Advanced Research
Intelligent
Embedded
Systems

Learning in Nonstationary Environments

Prof. C. Alippi and Prof. M. Roveri
Politecnico di Milano, DEIB, Italy

IEEE WCCI 2016 TUTORIAL
Vancouver, CANADA, Jul. 24, 2016

Summary

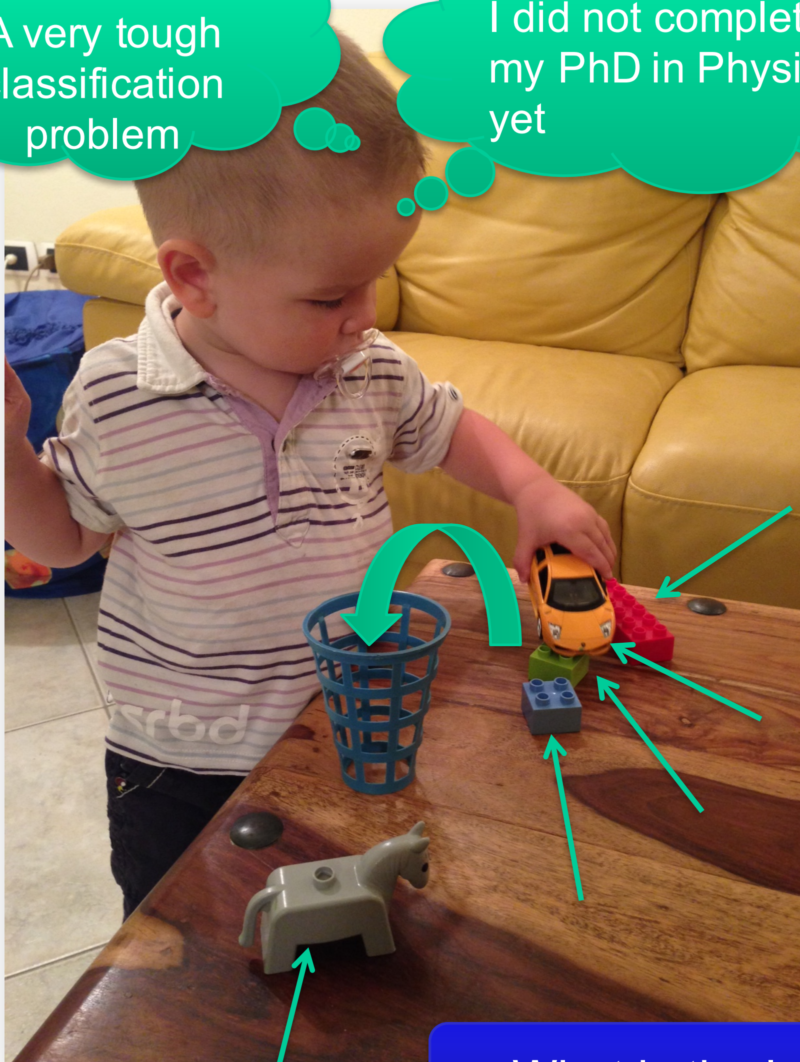
- Learning in nonstationary environments
- Searching for adaptation:
  - Instance selection
  - Instance weighting
  - Model ensemble
- Passive solutions
- Active solutions
- Comments, resources and future trends
- Let's play with Matlab…

A “toy” example

???
A very tough classification problem

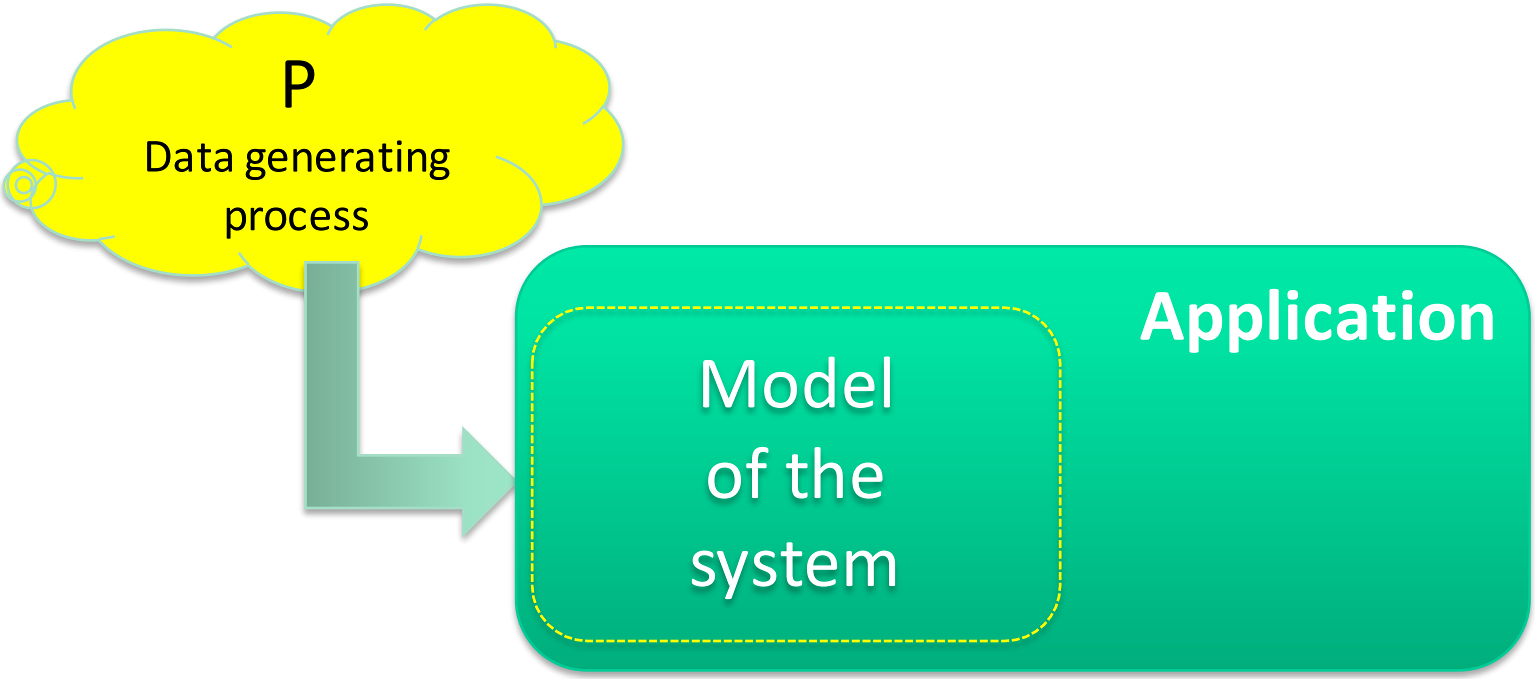
Physical model ?
I did not completed my PhD in Physics yet

Data-driven might be a good solution (brute-force as well)

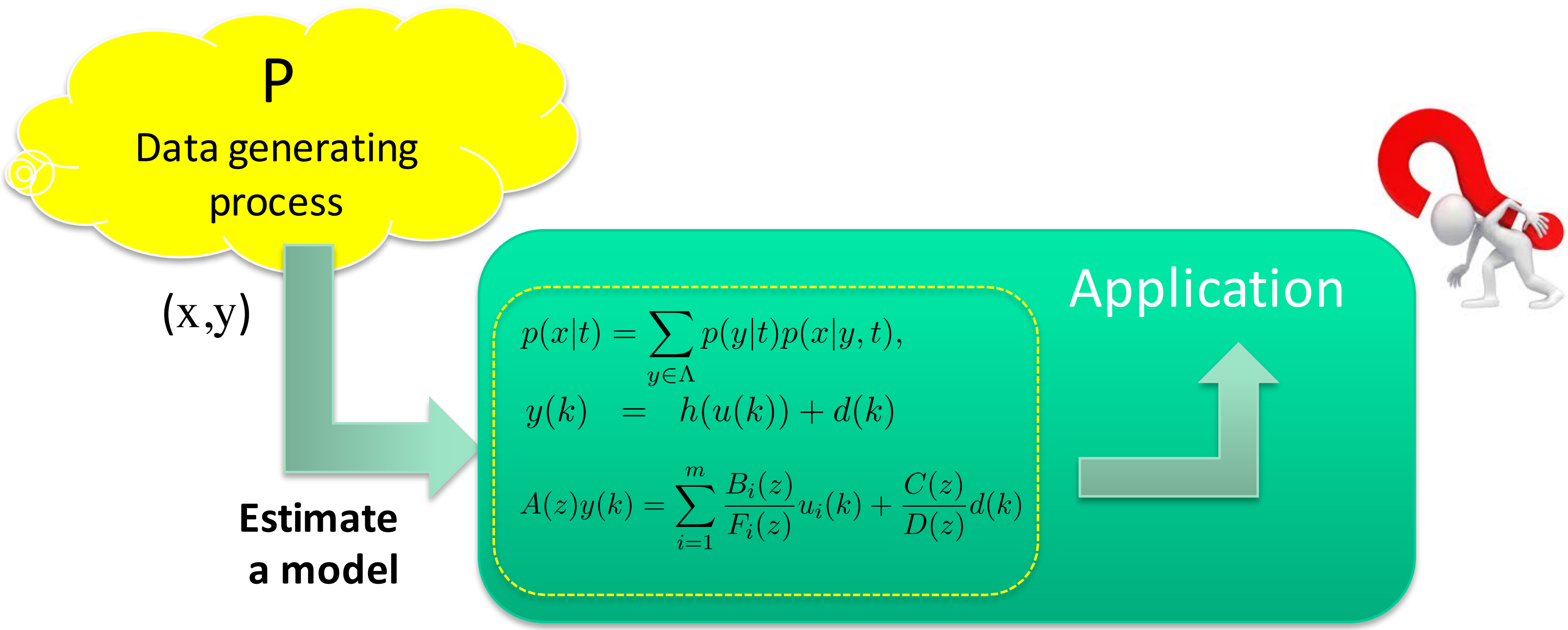


What is the learning goal here?

Data-processing and applications



Learning the system model



$$p(x|t) = \sum_{y\in\Lambda} p(y|t)p(x|y,t),$$

$$y(k) \quad = \quad h(u(k)) + d(k)$$

$$A(z)y(k) = \sum_{i=1}^{m} \frac{B_i(z)}{F_i(z)} u_i(k) + \frac{C(z)}{D(z)} d(k)$$

We will come back to the learning mechanism later

Non-linear regression: statistical framework

The time invariant process generating the data

$$y = g(x) + \eta,$$

provides, given input x_i output instance

$$y_i = g(x_i) + \eta_i$$

We collect a set of couples (training set)

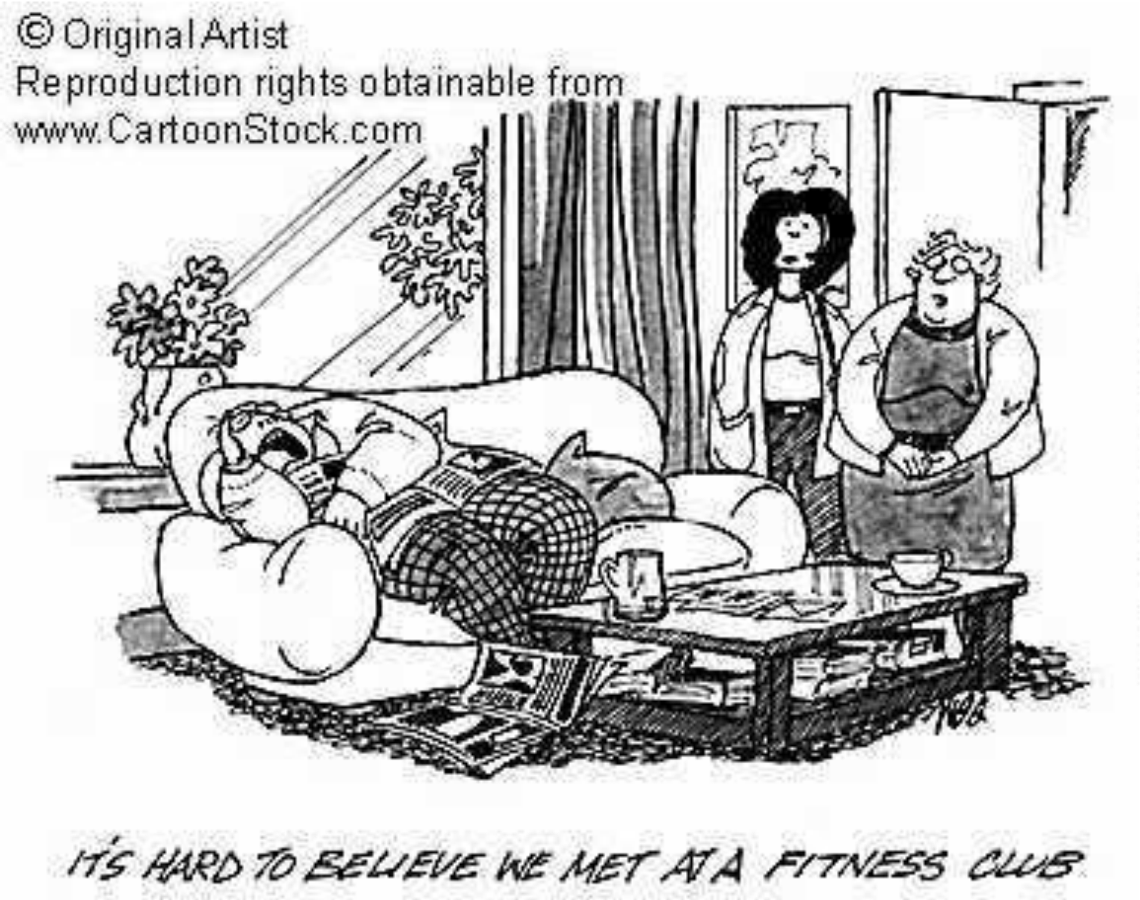
$$Z_N = \{(x_1, y_1), \ldots, (x_N, y_N)\}$$

And wish to model unknown $g(x)$ with parameterized **family of models** $f(\theta, x)$

The goal of **learning** is to build a model able to **explain past data** Z_N and **future instances** provided by the data generating process.

