

# Machine Learning (ML) meets Signal Processing (SP)

Laurent Oudre and Nicolas Vayatis

école —————  
normale —————  
supérieure —————  
paris-saclay —————

université  
PARIS-SACLAY

Luminy, January 2021

# Related research at Centre Borelli



<http://www.centreborelli.fr>

## Our ML/SP group at Centre Borelli

*Seniors:* Argyris Kalogeratos, Mathilde Mougeot, Laurent Oudre, NV

*Juniors:* Ioannis Bargiotas, Harry Sevi, Brian Tervil, Charles Truong

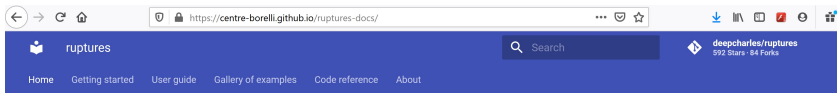
*PhD students:* Mounir Atiq, Nicolas Brunot, Batiste Le Bars, Sylvain Combettes, Alejandro de la Concha Duarte, Amir Dib, Mathilde Fekom, Marie Garin, Pierre Humbert, Sylvain Jung, Dimitri Keriven-Serpollet, Myrto Limnios, Alice Nicolaï, Antoine de Mathelin, Antoine Mazarguil, Anthea Merida, Ludovic Minvielle, Guillaume Richard, Theo Saillant, Anne Zhao

*Interns:* Theo Gnassounou, Frédéric Zheng

\*Industrial partners: Autorité des Marchés Financiers, Banque de France, CEA, CNES, EDF, Engie, Michelin, Renault, SNCF, Sigfox, Tarkett, Thalès

\*Main hospitals: HIA Percy, HIA Bégin, APHP Fernand Widal-Lariboisière, APHP Necker

# Python package: ruptures



## Home

Welcome to ruptures

## Welcome to ruptures

Maintained? **yes** build **passing** python 3.6 | 3.7 | 3.8 | 3.9 pypi package **1.1.2** docs **passing**  
license **BSD License** downloads **988k** code style **black** launch **binder**

ruptures is designed to perform offline change point algorithms within the Python language. Also in this library, new methods are presented.

### How to cite

If you use ruptures in a scientific publication, we would appreciate citations to the following paper:

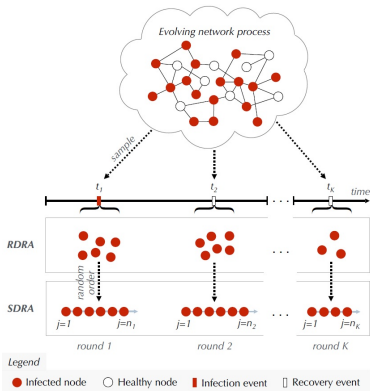
- Truong, C., Oudre, L., & Vayatis, N. (2020). Selective review of offline change point detection methods. *Signal Processing*, 167. [abstract] [doi] [pdf]

### Contact

Concerning this package, its use and bugs, use the [issue page](#) of the ruptures repository.

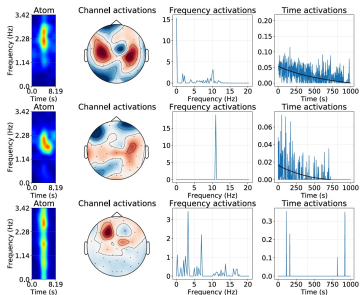
# PhD of Mathilde Fekom: Epidemics

- Motivation/Setup: Combining concepts from operations research for sequential decision process and epidemic control
- Central research question: efficient control/propagation of the diffusion process under partial information with limited resources



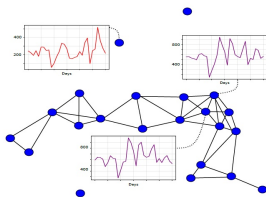
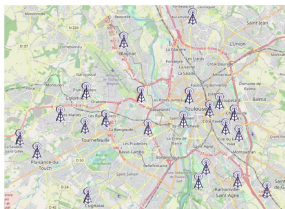
# PhD of Pierre Humbert: Depth of Anesthesia

- Motivation/Setup: monitor the vigilance states of a patient during general anesthesia through physiological signals
- Central research question: interpretation of frequency patterns of EEG signals and their correlation to other vital signals (ECG, blood pressure...)



# PhD of Batiste Le Bars: Telecommunication networks

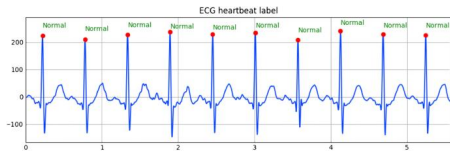
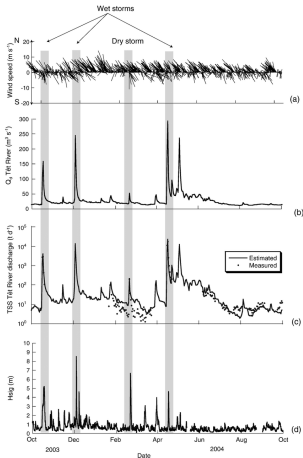
- Motivation/Setup: Monitoring telecommunication infrastructure based on low-frequency signals emitted by a fleet of moving sensors
- Central research question: anomaly detection and changepoint detection based on graph signal data



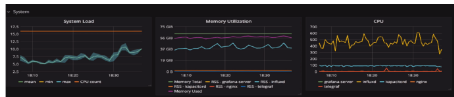
# Ubiquity of signals



# Time-dependent data are everywhere



Meteorology,  
Finance,  
Healthcare,  
Monitoring,  
Epidemiology,  
Sensor networks...

