = \hat{B}^\dagger X$ 
5:      $\hat{B} = X\hat{W}^\dagger$ 
6:   end while
7: end procedure
```

Alternating Least Squares Dataset

- Let's try alternating least squares on this dataset
 - ▶ Essayong cet methode sur ce dataset



The result



- Kinda good, but we can do better.
 - Ça fait quelque chose, mais pas idéale.

Table of Contents

The Decomposition Framework

Fixed Basis Decompositions (Linear Regression)

Learnable Basis Decompositions

- Principal Component Analysis

- Independent Component Analysis

- Non-Negative Matrix Factorization (NMF)

Principal Component Analysis

- The goal is to find uncorrelated latent components W .
 - ▶ Le but est de trouver des components W qui ne sont pas co-reliés.

Principal Component Analysis

- The goal is to find uncorrelated latent components W .
 - ▶ Le but est de trouver des components W qui ne sont pas co-reliés.
- We want to find an orthogonal transformation $B^\top B = I$, so that $W = B^\top X$.
 - ▶ On veut trouver une transformation orthogonale.
- Let's calculate:

$$\begin{aligned}\text{covar}(w) &= \text{covar}(B^\top x) \\ &= B^\top \underbrace{\mathbb{E}[(x - \mathbb{E}[x])(x - \mathbb{E}[x])^\top]}_{:=C} B\end{aligned}$$

Principal Component Analysis

- The goal is to find uncorrelated latent components W .
 - ▶ Le but est de trouver des components W qui ne sont pas co-reliés.
- We want to find an orthogonal transformation $B^T B = I$, so that $W = B^T X$.
 - ▶ On veut trouver une transformation orthogonale.
- Let's calculate:

$$\begin{aligned}\text{covar}(w) &= \text{covar}(B^T x) \\ &= B^T \underbrace{\mathbb{E}[(x - \mathbb{E}[x])(x - \mathbb{E}[x])^T]}_{:=C} B\end{aligned}$$

- So basically want to solve for B such that $B^T C B = I$, that is C is whitened.
 - ▶ Dans le fond on veut blanchir C .

Principal Component Analysis

- The goal is to find uncorrelated latent components W .
 - ▶ Le but est de trouver des components W qui ne sont pas co-reliés.
- We want to find an orthogonal transformation $B^T B = I$, so that $W = B^T X$.
 - ▶ On veut trouver une transformation orthogonale.
- Let's calculate:

$$\begin{aligned}\text{covar}(w) &= \text{covar}(B^T x) \\ &= B^T \underbrace{\mathbb{E}[(x - \mathbb{E}[x])(x - \mathbb{E}[x])^T]}_{:=C} B\end{aligned}$$

- So basically want to solve for B such that $B^T C B = I$, that is C is whitened.
 - ▶ Dans le fond on veut blanchir C .
- Any ideas? (Ei.. SV.. ?)

Eigenvectors to the rescue

- Consider the SVD of C , s.t. $C = U\Sigma U^\top$. **(Same as eigenvalue decomp. why?)**
 - ▶ Considérons le SVD de C .

Eigenvectors to the rescue

- Consider the SVD of C , s.t. $C = U\Sigma U^\top$. **(Same as eigenvalue decomp. why?)**

- ▶ Considérons le SVD de C .

- Let's substitute. Can you see the solution now? / Voyez-vous la solution?

$$B^\top U \Sigma U^\top B = I$$

- We can easily see that $B = U\Sigma^{-1/2}$ does the job!

- ▶ On a trouvé la solution $B = U\Sigma^{-1/2}$!

Eigenvectors to the rescue

- Consider the SVD of C , s.t. $C = U\Sigma U^\top$. **(Same as eigenvalue decomp. why?)**

- ▶ Considérons le SVD de C .

- Let's substitute. Can you see the solution now? / Voyez-vous la solution?

$$B^\top U \Sigma U^\top B = I$$

- We can easily see that $B = U\Sigma^{-1/2}$ does the job!

- ▶ On a trouvé la solution $B = U\Sigma^{-1/2}$!

- The columns of U are the eigenvectors of C !

- ▶ Les colonnes de U sont les vecteurs propres de C .

A note on variance

- Note that this way we maximize the variance along the direction of b_1 . / Cette solution maximise la variance sur la direction de b_1 .

$$\mathcal{V} := \text{var}(b_1^\top x) = b_1^\top \mathbb{E}[(x - \mathbb{E}[x])(x - \mathbb{E}[x])^\top] b_1$$

- Let's maximize this variance such that $b_1^\top b_1 = 1$. / Maximisons la variance telle que b_1 est de norme unitaire.