- **Up to now** we assumed the system model to be time invariant…

But everything and everybody changes over time ...



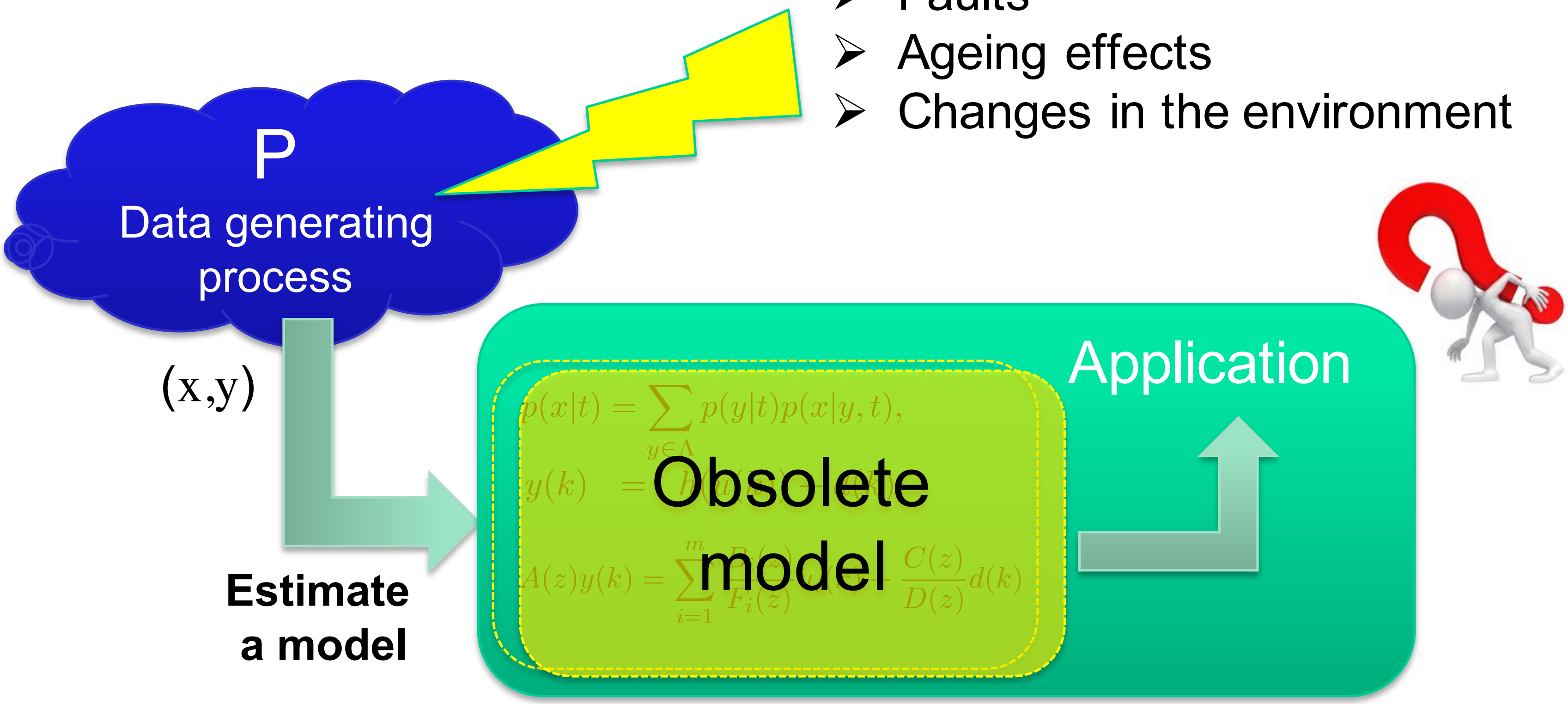
Be aware of *Gradual Concept drift…*

Ageing effects …

… changes in the system or the environment

Learning in Nonstationary Environments: the effect of the non-stationarity

- Faults
- Ageing effects
- Changes in the environment



Perturbed, incorrect and missing data
can hence heavily ***affect the subsequent processing phase***
so as to possibly induce wrong decisions or on-the-field reactions.

Stationarity and time invariance

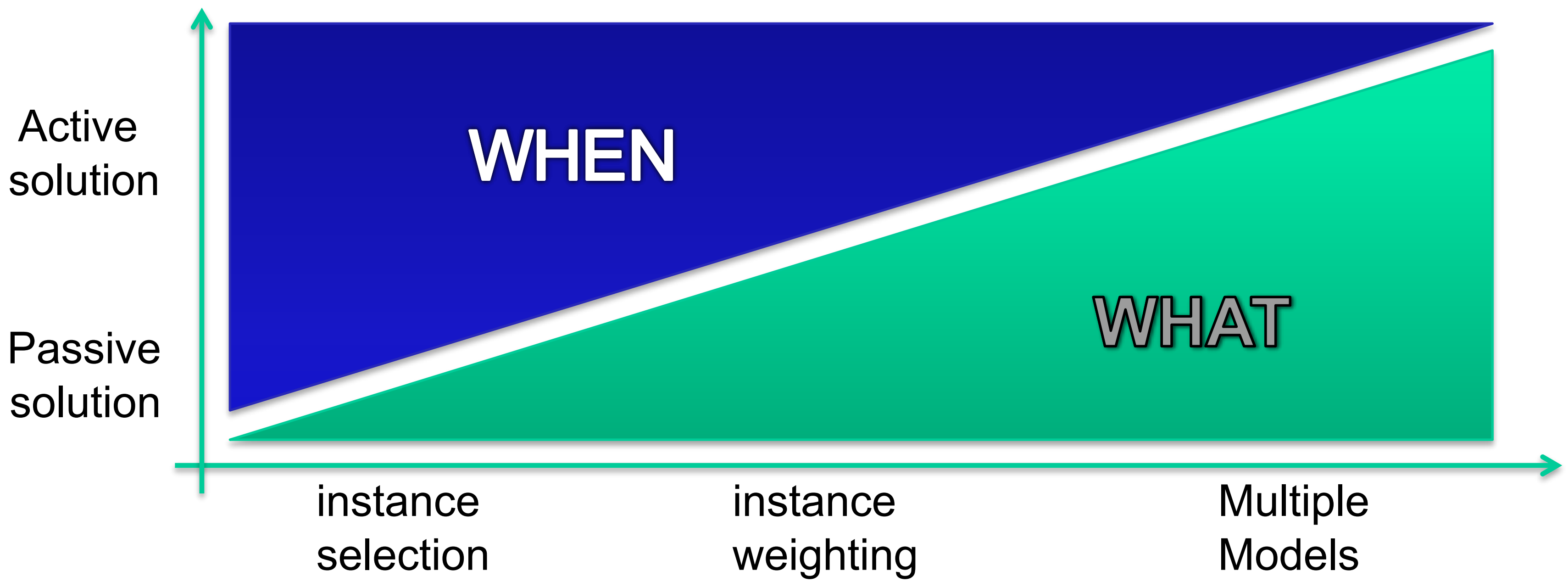
- Stationarity
  - We say that a data generating process is stationary when generated data are i.i.d. realizations of a unique random variable whose distribution does not change with time

- Time invariance
  - We say that a process is time invariant when its outputs do not explicitely depend on time

$$y(t) = a_1(e^{t_0-t})y(t-1) + a_2 y(t-2) + \eta, \eta = N(0, \sigma^2)$$

Searching for adaptation

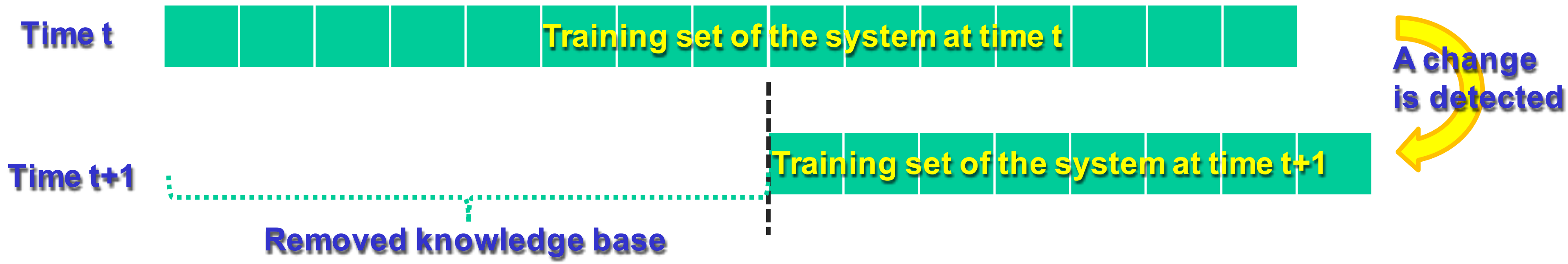
- Traditional assumption: stationarity hypothesis
- Adaptive solutions in a non-stationary framework:



- A comprehensive methodology addressing this problem is not available

WHAT: Instance Selection

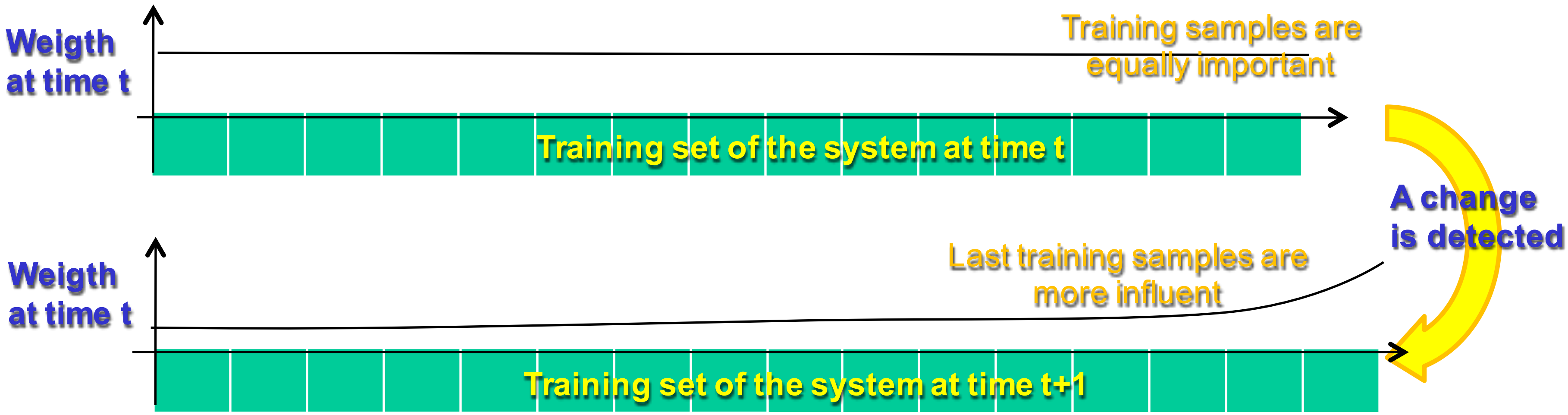
- **The idea**: *identifying the samples of the training set that are relevant to the current state of the process*.



- The adaptive systems generally rely on a window over the most recent training samples to process the upcoming data
  - **fixed window approach:** the length of the window is fixed a-priori by the user
  - **heuristic approaches:** adapt the window length over the latest samples to maximize the accuracy

WHAT: Instance Weighting

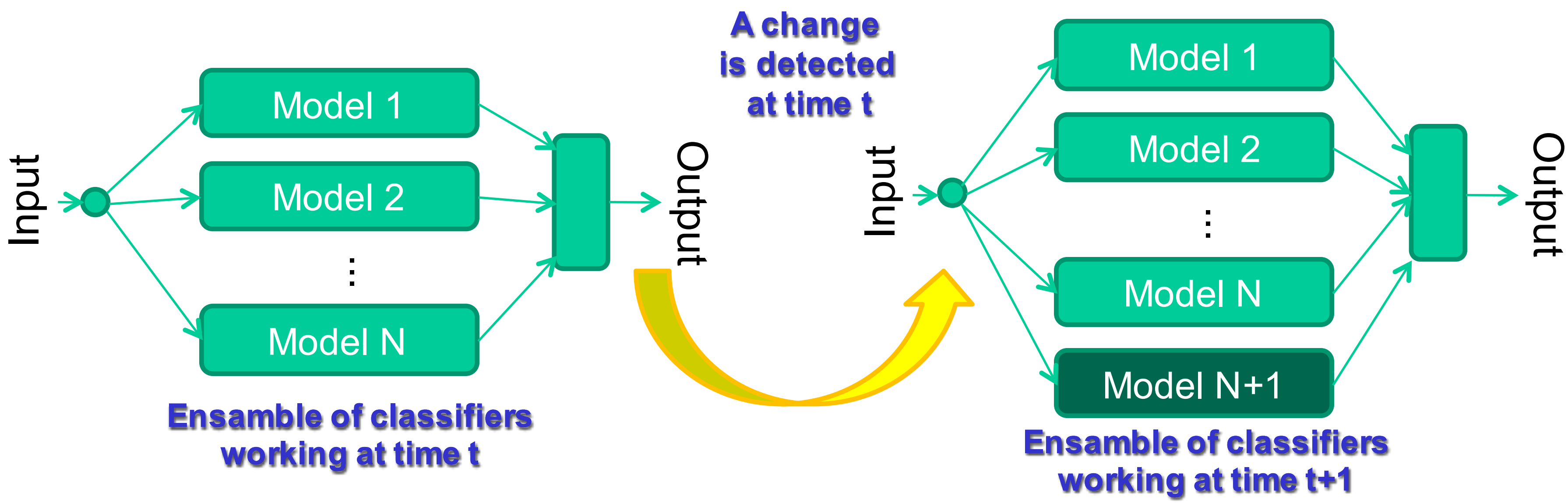
- **The idea**: *training samples are not removed from the training set of the system but all the training samples (suitably weighted) are considered*.