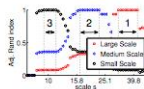
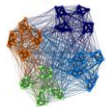
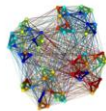
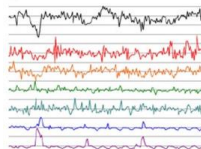


# Univariate vs. multivariate

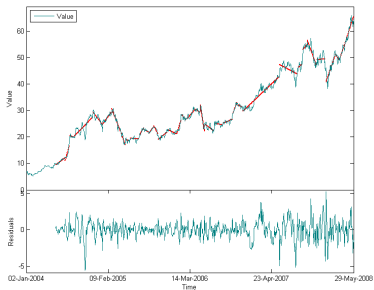


2D/3D trajectories,  
Multivariate time series,  
Multimodal data from  
sensor networks,  
Graph signals

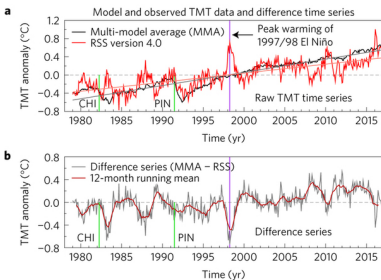
## Sensor Data



# Puzzling questions about signals, e.g. what is a trend?

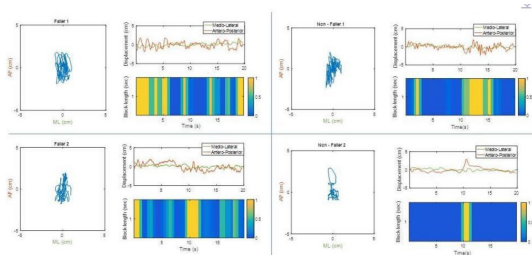
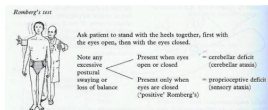


Sell or buy



*The "hiatus"*

# Recent projects: (1) Posture assessment with digital Romberg test



“Unquiet blocks” are more frequent in fallers than in non-fallers.

\*Colour-bars show probability of containing evidences of instability

From [Bargiotas et al. On the importance of local dynamics in statokinesigram: A multivariate approach for postural control evaluation in elderly (PlosOne, 2018)]

## Recent projects: (2) Gait assessment during locomotion

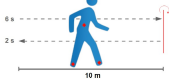
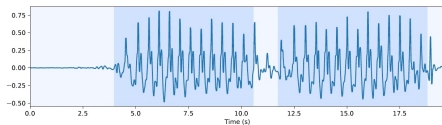
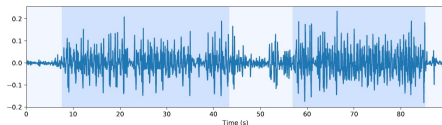


Figure 1.2: Scheme of the protocol used for the analysis of the human gait. Red dots indicate sensor positions.



(a) Healthy control subject.



(b) Osteoarthritis patient.

Figure 1.3: Vertical acceleration ( $\text{m/s}^2$ ) of the lower back sensor for two different subjects. Alternating colours mark the consecutive phases: "Stand", "Walk", "Turnaround", "Walk" and "Stop".

From [Truong et al. A Data Set for the Study of Human Locomotion with Inertial Measurements Units (IPOL, 2019)]

→ More details in Wednesday's workshop by Laurent Oudre!

## Time series are complex data

- Measure the behavior of complex systems/phenomena: biological, economic, industrial...
- Multivariate, multiscale, heterogeneous, multimodal, hidden variables/structures
- Potentially massive (e.g. sound : sampling frequency 44.1 kHz)
- Measurements in the physical world inherit of sensor/channel failures: noise, missing data, drift...
- Harder to index than video or images

## Typical problems

- Low-level (preprocessing) tasks: denoising, imputation, segmentation...
- High-level (decision) tasks: classification, prediction, event detection...
- Indirect (explainability) tasks: structure inference, fine quantification, interpretation...

# What Machine Learning may bring to Signal Processing



# Obvious contributions of Machine Learning to Signal Processing

- Increased **predictive power** thanks to nonparametric and/or nonlinear and/or high dimensional models
- Novel **estimation strategies** motivated by structural assumptions (sparse, low-rank...)
- Inspiring **optimization formulations** and algorithms to solve low-level or indirect tasks
- Mathematical theory?

## ML may also help on low-level tasks

- Video **denoising** without flow estimation with a deep convolutional network [Tassano, Delon, Veit (CVPR'2020)]
- **Imputation** using graph inference based on graph signal data using structured sparsity [Humbert, Le Bars, Oudre, Kalogeratos, Vayatis (under revision, 2020)]
- **Segmentation** with kernel methods under weak supervision [Truong, Oudre, Vayatis (ICASSP'2019)]

# Main topics for this talk

- A. Machine Learning in a nutshell
- B. Some straightforward applications
  - B.1. Predictive modeling and feature design
  - B.2. Sparse modeling with signal data
- C. Machine Learning strategies to face specific issues
  - C.1. Interpretability → representation learning
  - C.2. Taking advantage of weak supervision → metric learning
  - C.3. Model drift → transfer learning

# A. Machine Learning in a nutshell

# The goal of (supervised) machine learning

## Finding a function

- Example : Pedestrian detection from video cameras



- What is the search space for such a function?
- Complexity of learning: how many samples to find *the* function?

## Modeling (supervised) classification data

- Data sample:  $\{(X_i, Y_i) : i = 1, \dots, n\}$  where
  - $X_i$ 's encode images (pixel-wise or feature-wise)
  - $Y_i$ 's are binary values (presence/absence of a pedestrian)
- Probabilistic view on the pairs  $(X_i, Y_i)$ 
  - (H1)  $(X, Y)$  **random pair** with distribution  $P$  over  $\mathbb{R}^d \times \{0, +1\}$  (convention)
  - (H2) Data  $\{(X_i, Y_i) : i = 1, \dots, n\}$  are **independent and identically distributed (IID)** random variables with distribution  $P$

# Data, Learning, Prediction

- Training data:  $(X_1, Y_1), \dots, (X_n, Y_n)$  where  $X_i$ s are input data,  $Y_i$ s are labels
- Prediction objective: given a new  $X$ , predict  $Y$
- Learning objective: estimate a function  $\hat{f}_n$  which fits the data on average and makes predictions  $\hat{f}_n(X)$  hopefully fitting  $Y$  on average
- Examples of prediction objectives in supervised learning:
  - Classification:  $Y$  is discrete
  - Regression:  $Y \in \mathbb{R}$
  - Bipartite ranking or scoring: binary  $Y$ 's but the goal is to rank the  $X$ s according to  $\mathbb{P}\{Y = 1 \mid X = x\}$  rather than predicting  $Y$
  - Other prediction problems: multilabel, multitask, structured...