$$\mathcal{V} = b_1^\top C b_1 - \lambda b_1^\top b_1$$

$$\frac{\partial \mathcal{V}}{\partial b_1} = 2C b_1 - \lambda b_1$$

$$\rightarrow C b_1 = \lambda b_1$$

- So, we have the definition of an eigenvector... / Donc c'est la définition du vecteur propre de C .

A note on variance

- Note that this way we maximize the variance along the direction of b_1 . / Cette solution maximise la variance sur la direction de b_1 .

$$\mathcal{V} := \text{var}(b_1^\top x) = b_1^\top \mathbb{E}[(x - \mathbb{E}[x])(x - \mathbb{E}[x])^\top] b_1$$

- Let's maximize this variance such that $b_1^\top b_1 = 1$. / Maximisons la variance telle que b_1 est de norme unitaire.

$$\mathcal{V} = b_1^\top C b_1 - \lambda b_1^\top b_1$$

$$\frac{\partial \mathcal{V}}{\partial b_1} = 2C b_1 - \lambda b_1$$

$$\rightarrow C b_1 = \lambda b_1$$

- So, we have the definition of an eigenvector... / Donc c'est la définition du vecteur propre de C .
- Similarly the other principal components are found..
 - ▶ Similairement, les autres composants principaux sont trouvés..

The recipe for PCA

Algorithm 2 Principal Component Analysis

1: **procedure** PCA

Input: Data matrix $X \in \mathbb{R}^{M \times N}$, target dimension K .

Output: Basis $\hat{B}$ (principal components), embeddings $\hat{W}$.

2: Center the data: $\bar{X} = X - \mathbb{E}[x]$.

3: Form the covariance: $C = \bar{X}\bar{X}^\top = U\Sigma^2U^\top$.

4: Take the top- K eigenvectors of C (columns of U): $\hat{B} = U_{:, 1:K}$.

5: Project onto the basis: $\hat{W} = \hat{B}^\top \bar{X}$.

6: **end procedure**

The recipe for PCA

Algorithm 3 Principal Component Analysis

1: **procedure** PCA

Input: Data matrix $X \in \mathbb{R}^{M \times N}$, target dimension K .

Output: Basis $\hat{B}$ (principal components), embeddings $\hat{W}$.

2: Center the data: $\bar{X} = X - \mathbb{E}[x]$.

3: Form the covariance: $C = \bar{X}\bar{X}^\top = U\Sigma^2U^\top$.

4: Take the top- K eigenvectors of C (columns of U): $\hat{B} = U_{:, 1:K}$.

5: Project onto the basis: $\hat{W} = \hat{B}^\top \bar{X}$.

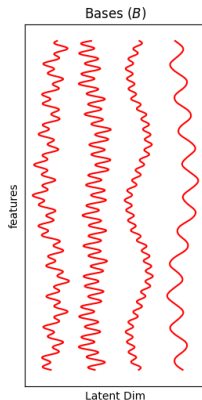
6: **end procedure**

■ Equivalently, use the left singular vectors of $\bar{X} = U\Sigma V^\top$ (SVD) — same U .

► De façon équivalente, on utilise les vecteurs singuliers gauche de $\bar{X}$ (SVD).

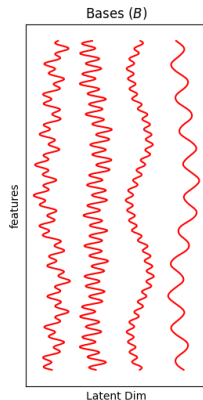
PCA on our sinusoid basis problem

- A bit better!



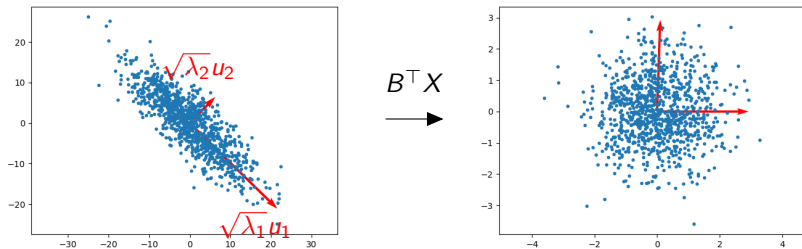
PCA on our sinusoid basis problem

- A bit better!



- We can improve this! (more on this later ICA)
 - ▶ On peut améliorer ça. (On verra)

Interpretation of PCA



Note that $B^T = \text{diag}([\sqrt{\lambda_1}, \sqrt{\lambda_2}])^{-1} U^T$.