- The training samples might be weighted according to
  - the **age**
  - the **relevancy to the current state** of the process in term of accuracy of the last batch of supervised data

WHAT: Multiple Models

- **The idea**: *the outputs of an ensemble of models are combined by means of voting or weighted mechanisms to form the final output*



- All these systems includes **techniques for dynamically including new models in the system or deleting obsolete ones** (i.e., pruning techniques aiming at removing the oldest model or the one with the lowest accuracy).

Critical analysis of the considered approaches

- **Instance selection**

  ↑: low computational-complexity
  reduced training set

  ↓: fixed windows or heuristics to adapt the window size
  forgetting mechanisms

- **Instance weighting**

  ↑: low computational-complexity
  availability of all the training samples for recurrent models

  ↓: heuristics to define the sample weights
  full training set

- **Multiple models**

  ↑: availability of a model for “each bunch of data”

  ↓: high computational-complexity

WHEN: passive vs active approach

- **Passive solutions** continuously adapt the model without the need to detect the change
  - Ensembles of models with the adaptation phase consisting in a continuous update of the weights of the fusion/aggregation rule and creation/removal of models

- **Active solutions** rely on triggering mechanisms to identify changes in the process and react by updating the model
  - The most popular triggering mechanism is the change detection

Passive approach: the general idea

- The underlying data distributions may (or may not) change at any time with any rate of change.
- A **continuous adaptation** of the model parameters every time new data arrive
- **Maintain an up-to-date model** at all times
  - **Avoid** the potential pitfall associated with **false alarms** in active solutions

Passive learning

Online (incremental) learning

$$V_N(\theta, \{(x_i, y_i)\}) = L(y_i, f(\theta, x_i))$$

$$\theta_{i+1} = \theta_i - \eta \frac{\partial L(y_i, f(\theta, x_i))}{\partial \theta}\Big|_{\theta_i}$$

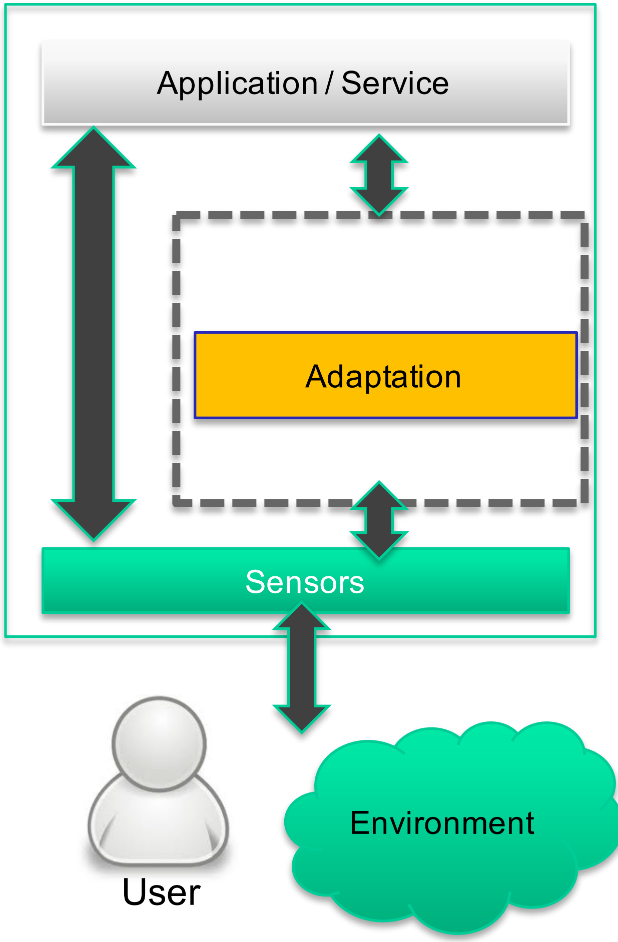
Batch learning

$$Z_{n,i} = \{(x_i, y_i), (x_{i-1}, y_{i-1}), \cdots (x_{i-n+1}, y_{i-n+1})\}$$

$$\theta_{i+1} = \theta_i - \eta \frac{\partial V_N(\theta, Z_{n,i})}{\partial \theta}\Big|_{\theta_i}$$

Ensemble learning

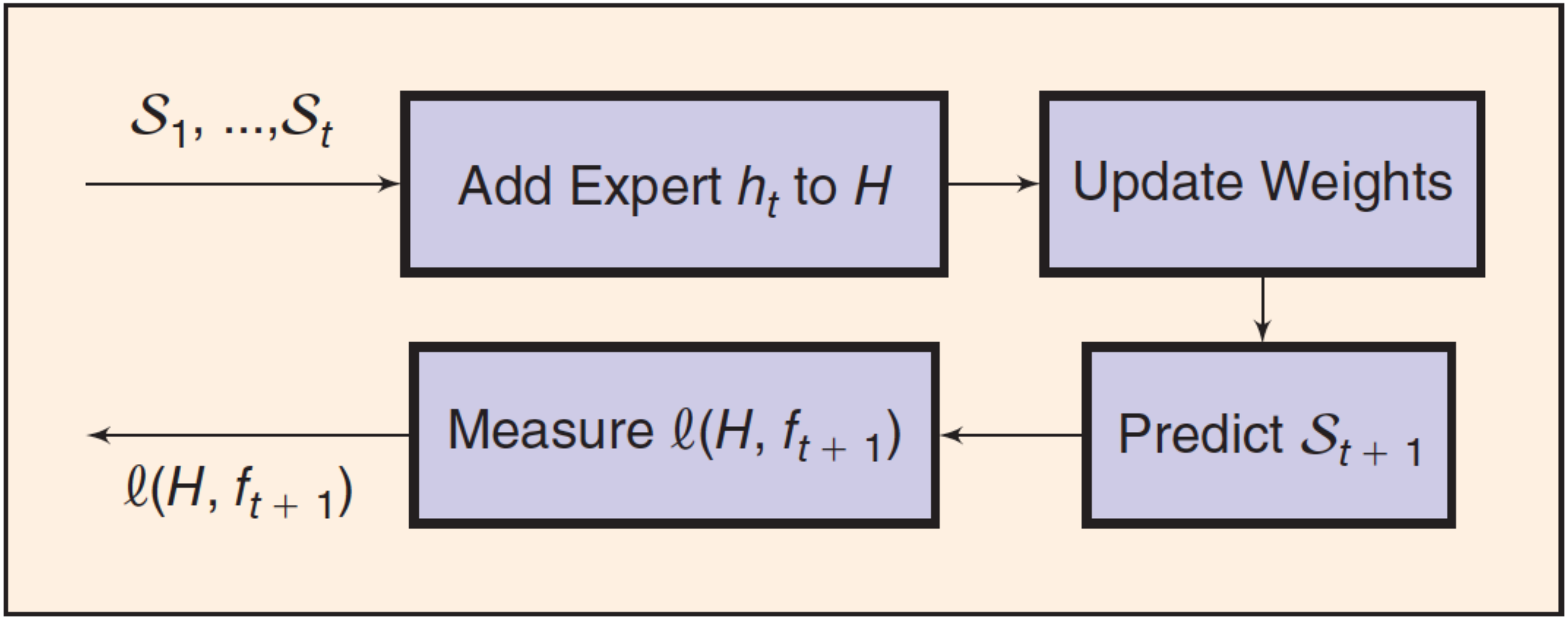
$$y(x) = \sum_{i=1}^{k} w_i M_i(x)$$



Ensemble-based mechanisms

- Ensemble-based approaches provide **a natural fit to the problem of learning in nonstationary environments**:
  a) more accurate than single classifier-based systems
  b) easily incorporate new data into a classification model (a new item into the ensemble)
  c) provide a natural mechanism to forget irrelevant knowledge (removing an item from the ensemble)

Incremental learning ensembles in nonstationary environments



Incremental learning ensembles in nonstationary environments: SEA

- The **streaming ensemble algorithm** (**SEA**) was one of the earliest ensemble approaches:
  - **New classifiers are added** as new batches of data arrive
  - Classifiers are removed as the ensemble reaches a predetermined size
  - **Which classifier must be removed**?
    - evaluation of classifier's predictions
    - age of the classifier
    - remove the least contributing member

Incremental learning ensembles in nonstationary environments: other solutions

- Several popular extensions to online bagging/boosting for nonstationary environments
  - **Online bagging & boosting** form the basis of online nonstationary boosting algorithm (e.g., ONSBoost)
- **Dynamic weighted majority** (DWM) extends weighted majority algorithm to data streams with concept drift, and uses an updated period to add/remove classifiers
- Other approaches:
  - **accuracy updated ensemble** (AUE)
  - **random forest algorithm** has also been extended to learning nonstationary data streams

Learn++.NSE

- Maintain an ensemble that applies **a time-adjusted loss function** to favor classifiers that have been performing well in recent times (not just the most recent ones)
- **A classifier** that performed poorly a long time ago **can be reactivated** (e.g., recurring or cyclic drift)

Input: Datasets $\mathcal{S}_t := \{(\mathbf{x}_i, y_i) : i \in [N_t]\}$, supervised learning algorithm BASE, and parameters a & b.
Initialize: $h_1 = \text{BASE}(\mathcal{S}_1)$ and $W_1^1 = 1$.
1: **for** $t = 2,3,\ldots$ **do**
2: Compute loss of the existing ensemble

$$E_t = \frac{1}{N_t}\sum_{j=1}^{N_t} 1_{H_{t-1}(\mathbf{x}_j) \neq y_j}, \qquad (1)$$

where 1_τ evaluates to 1 if $\tau = \texttt{True}$ otherwise it is 0.
3: Update instance weights

$$D_t(j) = \frac{1}{Z_t}\begin{cases} E_t & H_{t-1}(\mathbf{x}_j) = y_j \\ 1 & \text{otherwise} \end{cases}, \qquad (2)$$

where Z_t is a normalization constant.
4: $h_t = \text{BASE}(\mathcal{S}_t)$
5: Evaluate existing classifiers with new data

$$\varepsilon_k^t = \sum_{j=1}^{N_t} D_t(j) 1_{h_k(\mathbf{x}_j) \neq y_j}. \qquad (3)$$

Set $\beta_k^t = \varepsilon_k^t / (1 - \varepsilon_k^t)$.
6: Compute time-adjusted loss

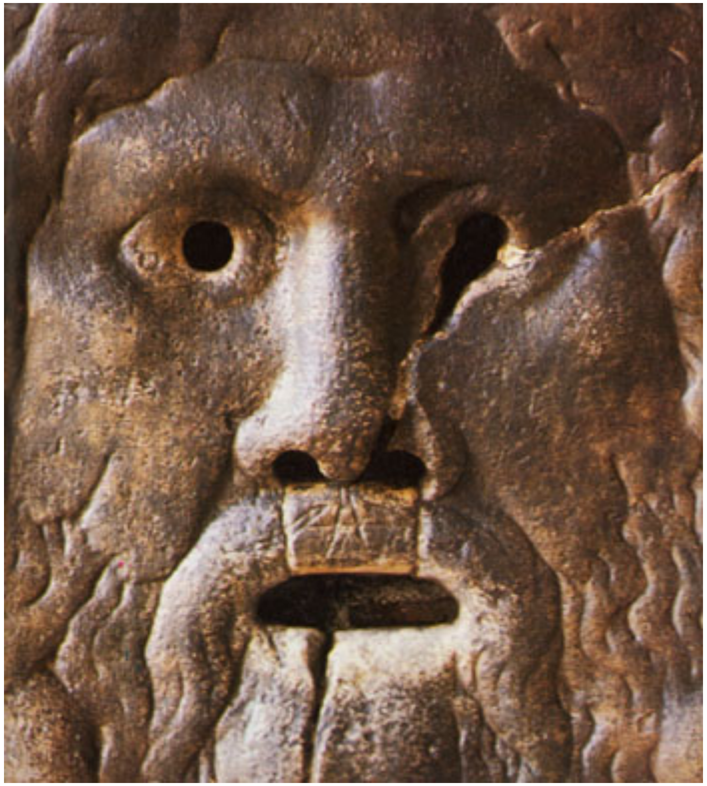
$$\varphi_k^t = \frac{1}{Z_t'}\frac{1}{1 + \exp(-a(t-k-b))}, \qquad (4)$$