## Decision rules are those functions

- Decision rules in the case of supervised binary classification

$$h : \mathbb{R}^d \mapsto \{0, 1\}$$

also called *classifiers* among a hypothesis class  $\mathcal{H}$ .

- Examples of hypothesis classes of classifiers:
  - Linear classifiers:

$$h_{\theta, \theta_0}(x) = \mathbb{I}(\theta^T x + \theta_0 \geq 0) , \quad \text{where } \theta \in \mathbb{R}^d, \theta_0 \in \mathbb{R}.$$

- More generally:

$$h_f(x) = \mathbb{I}(f(x) \geq 0)$$

where  $f : \mathbb{R}^d \mapsto \mathbb{R}$  and  $f$  can be implemented by logistic regression, decision trees, boosting, random forests, SVM, neural networks, ...

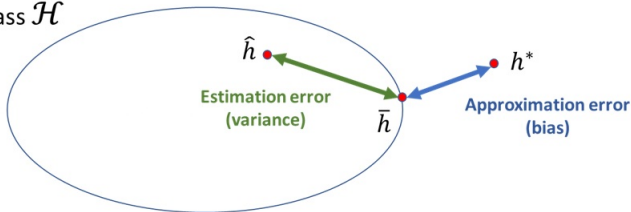


# The key trade-off in Machine Learning

- Denote by  $L(h)$  the error measure for any decision function  $h$
- We have:  $L(\bar{h}) = \inf_{\mathcal{H}} L$  , and  $L(h^*) = \inf L$
- Bias-Variance type decomposition of error for any output  $\hat{h}$  :

$$L(\hat{h}) - L(h^*) = \underbrace{L(\hat{h}) - L(\bar{h})}_{\text{estimation (stochastic)}} + \underbrace{L(\bar{h}) - L(h^*)}_{\text{approximation (deterministic)}}$$

Hypothesis class  $\mathcal{H}$



# The "mother" of ML algorithms

## Penalized optimization

- Learning process as the optimization of a data-dependent criterion:

$$\text{Criterion}(h) = \text{Training error}(h) + \lambda \text{Penalty}(h)$$

- Training error: data-fitting term related to a loss function
- Penalty: complexity of the decision function
- Constant  $\lambda$ : smoothing parameter tuned through cross-validation procedure

# Machine Learning algorithms

1. Local methods:
  - a. k-Nearest Neighbors
  - b. Decision trees (partition-based methods)
  - c. Local averaging (kernel smoothing)
2. Shallow risk-optimization methods
  - a. Boosting
  - b. Support Vector Machines
  - c. Neural networks (one-hidden layer)
3. Ensemble methods
  - + Bagging, Random Forests, Boosting
4. Deep neural networks

## What statistical learning theory says

- Local methods (nearest-neighbors, partition-based): consistency if locality parameter goes to 0 but not too fast (Stone's theorem)
- Risk-optimization based methods (boosting, SVM, shallow neural networks): consistency if
  - control on estimation error under **complexity control** and **IID assumption** on training data
  - **penalized risk optimization** performs regularized estimation and helps monitoring approximation error
- Deep learning methods: consistency if implicit regularization

## Definition of Rademacher complexity

- Consider a sample of  $D_n = (X_1, \dots, X_n)$  of IID random variables, and a vector of Rademacher random variables:  $\varepsilon = (\varepsilon_1, \dots, \varepsilon_n)^T$  with  $\varepsilon_i$ 's IID and independent of the training data such that  $\mathbb{P}(\varepsilon_i = 1) = \mathbb{P}(\varepsilon_i = -1) = 1/2$
- Then the Rademacher complexity of the set of functions  $\mathcal{H}$  is the sample-dependent quantity:

$$\hat{R}_n(\mathcal{H}) = \mathbb{E} \left( \sup_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^n \varepsilon_i h(X_i) \middle| D_n \right)$$

## Generalization error bound for the risk optimization methods

- Loss function:  $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow [0, +\infty]$
- Empirical risk of a decision rule  $h$ : this is a data-dependent functional

$$\widehat{L}_n(h) = \frac{1}{n} \sum_{i=1}^n \ell(h(X_i), Y_i)$$

- ERM = Empirical Risk Minimization

$$\widehat{h}_n = \arg \min_{h \in \mathcal{H}} \widehat{L}_n(h)$$

- Bound on the error: with probability at least  $1 - \delta$

$$L(\widehat{h}_n) \leq \inf_{h \in \mathcal{H}} L(h) + \widehat{R}_n(\mathcal{H}) + 3\sqrt{\frac{\log(2/\delta)}{2n}}$$

## New questions in ML with signal data ?

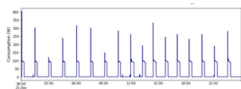
- Same algorithms, same theory (input space can be anything)
- In shallow learning: penalty design can induce specific structures in the function estimates (sparsity, structured sparsity, low rank, etc.)
- In deep learning: representation learning with little supervision (refer to Long Short-Term Memory Neural Networks)
- However, feeding ML with complex data requires huge effort on preprocessing and modeling the data to convert them into vectors (featurization) in order to apply them in the real world! This is the challenge!

## B.1. Featurization and predictive modeling on signal data

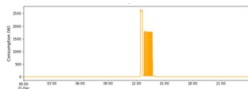


## Example 1 : Classification of distinct signals

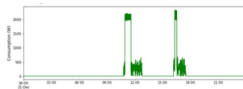
- Each  $X_i$  is a signal and  $Y_i$  is the signal label
- Example : classify appliances based on *distinct* electrical consumption signals



Fridge



Oven



Washing Machine

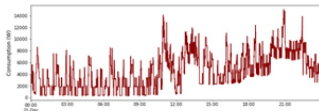
- Main practical issue: how to map (embed) each signal into a vector?