Dimensionality Reduction with PCA

- Note that PCA makes the following decomposition / On fait la décomposition suivante:

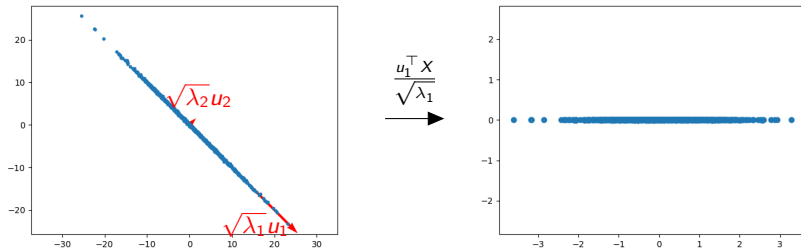
$$\text{var}(X) = \sum_{k=1}^K \lambda_k u_k u_k^\top$$

Dimensionality Reduction with PCA

- Note that PCA makes the following decomposition / On fait la décomposition suivante:

$$\text{var}(X) = \sum_{k=1}^K \lambda_k u_k u_k^\top$$

- Let's consider the following case / Considérons le cas suivant:

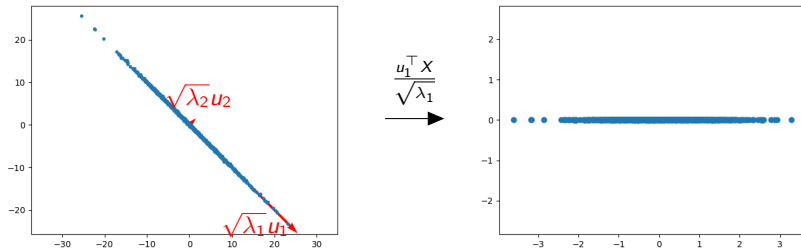


Dimensionality Reduction with PCA

- Note that PCA makes the following decomposition / On fait la décomposition suivante:

$$\text{var}(X) = \sum_{k=1}^K \lambda_k u_k u_k^\top$$

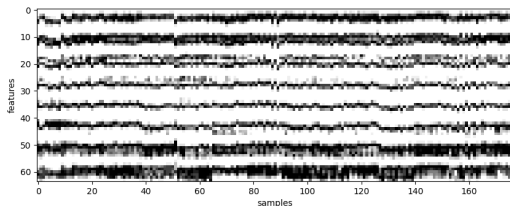
- Let's consider the following case / Considérons le cas suivant:



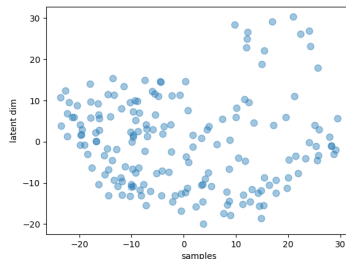
- $\lambda_1 = 10$, $\lambda_2 = 0.1$. Most of the variance is along one direction. We can only use one dim. / La variance est sur une direction. On peut s'en débarrasser d'une direction.

Embedding digits in 2 dimensions

- We only keep two dimensions / On garde juste 2 dimensions

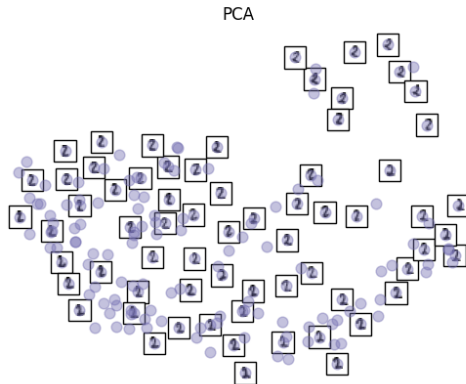


$$B^T(X - \mathbb{E}[x])$$



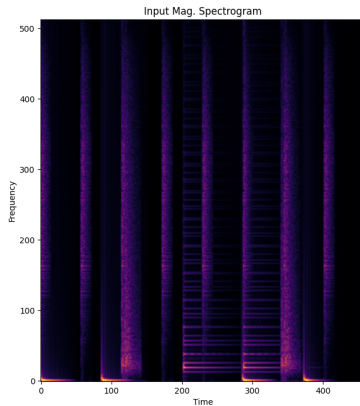
Embedding digits in 2 dimensions

- We only keep two dimensions / On garde juste 2 dimensions

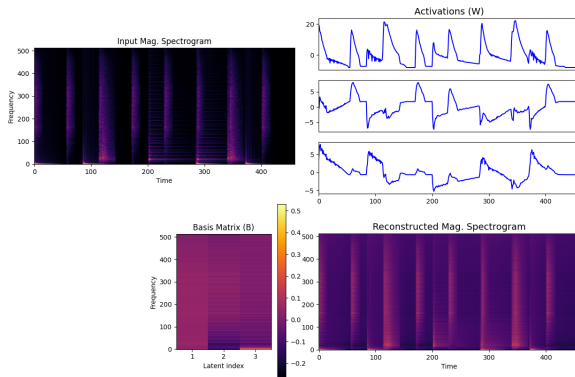


Embedding spectra

- Let's embed this spectrogram into a 3 dim. space / On va embedder ce spectrogram dans un 3 dim. espace. Listen



Embedding spectra with PCA



PCA on time-series

- Let's apply PCA on local windows of a time series / Appliquons PCA on des fenêtrés d'un time series

$$x_1, x_2, \dots, x_T$$

- Pack in a data matrix as follows

$$X = \begin{bmatrix} x_1 & x_{1+s} & x_{1+2s} & \dots \\ x_2 & x_{2+s} & x_{2+2s} & \dots \\ \vdots & \vdots & \vdots & \ddots \\ x_N & x_{N+s} & x_{N+2s} & \dots \end{bmatrix}$$

- Note that if we do $W = FX$, this is equal to Short-Time-Fourier-Transform that we saw last week.