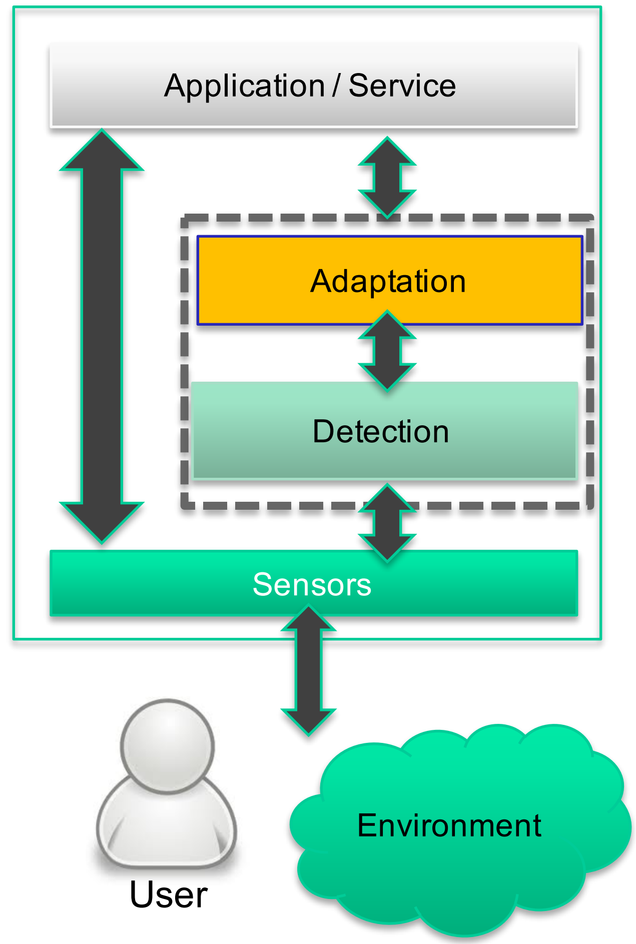
$$\rho_k^t = \sum_{j=0}^{t-k} \varphi_k^{t-j}\beta_k^{t-j}. \qquad (5)$$

7: Update classifier voting weights: $W_k^t = \log \frac{1}{\rho_k^t}$.
8: **end for**
Output: Learn++.NSE's prediction on $\mathbf{x}$

$$H_t(\mathbf{x}) = \arg\max_{\omega \in \Omega} \sum_{k=1}^{t} W_k^t 1_{h_k(\mathbf{x}) = \omega}. \qquad (6)$$

Active learning

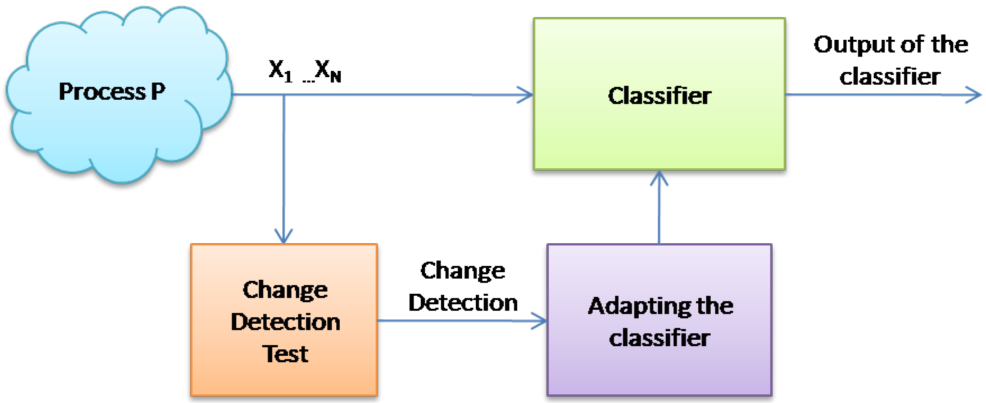




The Oracle provides information about an event, e.g., the occurrence of concept drift

WHEN: Triggering mechanisms

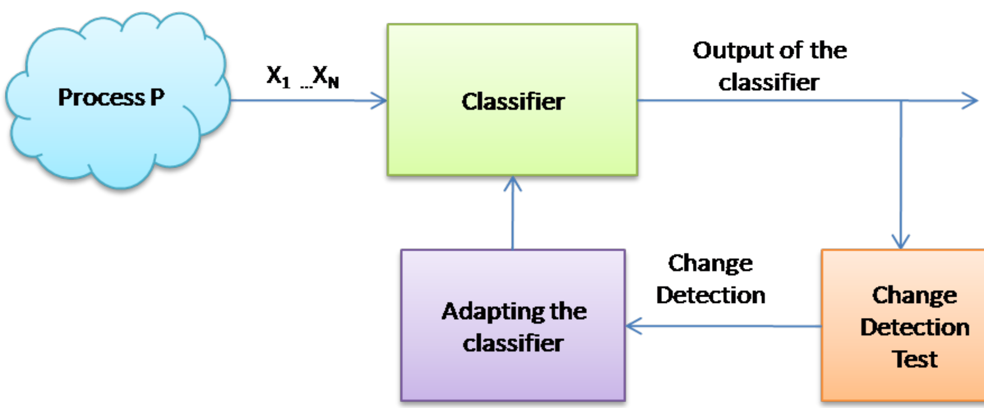
Change detection on the pdf of the inputs:



↑: monitoring the distribution of unlabeled observation

↓: this solution does not allow us for detecting changes that do not affect the distribution of observations

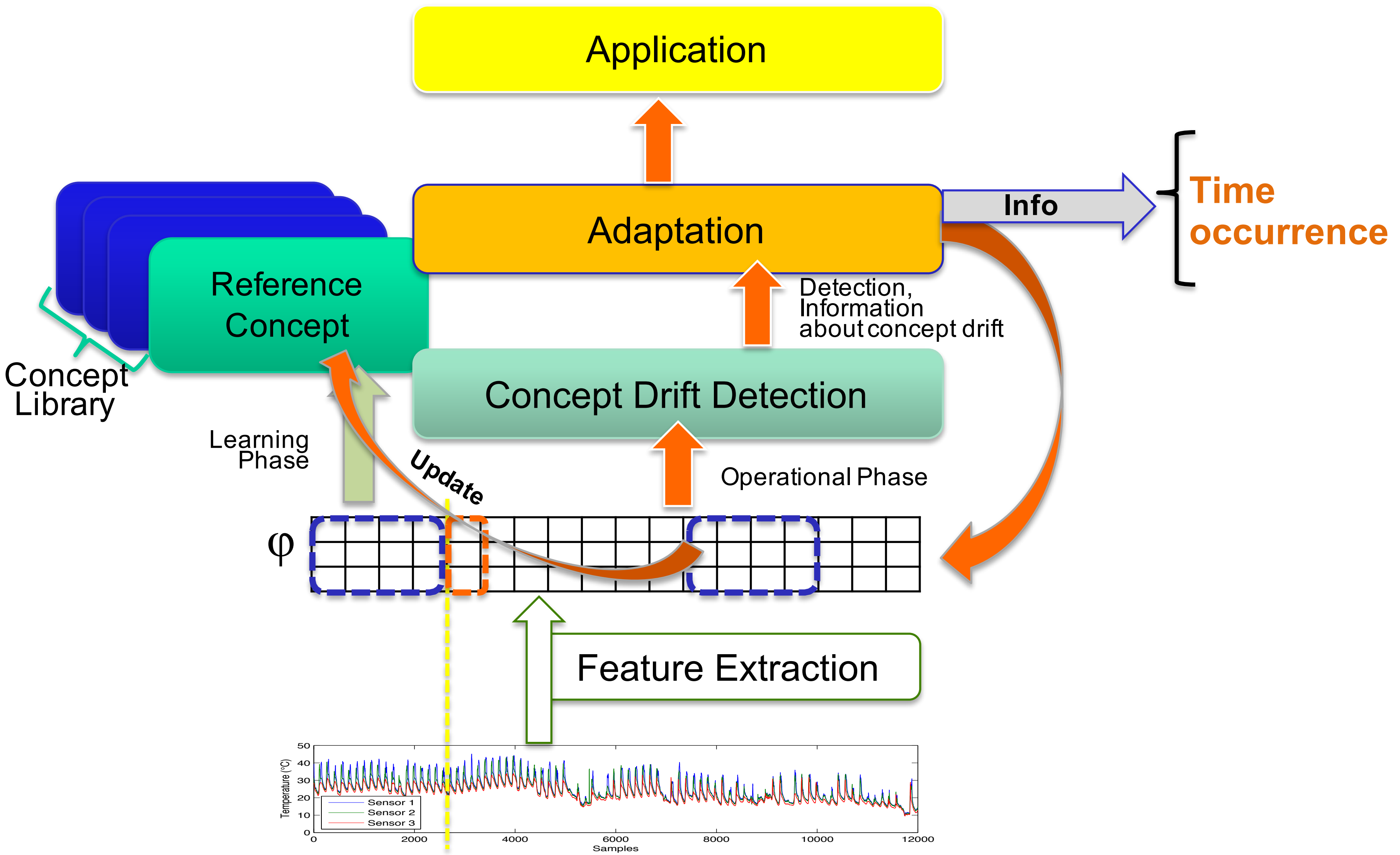
Change detection on the classification error



↑: reacting to changes when these directly influence its accuracy

↓: the need of supervised samples

The active learning framework within an evolving environment

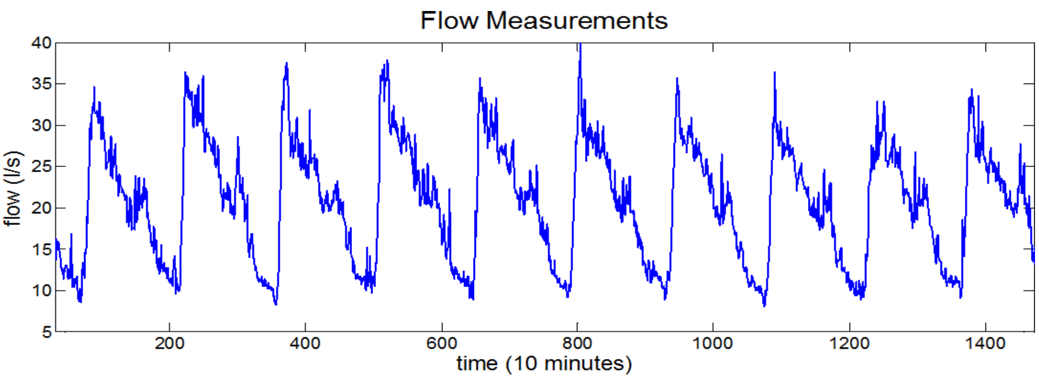


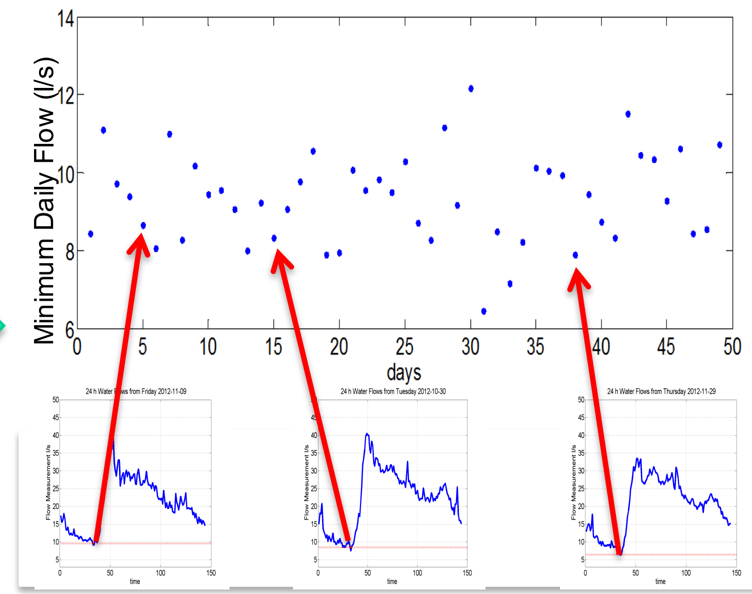
Features (1)

Features must be i.i.d (but data are generally signals)

- **Data space**
  - ✓ Raw data are used (e.g., the minimum of the water consumption of a day @ district metered area)