## Hints for featurization

- On one signal: "hand-crafted" approach combining various perspectives
  - coefficients of basis functions: cosines, wavelets, splines
  - statistical indicators: slope, quantiles...
  - geometric/expert-based features: zero-crossing, number of peaks-over-threshold...
- On many signals: spectral methods (with variational formulations) like Principal Component Analysis (PCA) and its variants (robust, sparse...), Nonnegative Matrix Factorization (NMF)...

## Less straightforward...

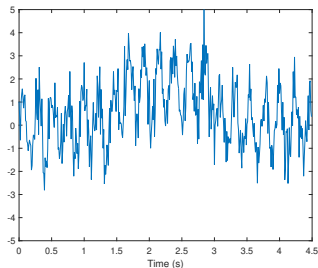
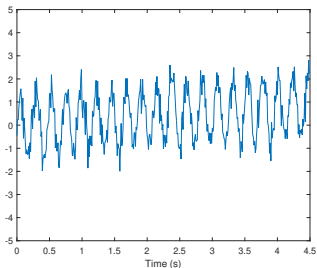
- Case of one signal continuously recorded



House

- Modeling choices?
  - What is the label  $Y$ ?
  - If the  $X$ s are portion of signal over time, how to segment them?
- Theoretical foundation: What about the IID assumption?  
Breaks of stationarity and overlapping segments question this assumption...

# About the (lack of) stationarity in time series



Smooth evolution vs. abrupt change

## How to deal with breaks of stationarity

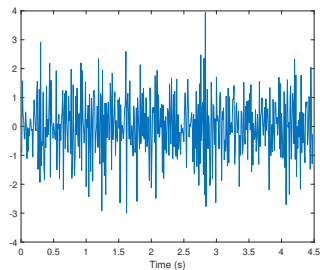
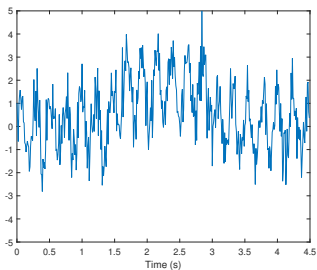
- Drift: Break of stationarity may be corrected after *detrending* the signal
- Abrupt changes: Divide the signal into small frames on which the signal is assumed to be stationary, amounts to performing signal *segmentation*
- A trick: Instead of working on the original signal  $x[n]$ , use the signal of successive differences:

$$x'[n] = x[n] - x[n - 1]$$

which in general has nicer stationarity properties

Reference on locally stationary process and macrotile estimation: [Donoho, Mallat, von Sachs, Samuelides (2003)]

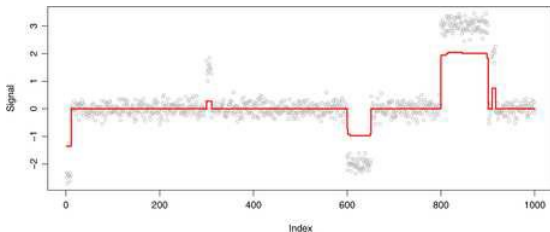
## Example



Use of derivation to make the signal *more stationary*

## B.2. Sparse modeling with signal data

## From LASSO to Fused LASSO



- Enforcing temporal coherence leads to adding a penalty term:

$$\hat{\beta}(\lambda, \mu) \in \arg \min_{\beta \in \mathbb{R}^d} \left\{ \|Y - X\beta\|^2 + \lambda \|\beta\|_1 + \mu \sum_{j=2}^d |\beta_j - \beta_{j-1}| \right\}$$

- Runs after signal approximation, achieves changepoint detection



# C.1. Learning sparse representations of a signal

## Motivations

- Some features (to represent the data) may be good for compression but not for interpretation (and vice versa); they may also simply fail to "lead to" sparse representations (e.g., learn functions that use only a few of the features)
- Can we learn data features (representation) so that the functions we learn (estimate) in that representation ("space") are also sparse?
- Idea is to exploit the fact that *similar patterns may be repeated in the data (even if they are not smooth)*
- (Can also be used to handle some cases of non-stationarity)

References: Olshausen and Field (1997) Kreutz-Delgado et al. (2003), Mairal, Elad, Sapiro (2008), Gribonval et al. (2015)

# Notion of representation

- In order to learn from time series, it is necessary to study them in the adequate representation space
- Frequency domain was especially adapted to study time signals...
- ... but it is not the only one !
- Now, besides off-the-shelf models, it is possible to **learn** the representation directly from the time series

## Dictionary decomposition

- In many cases, the time series  $x[n]$  is represented as a linear combination of several functions, that are stored into a **dictionary**
- $K$  : number of functions in the dictionary
- $\{d_k[n]\}_{1 \leq k \leq K}$  : functions in the dictionary (also called **atoms**)

$$x[n] = \sum_{k=1}^K z_k d_k[n]$$

## Dictionary decomposition

$$\begin{pmatrix} x[0] \\ x[1] \\ \vdots \\ \vdots \\ \vdots \\ x[N-1] \end{pmatrix} = \begin{pmatrix} d_1[0] & d_2[0] & \cdots & d_K[0] \\ d_1[1] & d_2[1] & \cdots & d_K[1] \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ d_1[N-1] & d_2[N-1] & \cdots & d_K[N-1] \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_K \end{pmatrix}$$

$$x = Dz$$

- $K$  : number of functions in the dictionary
- $D$  : dictionary
- $z$  : activation coefficients

## Looking for sparse representations

- If  $K \geq N$ , the dictionary is likely to contain redundancy, so a large number of  $z_k$  will be close to 0
- Finding a "good" representation boils down to finding a sparse activation vector
- Finding a sparse activation vector  $z$  from an input signal  $x$  and a dictionary  $D$  is called **sparse coding**

## Sparse coding - assume D is fixed

- $\ell_0$ -constraint:

$$z^* = \underset{\substack{z \\ \|z\|_0 = K_0}}{\operatorname{argmin}} \|x - Dz\|_2^2$$

Resolution: Iterative Hard Thresholding [Blumensath et al., 2008], Matching Pursuit [Mallat et al., 1993]

- $\ell_1$ -relaxation:

$$z^* = \underset{z}{\operatorname{argmin}} \|x - Dz\|_2^2 + \lambda \|z\|_1$$

Resolution: Iterative Soft Thresholding Algorithm (ISTA) [Daubechies et al., 2004]