 - ▶ Notez que si on utilise les bases de Fourier ça donne STFT.
 - ▶ s is the hopsize we saw for STFT. / s est le hopsize, même que STFT.

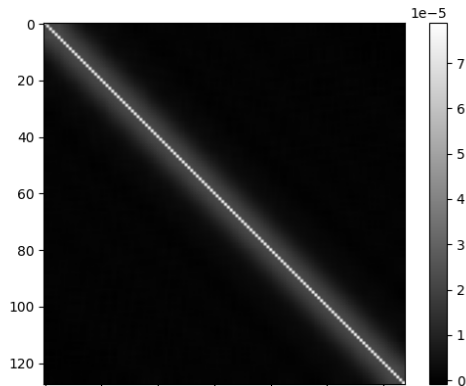
But let's do PCA instead

- Let's consider this process / Considerons cette processus

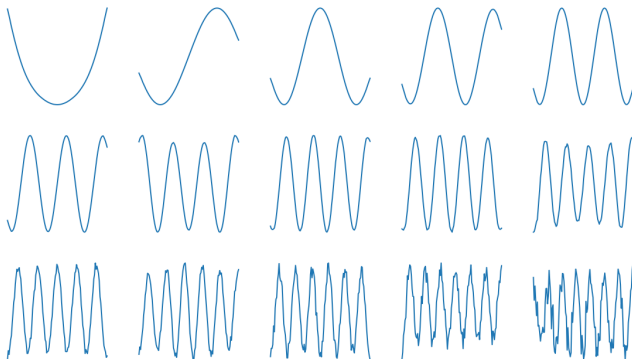
$$x_t = x_{t-1} + 0.825x_{t-2} + 0.65x_{t-3} + 0.475x_{t-4} + 0.3x_{t-5} + n$$

► $n \sim \mathcal{N}(0, 0.008^2)$

- The covariance matrix



Sinusoids!



Sinusoids (DCT bases are eigenvectors of circulant matrices) .. / Les bases sinusoids sont les vecteurs propres des matrices circulants.

More generally: Convolution as a Matrix Multiplication

- Convolution can be expressed as a matrix multiplication.
 - Convolution peut-etre exprimée comme une multiplication de matrices.

$$w * x = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \underbrace{\begin{bmatrix} w_1 & 0 & 0 & 0 \\ w_2 & w_1 & 0 & 0 \\ 0 & w_2 & w_1 & 0 \\ 0 & 0 & w_2 & w_1 \\ 0 & 0 & 0 & w_2 \end{bmatrix}}_{:=C} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

- C is a Circulant / Toeplitz matrix.

More generally: Convolution as a Matrix Multiplication

- Convolution can be expressed as a matrix multiplication.

► Convolution peut-etre exprimée comme une multiplication de matrices.

$$w * x = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \underbrace{\begin{bmatrix} w_1 & 0 & 0 & 0 \\ w_2 & w_1 & 0 & 0 \\ 0 & w_2 & w_1 & 0 \\ 0 & 0 & w_2 & w_1 \\ 0 & 0 & 0 & w_2 \end{bmatrix}}_{:=C} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

- C is a Circulant / Toeplitz matrix.
- And the eigenvectors are sinusoids.

More generally: Convolution as a Matrix Multiplication

- Convolution can be expressed as a matrix multiplication.

► Convolution peut-etre exprimée comme une multiplication de matrices.