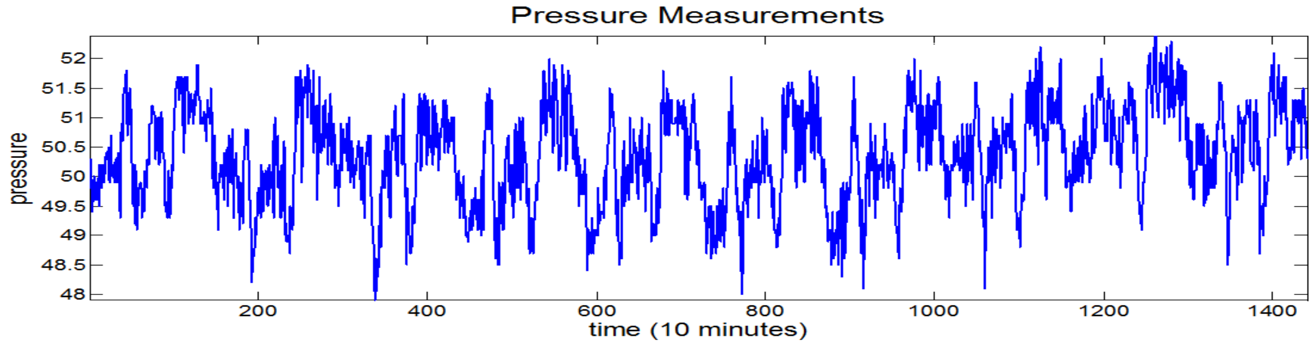
Water flow at the DMA

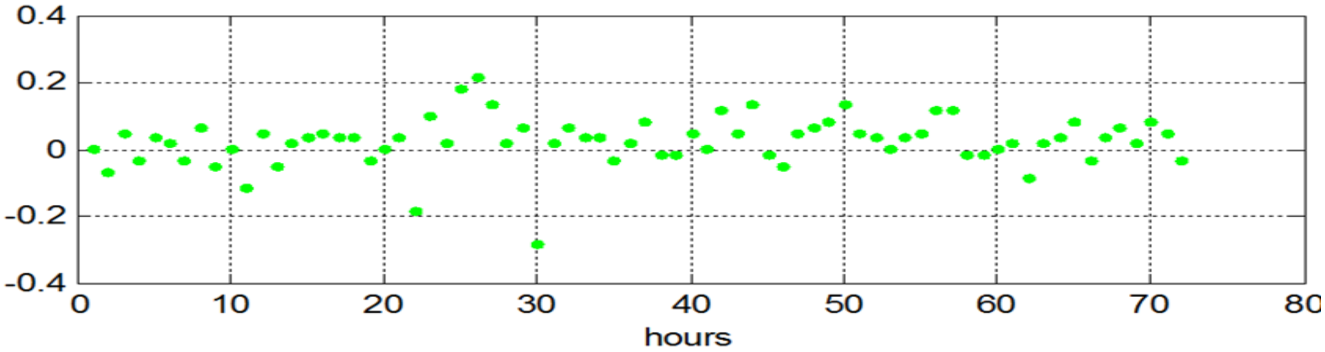




Features (2)

- **Feature space**
  - ✓ Any i.i.d. feature (e.g., residual, measurements in a quality analysis applications)

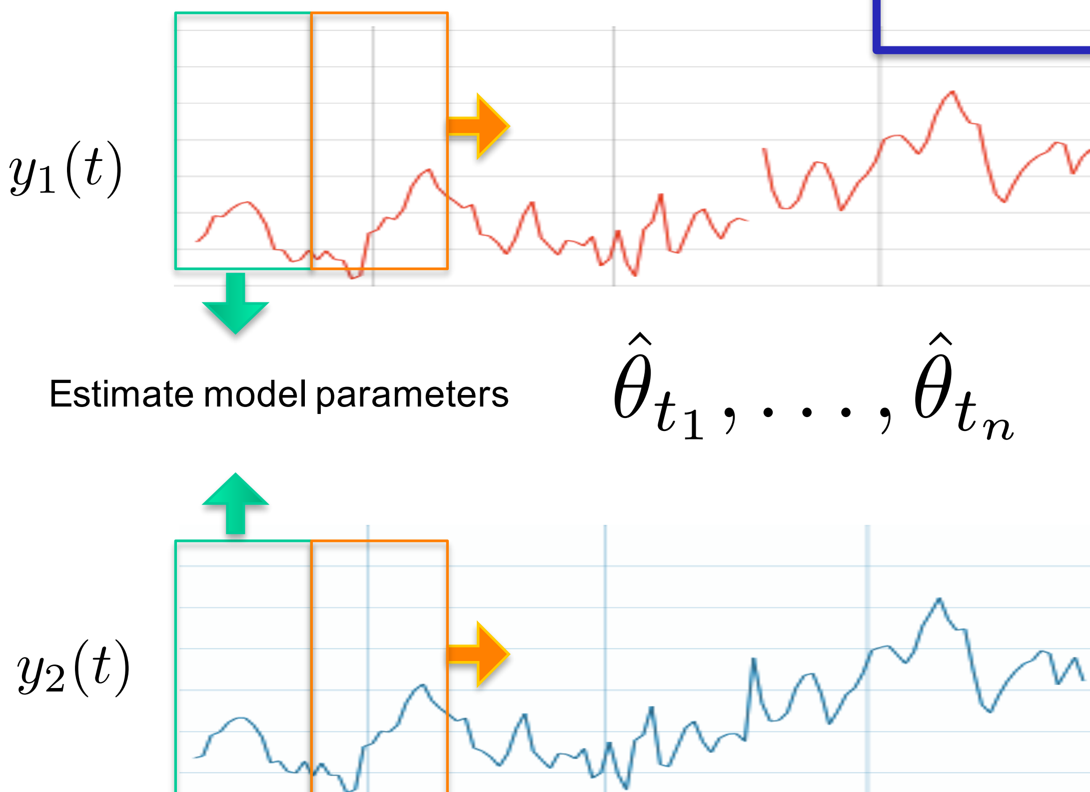




Features (3)

- **Model space**
  - ✓ LTI models are used to approximate the signal

$$y_1(t) \rightarrow \boxed{f_\theta^{1,2}} \longrightarrow y_2(t)$$



Active learning

Phase	How
Concept Drift Detection	Change detection tests CDT CPM Change point methods
Identification of the new state	Determination of consistent data instances
Retrain the application	Online, batch or full learning

Concept drift detection

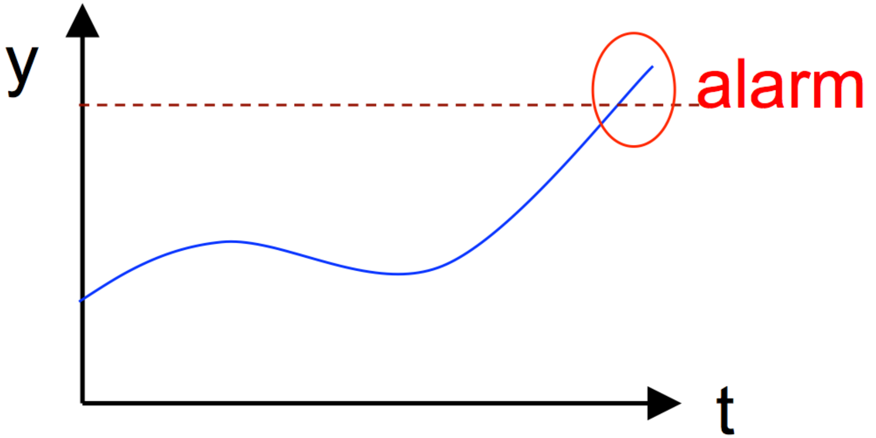
Ad hoc triggers designed to detect changes by inspecting sequences of data or derived features

- **Data-based methods**
  - Limit checking
  - Binary threshold
- **Statistical-based methods**
  - Statistical Hypothesis tests
  - Change-Point Methods
  - Change detection tests

Limit checking

- Testing if a given (measured) variable exceeds (indicating a change) or not a known absolute limit.

$$y(t) \le Y_{\lim} \to F(t) = 0$$
$$y(t) > Y_{\lim} \to F(t) = 1$$



- Variants:
  - Two limits, associated to different levels of safety.
  - Use of superior and inferior limits.
- Easy to implement.
- Too conservative (low change sensitivity).

Change detection with binary thresholds

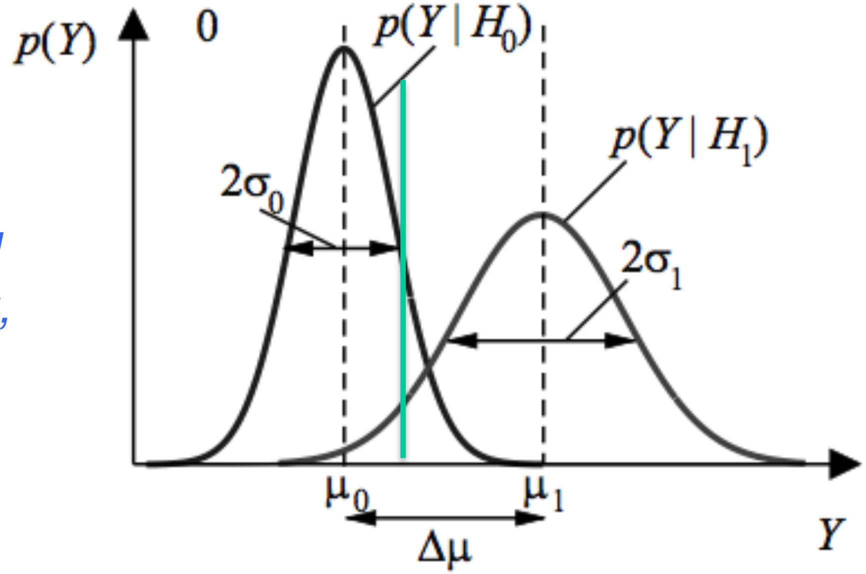
- Estimation of mean and variance
  - The monitored variables are usually stochastic variables $Y_i(t)$ with a certain pdf in nominal condition

$$\mu_i = E\{Y_i(t)\};\;\; \bar{\sigma}_i^2 = E\left\{[Y_i(t) - \mu_i]^2\right\}$$

- Changes are then expressed by

$$\Delta Y_i = E\{Y_i(t) - \mu_i\} \text{ and } \Delta\sigma^2 = E\left\{[\sigma_i(t) - \bar{\sigma}_i]^2\right\}$$

If the pdfs do not significantly overlap, one could use a fixed threshold based on σ, e.g., γ=2σ



Ratio between the detection of small changes and false alarms

More powerful techniques need to be considered

Statistical tests

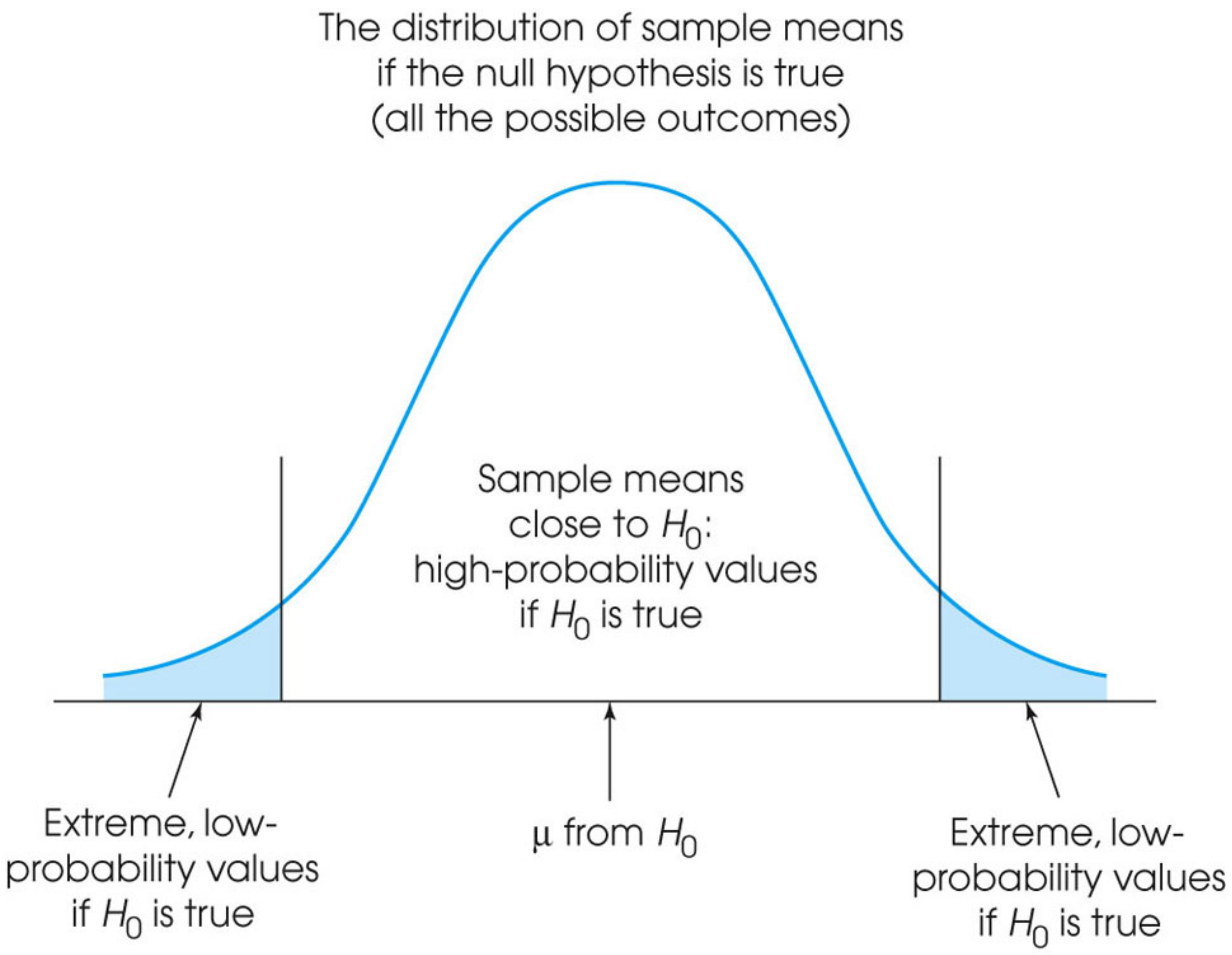
- off-line: fixed length sequence (after storing all data)
- on-line: at each time instant