## Other approaches for $\ell_1$ -regularization

- **FISTA** : Fast Iterative Soft Thresholding Algorithm [Beck et al., 2009]  
Additional step after the soft-thresholding step based on Nesterov momentum to accelerate gradient descent
- **ADMM** : Alternate Direction Method of Multiplier [Boyd et al., 2011]  
Introduction of auxiliary variables to split the problem into one data fitting term and one penalty term that are optimized independently
- **LARS** : Least angle regression [Efron et al., 2004]  
Iterative addition of activations linked to the atoms that are the most correlated with the residual and increasing of all chosen activations in their joint least squares direction
- **CD** : Coordinate descent  
Update each activation with closed-form solution until convergence



## Which choice for the dictionary?

- Depends on the signal, should be based either on expertise or on maths
- Classical choices:
  - **Fourier basis** (or DCT for real signals): link to DFT + most signals are bandlimited + handles the sinusoidal and seasonality component
  - **Polynomial functions**: handle the trend component
  - **Wavelet dictionary**: Haar, Daubechies,... handle multi-scales + localized phenomenon
- Other option: *Learn* the dictionary using a bunch of  $M$  signals

# Notations

- Input signals is a matrix  $X$  composed of the  $M$  signals

$$X = \begin{pmatrix} x_1[0] & x_2[0] & \cdots & x_M[0] \\ x_1[1] & x_2[1] & \cdots & x_M[1] \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ x_1[N-1] & x_2[N-1] & \cdots & x_M[N-1] \end{pmatrix}$$

- Dictionary  $D$  is still of size  $N \times K$ , but the activation vector becomes an activation matrix  $Z$  of size  $K \times M$

$$D = \begin{pmatrix} d_1[0] & d_2[0] & \cdots & d_K[0] \\ d_1[1] & d_2[1] & \cdots & d_K[1] \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ d_1[N-1] & d_2[N-1] & \cdots & d_K[N-1] \end{pmatrix} \quad Z = \begin{pmatrix} z_{1,1} & z_{1,2} & \cdots & z_{1,M} \\ z_{2,1} & z_{2,2} & \cdots & z_{2,M} \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ z_{K,1} & z_{K,2} & \cdots & z_{K,M} \end{pmatrix}$$

## Dictionary learning

$$D^* = \underset{\forall k, \|d_k\|_2 \leq 1}{\operatorname{argmin}} \|X - DZ\|_F^2$$

- Normalization constraint to prevent D from being arbitrarily large
- Convex problem with fixed X and Z
- Several solvers: Proximal Gradient Descent, block coordinate descent, K-Singular Value Decomposition (K-SVD), ADMM... but also online dictionary learning method that process data on the fly [[Mairal et al., 2009](#)]

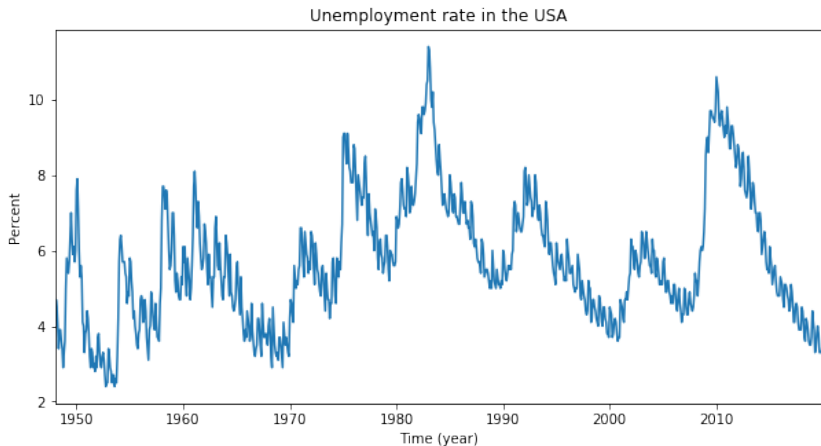
## Learning activations AND the dictionary

- In the general context the activations are unknown and we need to solve the global problem

$$(D^*, Z^*) = \underset{\substack{\forall k, \|d_k\|_2 \leq 1 \\ \forall k, z_k \text{ is sparse}}}{\text{argmin}} \|X - DZ\|_F^2$$

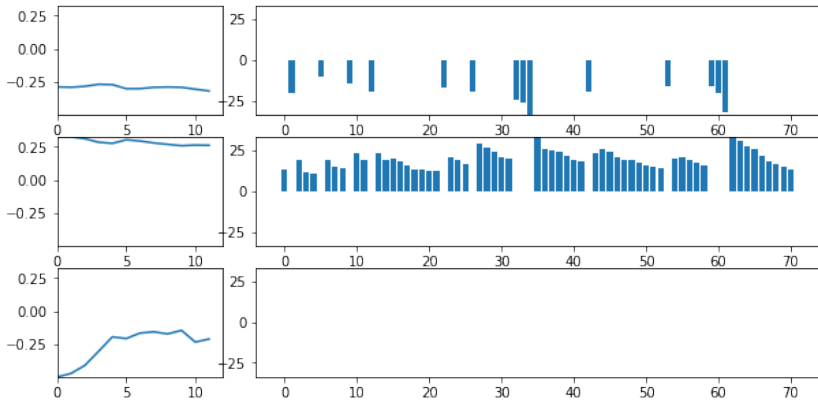
- This problem cannot be solved as such: in most cases we alternate between two subproblems
  - Sparse coding with known D
  - Dictionary learning with known Z
- Alternated resolution until the norm of the residual becomes small enough