$$w * x = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \underbrace{\begin{bmatrix} w_1 & 0 & 0 & 0 \\ w_2 & w_1 & 0 & 0 \\ 0 & w_2 & w_1 & 0 \\ 0 & 0 & w_2 & w_1 \\ 0 & 0 & 0 & w_2 \end{bmatrix}}_{:=C} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

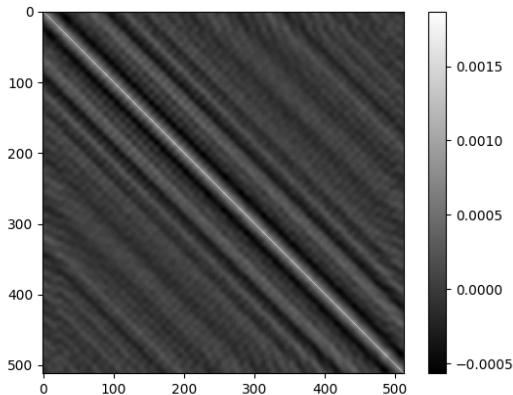
- C is a Circulant / Toeplitz matrix.
- And the eigenvectors are sinusoids.
- Not only that, the eigenvectors form the DFT matrix!

<https://web.mit.edu/18.06/www/Spring17/Circulant-Matrices.pdf>

Same thing on speech

- And here's the covariance matrix for a 14sec long speech signal.

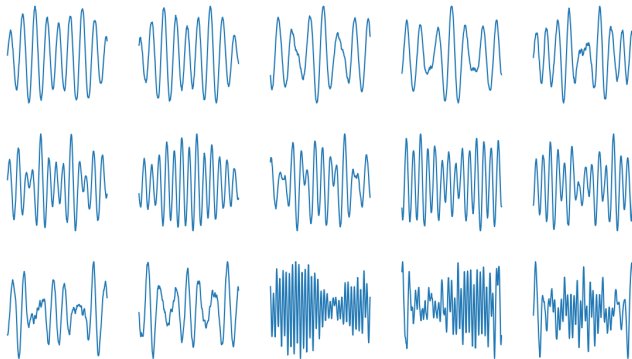
- ▶ Matrice de covariance pour un parole de 14secondes.



- Seems like we have high covariance in the neighborhood, then some periodicity.

- ▶ Haut covariance locale, et un peu de périodicité.

Sinusoids!



Listen

So, it seems sinusoidal bases are kinda statistically optimal for local covariance as well.. / Les bases sinusoids sont optimale si on a une covariance locale!.

Table of Contents

The Decomposition Framework

Fixed Basis Decompositions (Linear Regression)

Learnable Basis Decompositions

Principal Component Analysis

Independent Component Analysis

Non-Negative Matrix Factorization (NMF)

Independent Component Analysis (ICA)

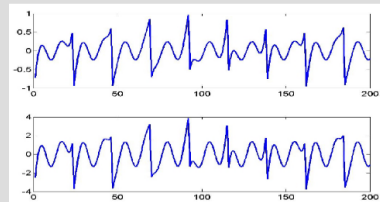
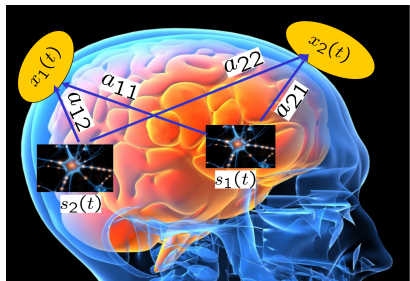
- ICA estimates a square mixing matrix $B \in \mathbb{R}^{K \times K}$, such that,
 - ▶ ICA estime un matrice carré B , telle que,

$$x = Bw + n$$

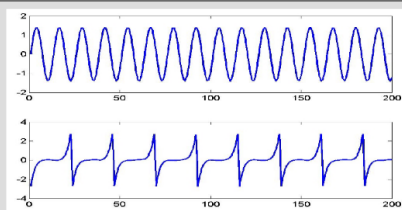
the elements of $w \in \mathbb{R}^K$ are statistically independent. / les éléments du vecteur w sont statistiquement indépendant.

- We want to achieve $p(w) = p(w_1)p(w_2) \dots p(w_K)$.
 - ▶ On veut que le probabilité joint $p(s)$ se factorise.