- **Statistical hypothesis tests:**
  - Off-line
  - Control of FPs
- **Change detection tests**
  - On-line
  - No control of FPs

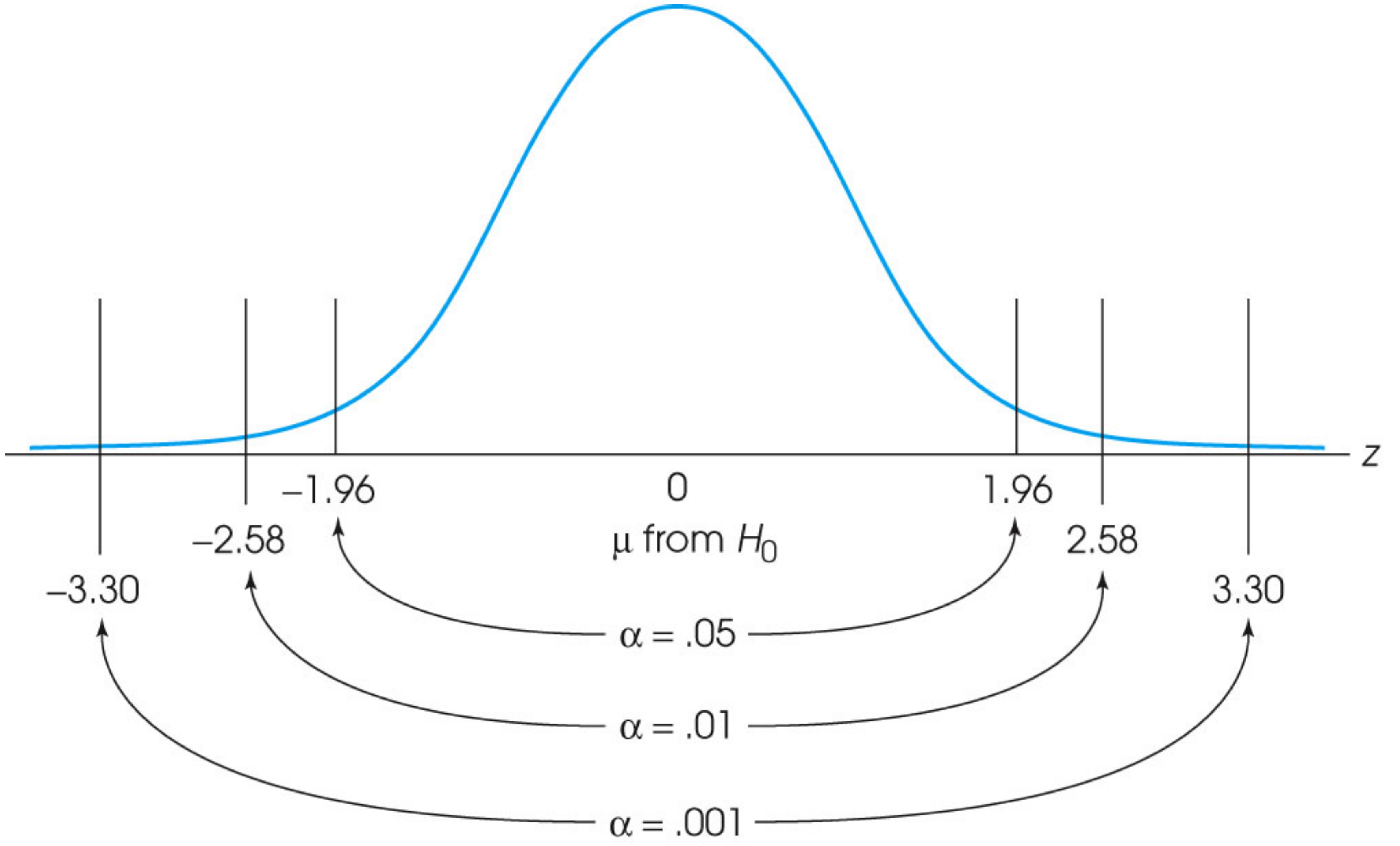
Statistical hypothesis tests

- Theory of statistics
- Testing one hypothesis (H_0) against one or more alternative hypotheses H_1 ,.., H_N
  - H_0: null hypothesis (no change) ➔ Y in Y_0
  - H_1 ,.., H_N: change hypothesis ➔ Y in Y_1
- *Decision*: Based on the assumption that the null hypothesis is true if no fault occurs, the null hypothesis is rejected and the alternate hypothesis is accepted if the sample of the random variable Y falls outside the region of acceptance. Otherwise, H_0 is accepted and H_1 rejected

Regions of rejection and acceptance for a HT



How to set the regions?



Hypothesis tests: the literature

	Test family	*Type (P/NP)*	*Change (Ab/Dr)*	*Entity under test*	*1D/ ND*	*On-line/ Off-line*	*Training Set /A priori information*	*Notes*
Z-test	Statistical Hypothesis testing	Parameteric	Abrupt	Mean	1D	Off-line	Parameters	Assume normality and known variance
t-test	Statistical Hypothesis testing	Parameteric	Abrupt	Mean	1D	Off-line	None	Assume normality
Mann-Whitney U test	Statistical Hypothesis testing	Non Parameteric	Abrupt	Median	1D	Off-line	None	Rank Test
Kolmogorov-Smirnov test	Statistical Hypothesis testing	Non Parameteric	Abrupt	Pdf	1D	Off-line	None	Also goodness of fit test
Kruskal-Wallis test	Statistical Hypothesis testing	Non Parameteric	Abrupt	Median	1D	Off-line	None	Mann-Whitney based, Multiple subsets

Change point methods

CPMs inspect a sequence of data and check for concept drift

Given sequence

$$\mathscr{X} = \{x(t), t = 1, \ldots, n\}$$

Produce a generic partitioning

$$\mathscr{A}_\tau = \{x(t),\ t = 1, \ldots, \tau\},$$
$$\mathscr{B}_\tau = \{x(t),\ t = \tau + 1, \ldots, n\}$$

and

$$\tau \text{ is a change point if } x(t) \sim \begin{cases} \mathscr{F}_0, & \text{for } t < \tau \\ \mathscr{F}_1, & \text{for } t \geq \tau \end{cases}$$

In practice

$$\begin{cases} \text{The estimated change-point in } \mathscr{X} \text{ is } M & \text{if } \mathscr{T}_M \geq h_{n,\alpha} \\ \text{No change-point identified in } \mathscr{X}, & \text{if } \mathscr{T}_M < h_{n,\alpha} \end{cases}$$

Change point methods

Example

$$x(t) \sim \begin{cases} \mathcal{N}(0,1), & \text{if } t < 350 \\ \mathcal{N}(-1,1), & \text{if } t \geq 350 \end{cases}$$

With hypothesis test

$$H_0 : \forall t,\ x(t) \sim \mathscr{F}_0$$

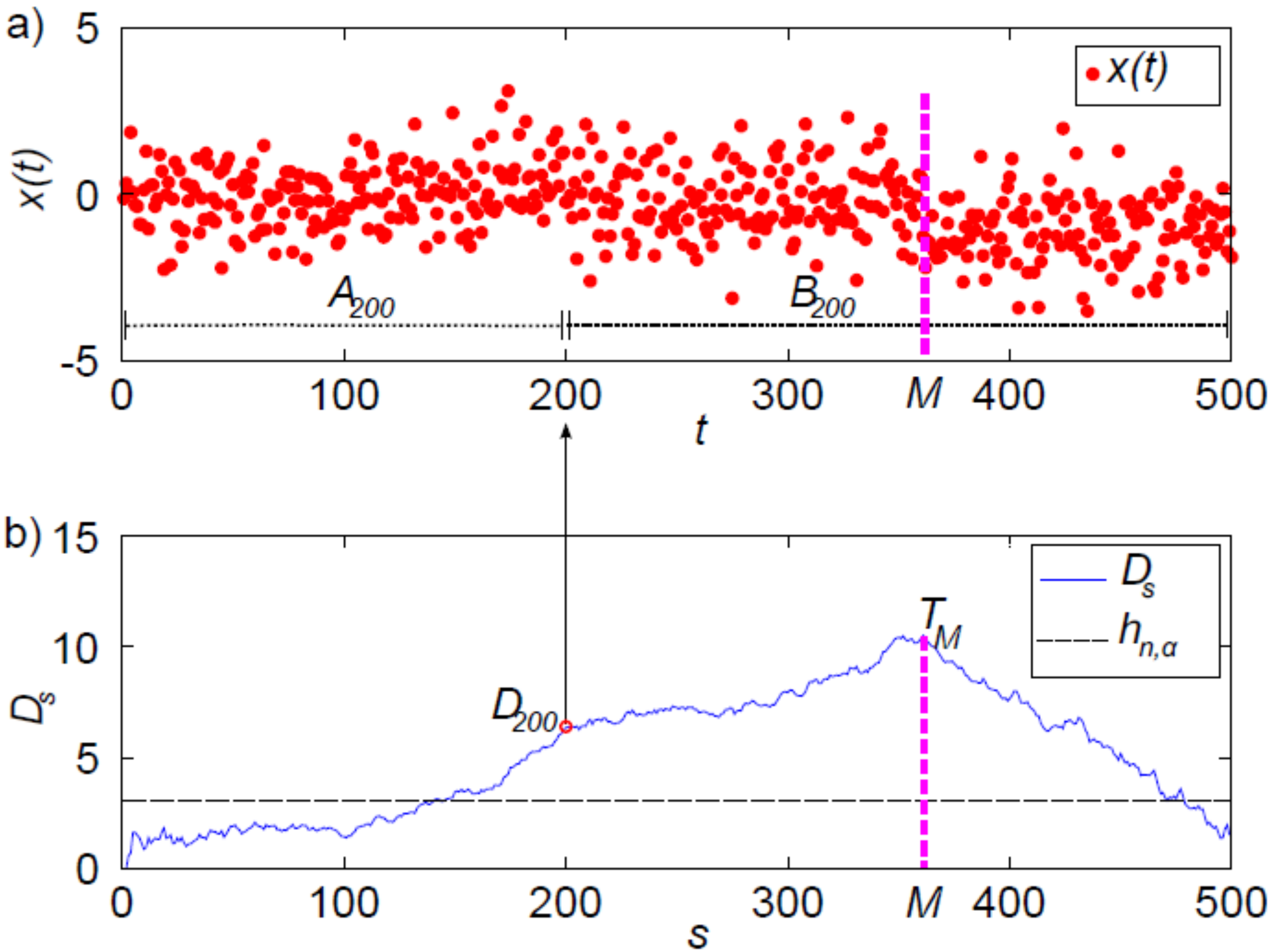
$$H_1 : \exists\, \tau\ x(t) \sim \begin{cases} \mathscr{F}_0, & \text{if } t < \tau \\ \mathscr{F}_1, & \text{if } t \geq \tau \end{cases}$$

For instance consider the Student t statistics for the means

$$D_\tau = \sqrt{\frac{\tau(n-\tau)}{n}} \frac{\bar{\mathscr{A}}_\tau - \bar{\mathscr{B}}_\tau}{S_\tau}$$

Change point methods

Threshold e.g., $h_{500,0.05} = 3.225$ provided by the CPM package



Change-detection tests

- *Change detection tests are methods designed to detect variations in the pdf of the process generating the data*

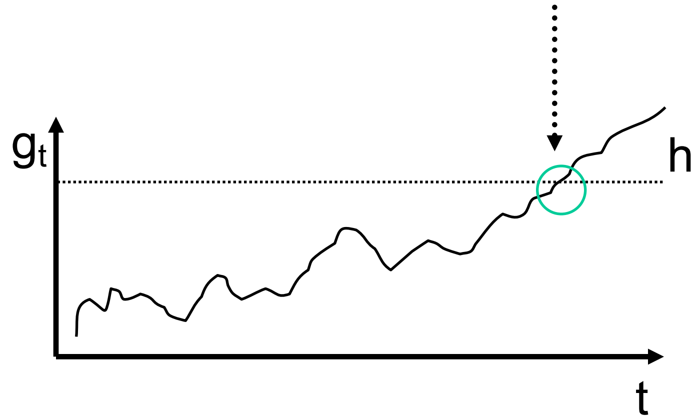
- **Parametric approach**: knowledge of the pdf before and after the change
  - CUSUM test
  - Shiryaev-Robert test
- **Nonparametric approach:**
  - CI-CUSUM test, NPCUSUM test
  - ICI-based change detection test
- **Semi-parametric approach:**
  - Semiparametric log-likelihood criterion (SPLL)

The CUSUM test

- $X = \{x_1, x_2, .., x_N\}: \; p_\theta(x)$
- The change at t_0 modeled as a transition from θ_0 to θ_1 (Hp: we keep the pdf structure)
- Measure a discrepancy at time time t: $s_t = \ln \frac{p_{\theta_1}(x_t)}{p_{\theta_0}(x_t)}$
- Evaluate the cumulative sum $S_t = \sum_{i=1}^{t} s_t$

Kulback-Leibler

- CUSUM identifies a change at time $\bar{t}$
  when $g_t = S_t - m_t \geq h \big|_{\bar{t}}$
  with $m_t = \min_{1 \leq i \leq t}(S_t)$