## Macroeconomic series example (1/5)



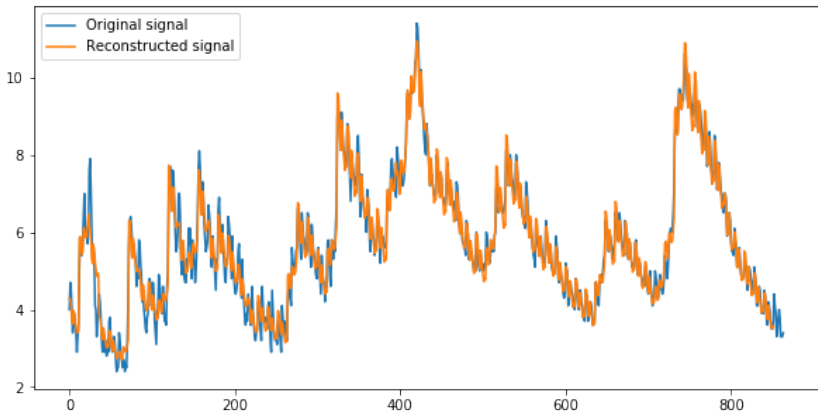
72 years, 1 sample per month: transformation into 72  
non-overlapping frames of 12 samples

## Macroeconomic series example (2/5)



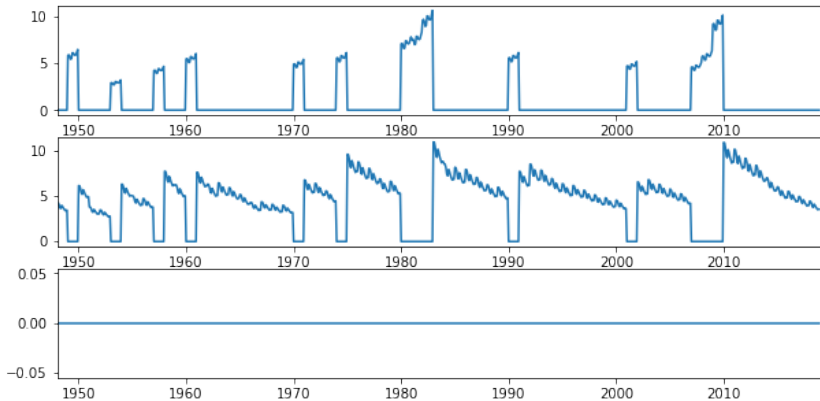
$K = 3$  atoms,  $\ell_0$ -sparse coding with  $K_0 = 1$

## Macroeconomic series example (3/5)



Reconstruction by using all 3 atoms

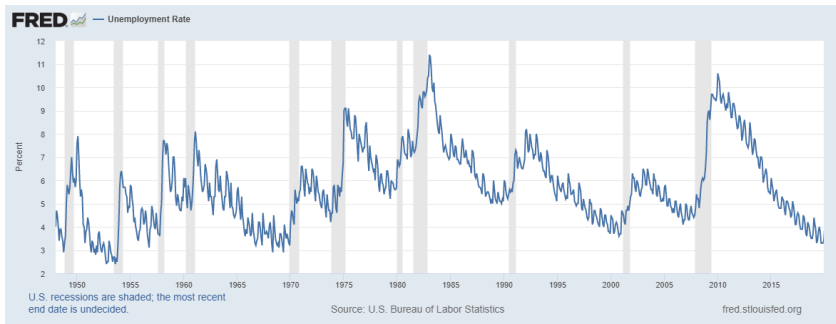
## Macroeconomic series example (4/5)



Reconstruction by using each atom individually



# Macroeconomic series example (5/5)



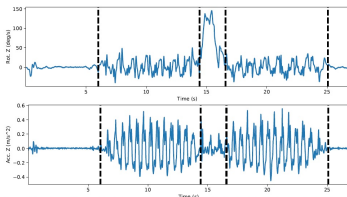
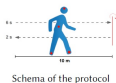
All recession periods are captured by the first atom

## C.2. Metric learning

# Motivation: activity recognition (1/2)

Context: activity recognition with accelerometric signals recording gait during a walk protocol

Use case.

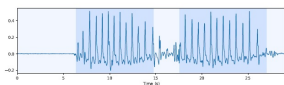


Signal example, from a walking exercise. Rot. Z (top) and Acc. Z (bottom). Barrois-Müller, et al. (Clinical Neurophysiology, 2016)

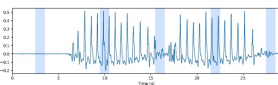
- First step: signal segmentation (unsupervised task)
- Second step: featurization of each segment ("manual" task)
- Third step: learn a classifier (supervised task)

## Motivation: activity recognition (2/2)

- Focus on first step: assume supervision of each activity



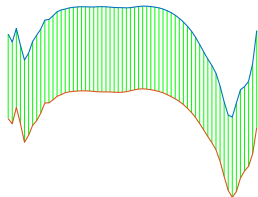
Example of a fully annotated signal



Example of a partially annotated signal

- Direct use of labels to validate segmentation requires full annotation
- Partial annotation may be used to improve accuracy on segmentation by metric learning

## Comparing signals: the limitations of Euclidean distance

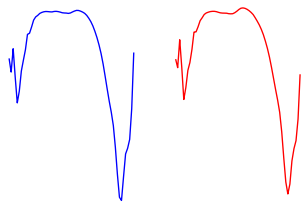


$$d_{EUC}(x, y) = \sqrt{\sum_{n=1}^N (x[n] - y[n])^2}$$

- Sensitive to time shifts, amplitude changes, offsets and dilatation/contraction
- Necessity to have a perfect match between the timelines
- Sensitive to outliers

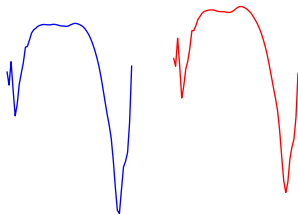
# What influences Euclidean distance?

Baseline case



$$d_{EUC} = 3.5$$

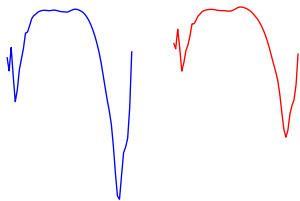
Offset



$$d_{EUC} = 9.6$$

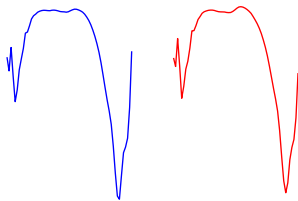
# What influences Euclidean distance?