ICA Application



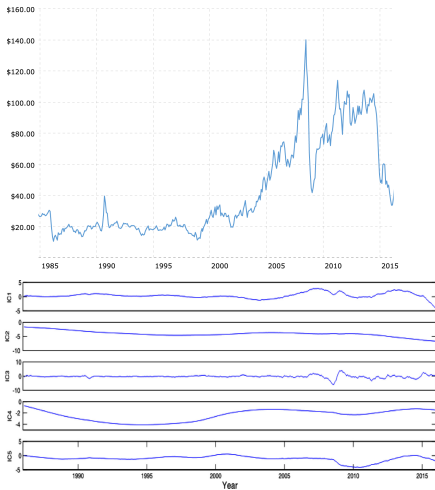
Observations (Mixtures)



ICA estimated signals

[images taken from https://www.cs.cmu.edu/~bapoczso/other_presentations/ICA_26_10_2009.pdf]

Source Separation for Financial Data



- In the paper **Factor analysis of financial time series using EEMD-ICA based approach** the authors decompose oil prices using an ICA variant.
- They claim:
 - ▶ IC1 is correlated to USD.
 - ▶ IC2 is correlated to oil supply and demand.
 - ▶ IC3 is correlated to political and extreme events.
 - ▶ IC4 reflects cyclical nature of oil prices.
 - ▶ IC5 is correlated with stock, gold markets.

Methods to solve ICA (high-level)

- Non linear decorrelation $\mathbb{E}[f(w_i)g(w_j)]$, for fixed f, g .
 - ▶ Decorrelation non-linéaire pour $\mathbb{E}[f(w_i)g(w_j)]$, f, g sont fixes..
 - ▶ Cichocki-Unbehauen algorithm

Methods to solve ICA (high-level)

- Non linear decorrelation $\mathbb{E}[f(w_i)g(w_j)]$, for fixed f, g .
 - ▶ Decorrelation non-linéaire pour $\mathbb{E}[f(w_i)g(w_j)]$, f, g sont fixes..
 - ▶ Cichocki-Unbehauen algorithm

- Higher order diagonalization.

- ▶ Diagonalize

$$Q(s) := \mathbb{E}[w_i w_j w_k w_l] - \mathbb{E}[w_i w_j] \mathbb{E}[w_k w_l] - \mathbb{E}[w_i w_k] \mathbb{E}[w_j w_l] - \mathbb{E}[w_i w_l] \mathbb{E}[w_j w_k]$$

- ▶ Remember PCA diagonalizes $\mathbb{E}[ww^\top]$.

Methods to solve ICA (high-level)

- Non linear decorrelation $\mathbb{E}[f(w_i)g(w_j)]$, for fixed f, g .
 - ▶ Decorrelation non-linéaire pour $\mathbb{E}[f(w_i)g(w_j)]$, f, g sont fixes..
 - ▶ Cichocki-Unbehauen algorithm
- Higher order diagonalization.
 - ▶ Diagonalize

$$Q(s) := \mathbb{E}[w_i w_j w_k w_l] - \mathbb{E}[w_i w_j] \mathbb{E}[w_k w_l] - \mathbb{E}[w_i w_k] \mathbb{E}[w_j w_l] - \mathbb{E}[w_i w_l] \mathbb{E}[w_j w_k]$$

- ▶ Remember PCA diagonalizes $\mathbb{E}[ww^\top]$.
- Info-theoretic approach

$$\min \text{KL}(p(w) \| p(w_1)p(w_2) \dots p(w_K)) = \min \int p(w) \log \frac{p(w)}{\prod_k p(w_k)}$$

- ▶ We try to make the product of marginals become the joint / on essaie de faire la produit de marginales égale à joint.

Methods to solve ICA (high-level)

- Non linear decorrelation $\mathbb{E}[f(w_i)g(w_j)]$, for fixed f, g .
 - ▶ Decorrelation non-linéaire pour $\mathbb{E}[f(w_i)g(w_j)]$, f, g sont fixes..
 - ▶ Cichocki-Unbehauen algorithm

- Higher order diagonalization.

- ▶ Diagonalize

$$Q(s) := \mathbb{E}[w_i w_j w_k w_l] - \mathbb{E}[w_i w_j] \mathbb{E}[w_k w_l] - \mathbb{E}[w_i w_k] \mathbb{E}[w_j w_l] - \mathbb{E}[w_i w_l] \mathbb{E}[w_j w_k]$$

- ▶ Remember PCA diagonalizes $\mathbb{E}[ww^\top]$.

- Info-theoretic approach

$$\min \text{KL}(p(w) \| p(w_1)p(w_2) \dots p(w_K)) = \min \int p(w) \log \frac{p(w)}{\prod_k p(w_k)}$$

- ▶ We try to make the product of marginals become the joint / on essaie de faire la produit de marginales égale à joint.

- More: FastICA, Neural Nets, Negentropy (Measure of non-gaussianity), More...