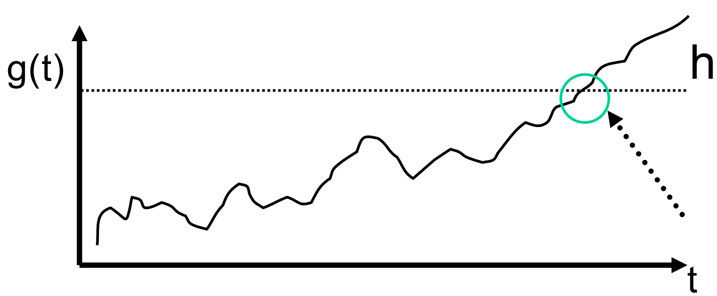
The CI-CUSUM test

Self-configuration procedure

1. Observations $X = \{x(t), t = 1,..,T\},\ x(t) \in \Re^d$
2. Partitions of X into disjoint intervals $Y(s) = \{x(t), (\nu - 1) \cdot s \le t < s \cdot \nu\},$
3. Extract the average feature vector $\varphi_y(s)$ (e.g., mean, var., kur., skew.) from each subsequence $Y(s)$
4. The pdf is gaussian from the central limit theorem
5. Estimate the null hypothesis Θ^0 from $TS = \{\varphi_y(s), s \le s_0\}$
6. Define m alternative hypotheses $\{\Theta^j\}, j = 1,..,m$ as "not being in Θ^0"

Running

7. Measure the discrepancy at time s as $R_j(s) = \sum_{\tau=1}^{s} \ln\left(\frac{N_{\Theta^j}(\varphi_y(\tau))}{N_{\Theta^0}(\varphi_y(\tau))}\right), j = 1,..,m$
8. CI-CUSUM identifies a change at time $\bar{s}$ if $g(\bar{s}) = R_j(\bar{s}) - \min_{1 \le \tau \le \bar{s}} R_j(\tau) > h_j$

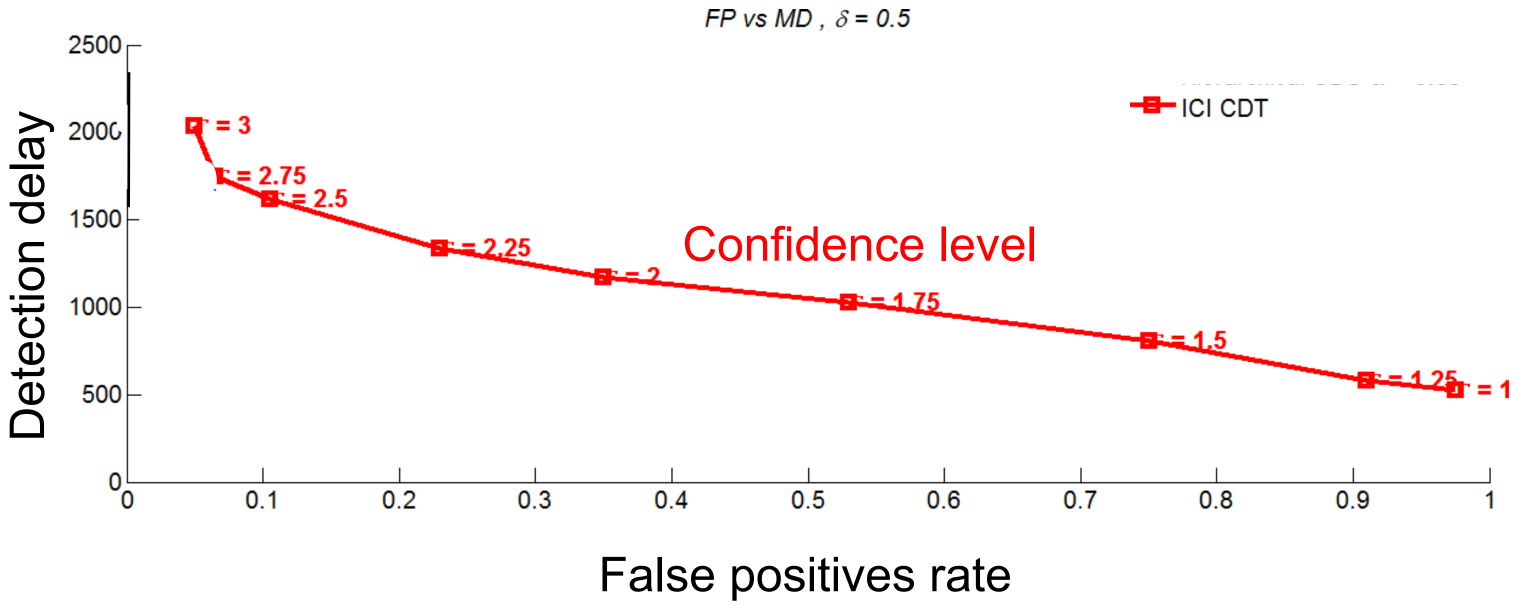


The ICI-based change detection test

- The test relies on a **set of functions that transform the observations into Gaussian distributed features**
- ICI rule: a method for developing **adaptive estimates for regression** of functions from noisy observations (signal and image denoising)

The **ICI rule**, combined with a polynomial regression technique, **assesses the stationary of the features** (and hence of the process)

Particularly effective in detecting changes but ....



How to increase promptness in detection still maintaining robustness w.r.t false positives?

The answer to the question "what happened?"
is not enough ...

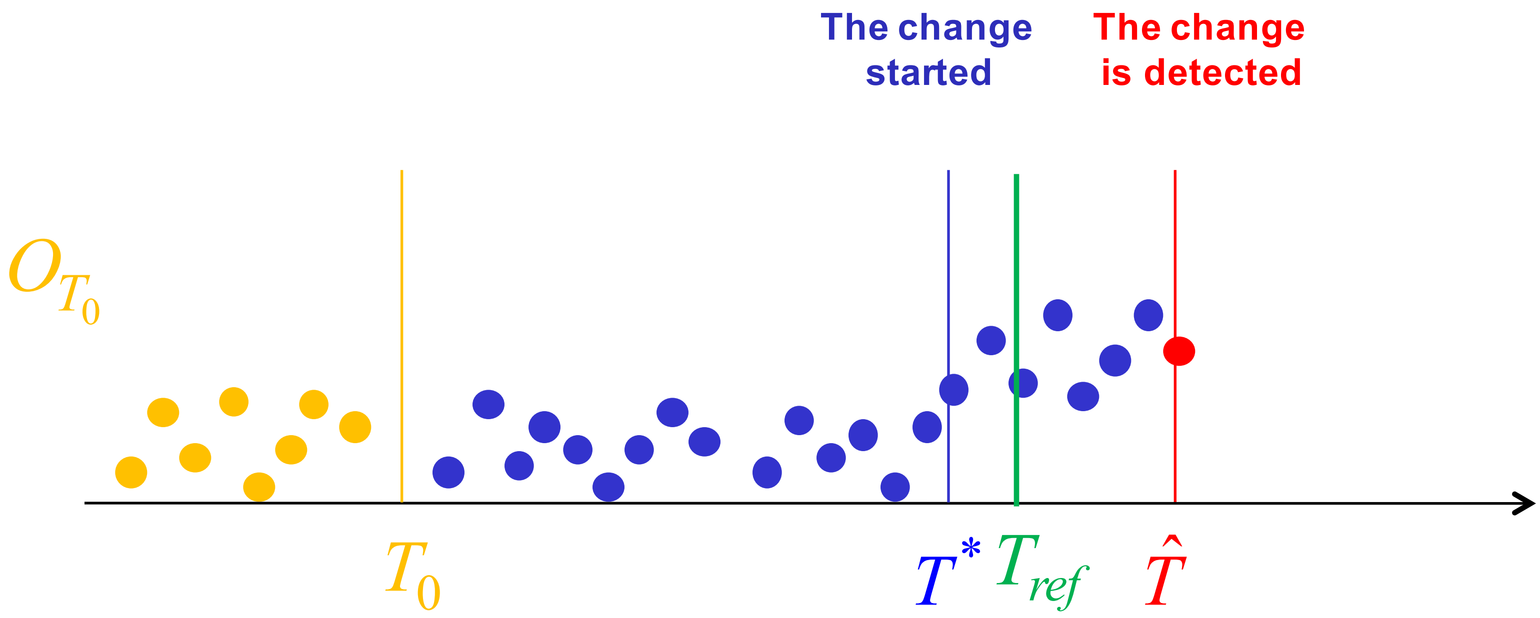
"Apparently you collapsed when told the price of these ..."

... Tell me: "when did it happen?"

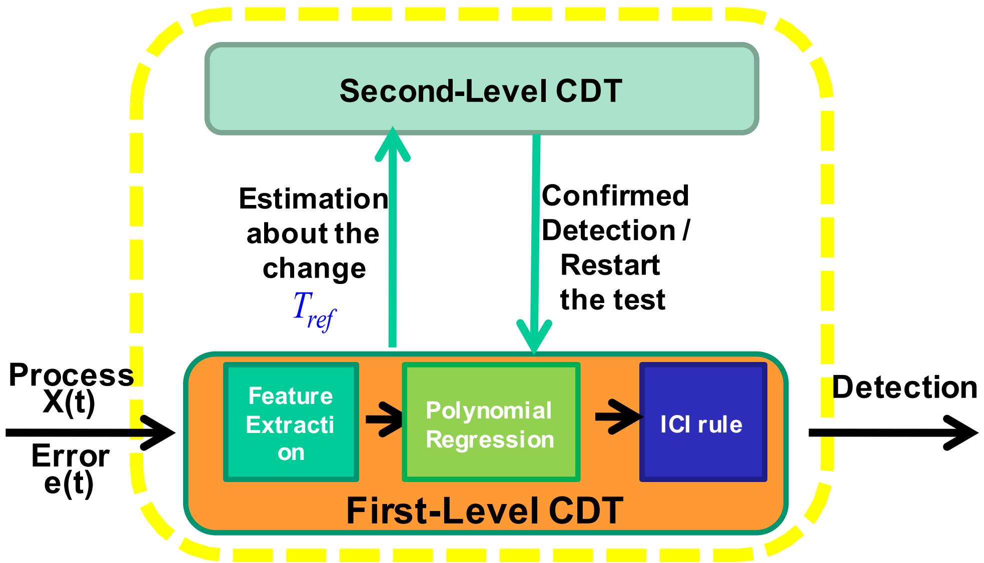
What and when ...

- Not only detection of the change, but also **estimation of the time instant** the process becomes non stationary

After the detection, we want an estimate T_{ref} of T^* by means of a refinement procedure



Hierarchical CDT



A **multivariate hypothesis** test based on the Hotelling T-square statistics

$$S = (\bar{F}^0 - \bar{F}^1)' \left(\left(\frac{1}{n_0} + \frac{1}{n_1} \right) \Sigma \right)^{-1} (\bar{F}^0 - \bar{F}^1),$$

$$\left(\frac{n_0+n_1-2}{n_0+n_1-N-1} \right) \mathcal{F}(N, n_0 + n_1 - N - 1),$$

Change-point methods: statistical tests able to assess whether a given data-sequence contains (or not) a change point

$$\mathscr{X} = \{x(t), t = 1, \dots, n\}, \quad \begin{cases} \mathscr{A}_\tau = \{x(t),\ t = 1, \dots, \tau\}, \\ \mathscr{B}_\tau = \{x(t),\ t = \tau + 1, \dots, n\}, \end{cases}$$

Compute $\mathscr{T}_\tau = \mathscr{T}(\mathscr{A}_\tau, \mathscr{B}_\tau),$

$$\mathscr{T}_M = \max_{s=1,\dots,n} (\mathscr{T}_\tau)$$

$$\begin{cases} \text{The estimated change-point in } \mathscr{X} \text{ is } M_{\mathscr{X}} & \text{if } \mathscr{T}_M \geq h_{n,\alpha} \\ \text{No change-point identified in } \mathscr{X}, & \text{if } \mathscr{T}_M < h_{n,\alpha} \end{cases}$$

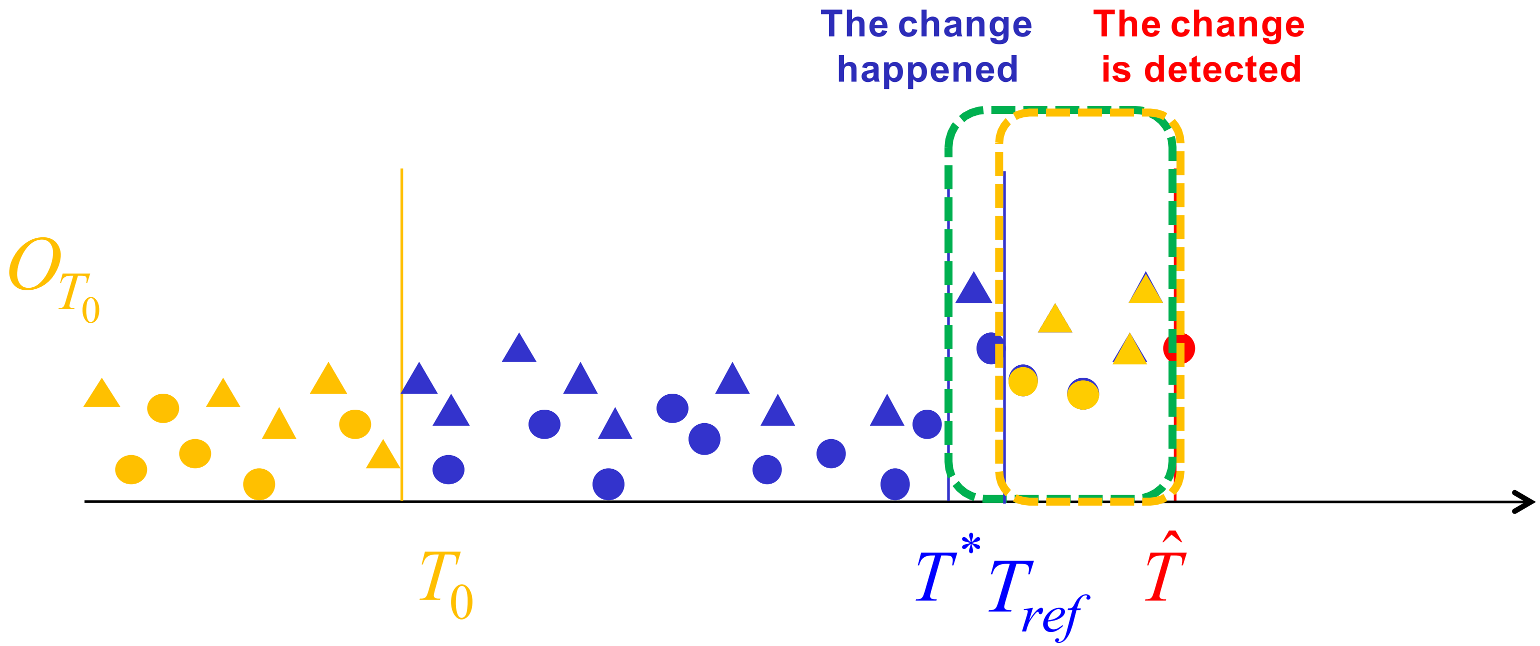
Second level change-detection test aiming at confirming (or not) the change hypothesis:

- Multivariate hypothesis test
- Change-point methods

Which data are consistent with the current status?