Amplitude shift (70 %)



$$d_{EUC} = 12.8$$

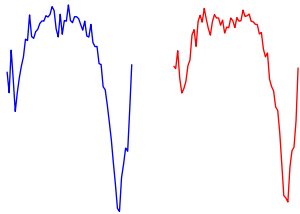
Time shift (1.6 %)



$$d_{EUC} = 5.3$$

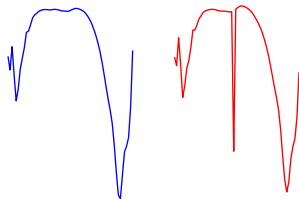
# What influences Euclidean distance?

Additive white Gaussian noise



$$d_{EUC} = 6.9$$

Outliers



$$d_{EUC} = 10.3$$



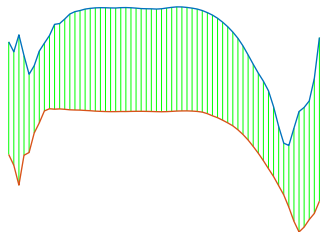
# A classical alternative to Euclidean distance

## **New notion of distance : Dynamic Time Warping (DTW)** [Berndt et al., 1994]

- To decrease the sensitivity with respect to timelines (contraction/dilatation, time shifts...)
- To be able to compare two time series of different lengths
- To take into account nonlinear timeline modifications

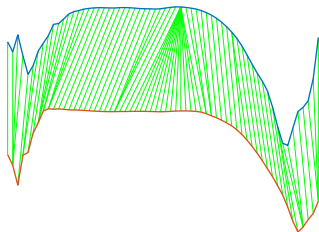
## Differences between Euclidean and DTW

Euclidean distance



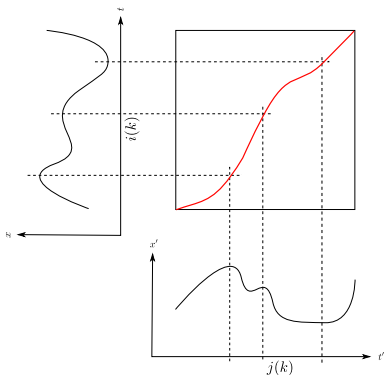
Sample  $x[n]$  is associated to sample  $y[n]$

DTW



Sample  $x[i_k]$  is associated to sample  $y[j_k]$

## Notion of path



- DTW computes a correspondence between the elements of  $x$  and those of  $y$ .
- Mapping function called **path** :

$$P = ((i_1, j_1)), \dots, (i_{K_P}, j_{K_P}) \in (\mathbb{N} \times \mathbb{N})^{K_P}, K_P \in \mathbb{N}$$

$$(i_k, j_k) \in P \Leftrightarrow y[j_k] \text{ is matched with } x[i_k]$$

## Choice of the optimal path

- The path  $P$  is evaluated through a cost function

$$w(P) = \sum_{k=1}^{K_P} (x[i_k] - y[j_k])^2$$

- The final DTW distance is computed as the minimal value for the cost function

$$d_{DTW}(x, y) = \sqrt{\min_{P \in \mathcal{P}} w(P)}$$

- Note that if  $K_P = N$  and  $i_k = j_k = k$ , DTW is exactly equal to the Euclidean distance
- For practical implementation, build acceptable paths

## Limitations of DTW

- DTW needs that the first and last samples are aligned, which can be a limitation in case of time shifts. Some variants exist to avoid this problem.
- DTW is sensitive to scale (can be solved by normalized time series beforehand), noise and outliers

# ML at rescue: Metric learning

## Principle

- Mahalanobis pseudodistance between  $T$ -dimensional vectors  $x, x'$  defined as:

$$d_M(x, x') = \sqrt{(x - x')^T M (x - x')}$$

where  $M$  is a square symmetric positive semidefinite (PSD) matrix such that, for any  $x$ ,  $x^T M x \geq 0$  ( $M \succeq 0$ ).

- Denote by  $\mathbb{S}_+^T$  the cone of symmetric PSD real-valued matrices. For any  $M \in \mathbb{S}_+^T$  there exists a matrix  $L$  such that  $M = L^T L$ . Note that if  $\text{rank}(M) = r$  then  $L$  is rectangular  $r \times T$ .

# Original metric learning formulation

[Xing, Ng, Jordan and Russell (2002)]

- Setup: Consider  $n$  signals  $X_1, \dots, X_n$  in  $\mathbb{R}^T$  and assume we have labels on pairwise comparisons:  $Y_{ij} = 1$  if  $X_i$  and  $X_j$  are similar, and  $Y_{ij} = 0$  otherwise (or if we do not know).
- Formulation: Consider the optimization problem

$$\min_{M: M \succeq 0} \sum_{i \neq j} Y_{ij} d_M^2(X_i, X_j) \text{ such that } \sum_{i \neq j} (1 - Y_{ij}) d_M^2(X_i, X_j) \geq 1$$

- Resolution: The problem is convex but expensive to solve with Newton's method if  $M$  is of full rank. It may be solved efficiently with projected gradient descent. Still requires  $O(T^3)$  operations to project on the cone  $\mathbb{S}_+^T$ . Other variants try to avoid PSD constraint.

For more details: [Bellet, Habrard, Sebban (2014)] or [Suarez-Diaz, Garcia, Herrera (2020)]

## Results obtained

- Two changepoint detection algorithms have been applied to detect transition between activities on gait signals:
  - greedy kernel based changepoint detection
  - test-based changepoint detection with a sliding window
- Accuracy on average (standard deviation) in seconds for the two methods

	♡gkCPD	gkCPD	♡Win ( $c_{rbf}$ )	Win ( $c_{rbf}$ )
Stand/Walk	0.47 ( $\pm 0.42$ )	0.76 ( $\pm 0.99$ )	5.31 ( $\pm 2.86$ )	5.28 ( $\pm 2.53$ )
Walk/Turnaround	0.61 ( $\pm 1.23$ )	1.07 ( $\pm 2.04$ )	1.01 ( $\pm 1.42$ )	1.12 ( $\pm 1.33$ )
Turnaround/Walk	0.94 ( $\pm 1.83$ )	1.14 ( $\pm 2.04$ )	1.83 ( $\pm 2.52$ )	2.40 ( $\pm 2.59$ )
Walk/Stand	0.53 ( $\pm 0.36$ )	0.62 ( $\pm 0.42$ )	1.84 ( $\pm 2.60$ )	2.16 ( $\pm 2.62$ )