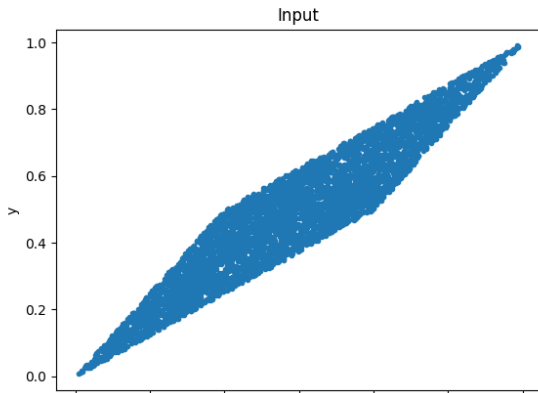
PCA vs ICA

- Let's consider this toy example

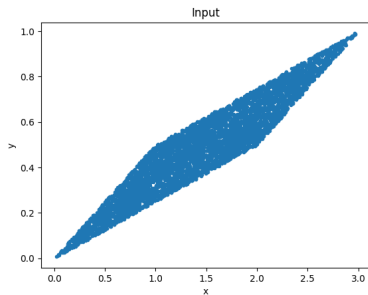
$$r_1, r_2 \sim \mathcal{U}(0, 1)$$

$$x = r_1 + r_2$$

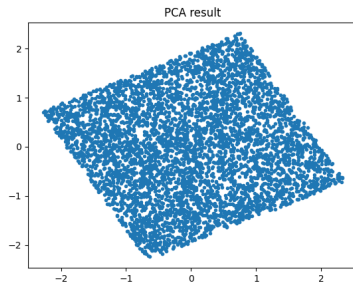
$$y = 2r_1 + r_2$$



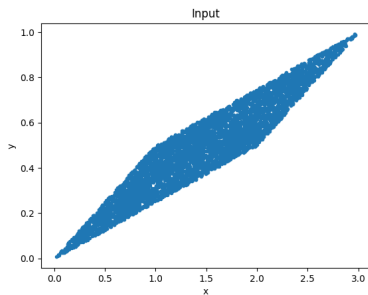
PCA vs ICA



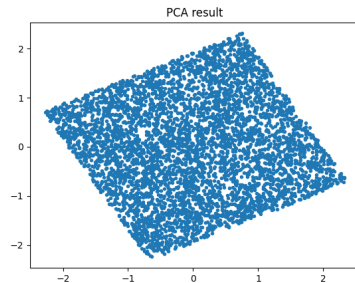
PCA
→



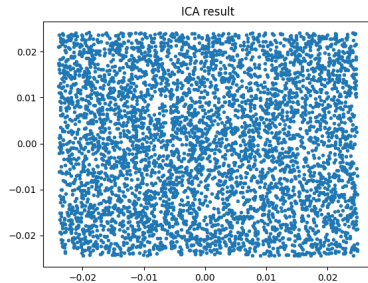
PCA vs ICA



PCA



ICA



PCA on steroids

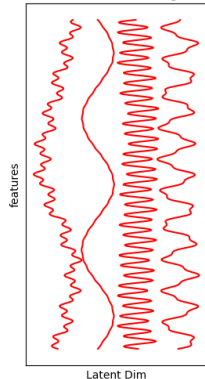
- We were doing the decomposition / On faisait la décomposition,

$$X = BW$$

- We can apply ICA to obtain / On peut appliquer ICA pour obtenir

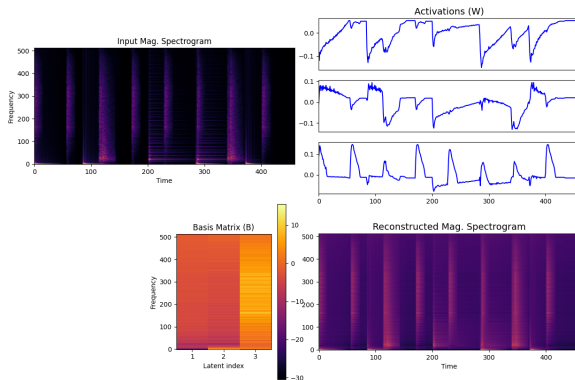
$$X = BB_I W_I = \tilde{B} W_I$$

Bases (B) after ICA mixing matrix



- Closer to sinusoids!

Embedding spectra with ICA



Bit better but we can do better..

- PCA assumes that everything is Gaussian. (For Gaussian data it does return independent dimensions)
 - ▶ PCA suppose que le monde est Gaussienne.
- iCA does not assume Gaussian, and try to achieve independence.
 - ▶ ICA essaie d'obtenir l'indépendance.
- Most ICA estimators are approximate
 - ▶ La majorité des estimateurs ICA sont approximatives.
- We don't have an important ordering of components, so no dim. reduction
 - ▶ On n'a pas un ordering des composant, donc on ne peut faire une reduction de dimensions.
 - ▶ We can however combine it PCA to improve it. / On peut le combiner avec PCA pour l'améliorer.

Table of Contents

The Decomposition Framework

Fixed Basis Decompositions (Linear Regression)

Learnable Basis Decompositions

Principal Component Analysis

Independent Component Analysis

Non-Negative Matrix Factorization (NMF)

Non-Negative Matrix Factorization

- We want to again optimize for B, W , but for $W \geq 0, W \geq 0$. i

$$\min_{B, W} \|X - BW\|$$
$$s.t. B \geq 0, W \geq 0.$$

- First proposed in 1999 Nature paper. / Proposé dans un papier Nature en 1999.
- Works pretty well on data non-négative. Fonctionne magiquement bien sur le data non-negative.
- We often work with non-negative data. (counts, pixels, energy...) / On travaille souvent avec du data non-négative.

Non-Negative Matrix Factorization

- We want to again optimize for B, W , but for $W \geq 0, W \geq 0$. i

$$\begin{aligned} \min_{B, W} & \|X - BW\| \\ \text{s.t.} & B \geq 0, W \geq 0. \end{aligned}$$

- First proposed in 1999 Nature paper. / Proposé dans un papier Nature en 1999.
- Works pretty well on data non-négative. Fonctionne magiquement bien sur le data non-negative.
- We often work with non-negative data. (counts, pixels, energy...) / On travaille souvent avec du data non-négative.
- If we have negative values in our estimates, they cancel out, harm interpretability. / Si on a des valeurs négatives dans les parameters, ça nuit l'interprétabilité.

But how?

- We can alternate the least squares solutions and also project such,
 - ▶ On peut juste alterner entre les solutions least-squares avec une addition des projections,

Algorithm 4 Alternating Least Squares for NMF

1: **procedure** ALTERNATING LEAST SQUARES FOR NMF

Input: Input Data Matrix X . Threshold value ϵ .

Output: Estimated Basis and Activation Matrices $\widehat{B}$, $\widehat{W}$.

2: Initialize $\widehat{B} \geq 0$, $\widehat{W} \geq 0$.

3: **while** $\|X - \widehat{B}\widehat{W}\| \geq \epsilon$ **do**

4: $\widehat{W} = \widehat{B}^\dagger X$; $\widehat{W} = \max(0, \widehat{W})$

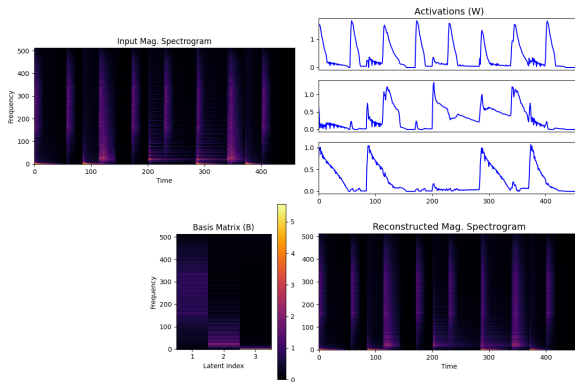
5: $\widehat{B} = X\widehat{W}^\dagger$; $\widehat{B} = \max(0, \widehat{B})$

6: **end while**

7: **end procedure**

- There are also other algos. (e.g. Multiplicative Updates, probabilistic versions..)
 - ▶ Y a des autres algos. aussi.

NMF to rescue



PCA, NMF or ICA?

- It depends. / Ça depends.
- PCA is great for dim. reduction / PCA est très utile pour réduire la dimensionnalité.
- ICA gives more sparse/independent embeddings / ICA donne des embeddings plus parsimonieux.
- NMF gives interpretable results, but only for non-negative / NMF donne des résultats interprétables, mais juste pour des données non-négatives.

- We have introduced a framework that handles fixed basis regression, and learnable-basis regression.
 - ▶ On a introduit un framework qui peut gérer la regression avec des bases fixés, et regression avec des bases apprises.
- We have talked about very important latent variable methods such as PCA, ICA, NMF.
 - ▶ On a parlé des méthodes importants de variables latents comme PCA, ICA et NMF.

Suggested Reading

- Chapters 4, 12, Bishop
- The NMF Nature paper <https://www.nature.com/articles/44565/>
- Eigenfaces <https://en.wikipedia.org/wiki/Eigenface>
- Circulant matrices
<https://web.mit.edu/18.06/www/Spring17/Circulant-Matrices.pdf>

- What if we want to learn non-linear embeddings? Manifold methods!
 - ▶ Qu'est-ce qu'on fait si on veut apprendre des embeddings non-linéaires? Méthodes de manifolds!