## Other applications of metric learning in SP

- Clustering with side information
- Classification with nearest-neighbor algorithms
- Application of various-purpose kernel-based algorithms (like Support vector Machines)
- Pattern matching

# C.1. Transfer learning

# Motivation: fall detection in nursing homes (1/2)

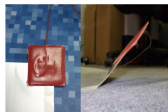
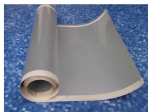
- Industrial application: fall detection technology incorporated in a flooring element

- ▶ Piezoelectric principle:

$$d = \frac{Q}{F},$$

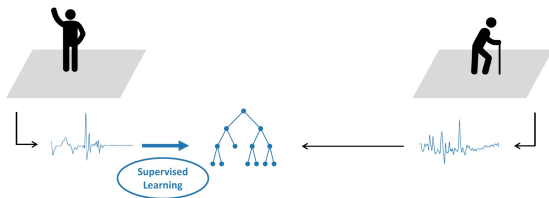
(simple version) with  $d$  the *piezoelectric constant*.  
When stressed or squeezed, the material emits charges.

- ▶ How does this look like ?  
0.3 mm thick and 60 cm wide roll with customizable length
- ▶ How is it installed ?
  - ▶ Under the flooring
  - ▶ Several connected bands for each area, hence one area corresponds to one input



## Motivation: fall detection in nursing homes (2/2)

- Key challenge: from the lab to the field!



- Key observation: signals have different characteristics because the populations are different
- Real data are not easy to annotate and falls are rare
- Is it possible to *transfer* knowledge from the lab (source domain) to the field (target domain)?

# ML at rescue: Transfer Learning

- Focus here on transductive transfer learning, also known as *Domain Adaptation*
- Assumption: we can learn a same task from data on *source* domain and *target* domain which have different distributions
- Typically, we have no or few labelled data on target domain

## Theoretical setup for domain adaptation

- Consider labeling functions  $f$ ,  $g$  and the symmetric loss  $\ell$  over pairs of labels which obeys the triangle inequality.
- We introduce the expected loss over any marginal distribution  $Q$  by:

$$L_Q(f, g) = \mathbb{E}_Q(\ell(f(X), g(X)))$$

- Consider a hypothesis class  $\mathcal{H}$  and the marginal distributions  $S$  on source domain and  $T$  on target domain, then the discrepancy distance between these two is defined as:

$$\Delta(S, T) = \max_{h, h' \in \mathcal{H}} |L_S(h', h) - L_T(h', h)|$$

## Domain adaptation bound Mansour et al. (2009)

- Consider the true labeling functions  $f_S^*$  and  $f_T^*$  for each distribution and define the best in the class by:

$$\bar{h}_Q \in \arg \min_{h \in \mathcal{H}} L_Q(h, f_Q^*), \quad \forall Q \in \{S, T\}$$

- Then we have, for any  $h \in \mathcal{H}$ :

$$L_T(h, f_T^*) \leq L_T(\bar{h}_T, f_T^*) + L_S(h, \bar{h}_S) + \Delta(S, T) + L_S(\bar{h}_S, \bar{h}_T)$$

- From there, empirical generalization error bounds on the excess of risk  $L_T(h, f_T^*) \leq L_T(\bar{h}_T, f_T^*)$  can be derived in terms of empirical loss of  $h$ , discrepancy between empirical distributions and Rademacher average of the hypothesis class  $\mathcal{H}$ .

## What domain adaptation theory says

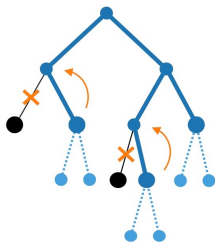
- The driving element to monitor the error on the target domain is the distance between source and target domains marginal distributions (expected!).
- Provides guarantees for most practical strategies:
  - Instance-based (with reweighting)
  - Feature-based (with projections)
  - Model-based

Reference: Book by [Redko, Habrard, Morvant, Sebban, Bennani (2019)]



# An example of model-based transfer on decision trees

- SER: Structure Expansion and Reduction

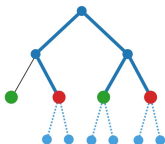


1. Expansion
2. Reduction

- Idea: train on source domain, extend on target domain the active nodes, then cut the inactive edges
- Reference: [Segev, Harel, Mannor, Crammer, El-Yaniv (2016)]

# A practical solution to the fall detector problem

- SER has to be adapted to take into account class imbalance (few falls) with conditional reduction



Minority class

- ▶ Expansion left unchanged
- ▶ Reduction constrained

SER <sub>R</sub>	SER <sub>LL</sub>
If node is of minority class, then no pruning	If node is of minority class <b>and</b> still significant considering Target <b>and</b> $R_L > 0.5$ , then no pruning

- Idea: preserve nodes from minority class (condition may be relaxed depending on the representation of the node in target domain)
- Reference: [Minvielle, Atiq, Peignier, Mougeot (2019)]

# Conclusion

## Findings

- Preprocessing is a noble task. Indeed, ML may perform denoising, segmentation, etc.
- Thanks to nonconvex optimization formulations, it is possible to build data-driven and meaningful representations for signals which may serve for several purposes.
- ML ideas such as incorporating soft supervision may improve unsupervised tasks such as segmentation.
- ML may help to deal with breaks of stationarity (e.g. with domain adaptation).

## Resources

- MVA course by Laurent Oudre

<http://www.laurentoudre.fr/ast/>

- *ruptures*: Python package for changepoint detection

<https://centre-borelli.github.io/ruptures-docs/>

- Human locomotion data set on IPOL

<http://www.ipol.im/pub/art/2019/265/>

- Reproducible research on IPOL

Statokinesigram example:

<https://www.ipol.im/pub/art/2019/251/>

Audio example: <http://www.ipol.im/pub/art/2015/64/>