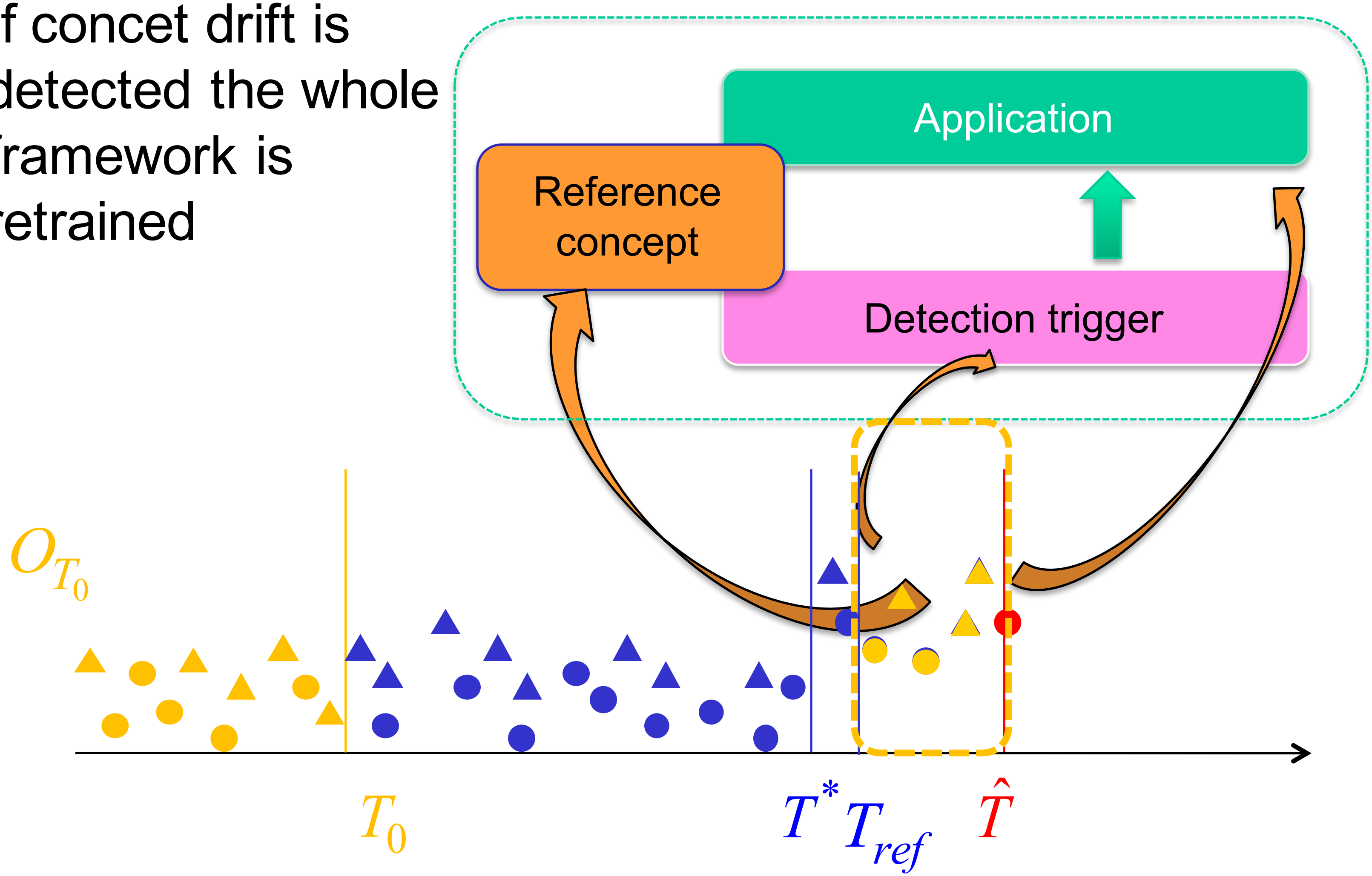
- Instances: between T^* and $\hat{T}$

T* is unknown: use estimates T_{ref} and $\hat{T}$



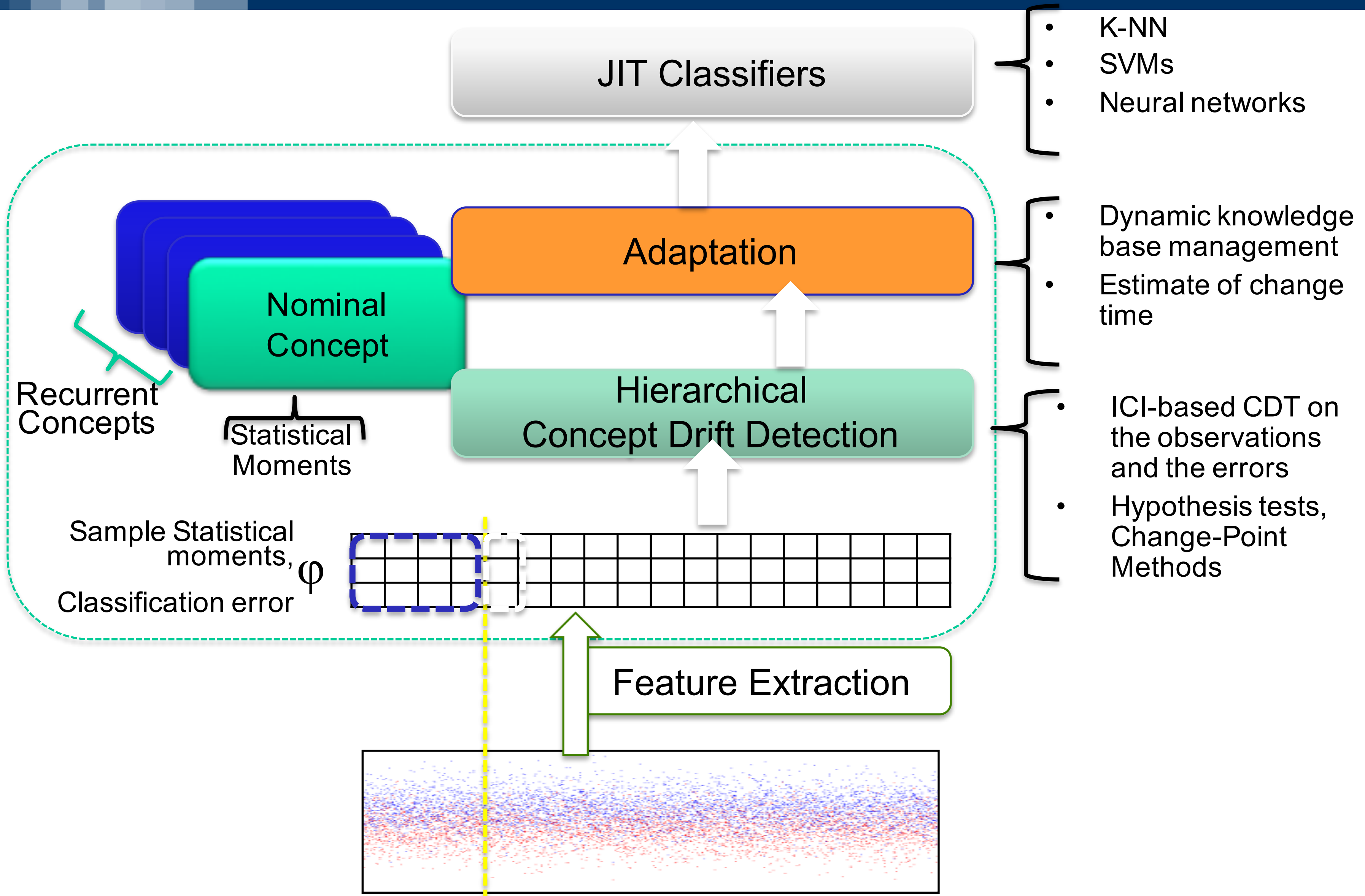
Retrain the application

- If concet drift is detected the whole framework is retrained



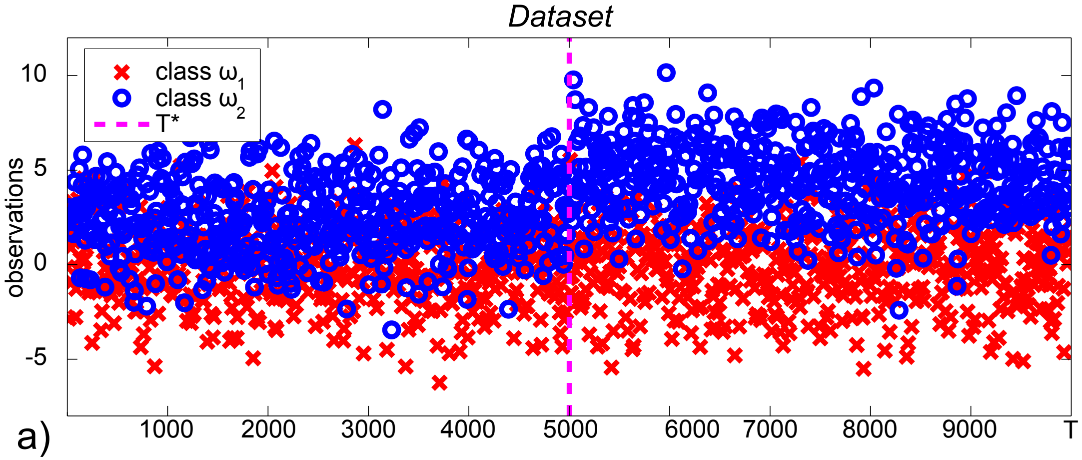
An example:
Just-in-Time Adaptive Classifiers

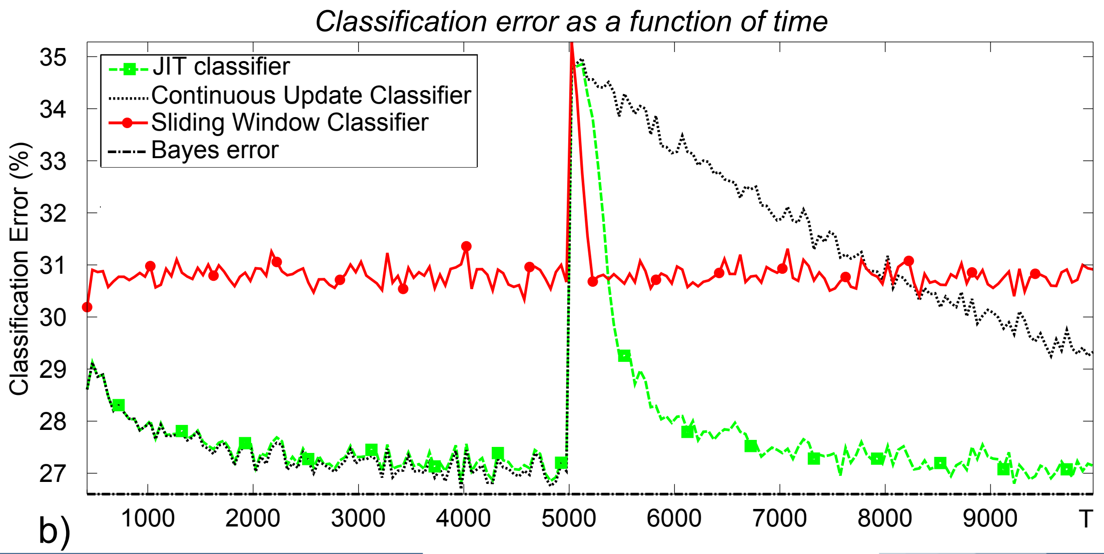
Just-in-Time Adaptive Classifiers



Asymptotic optimality with JIT classifiers

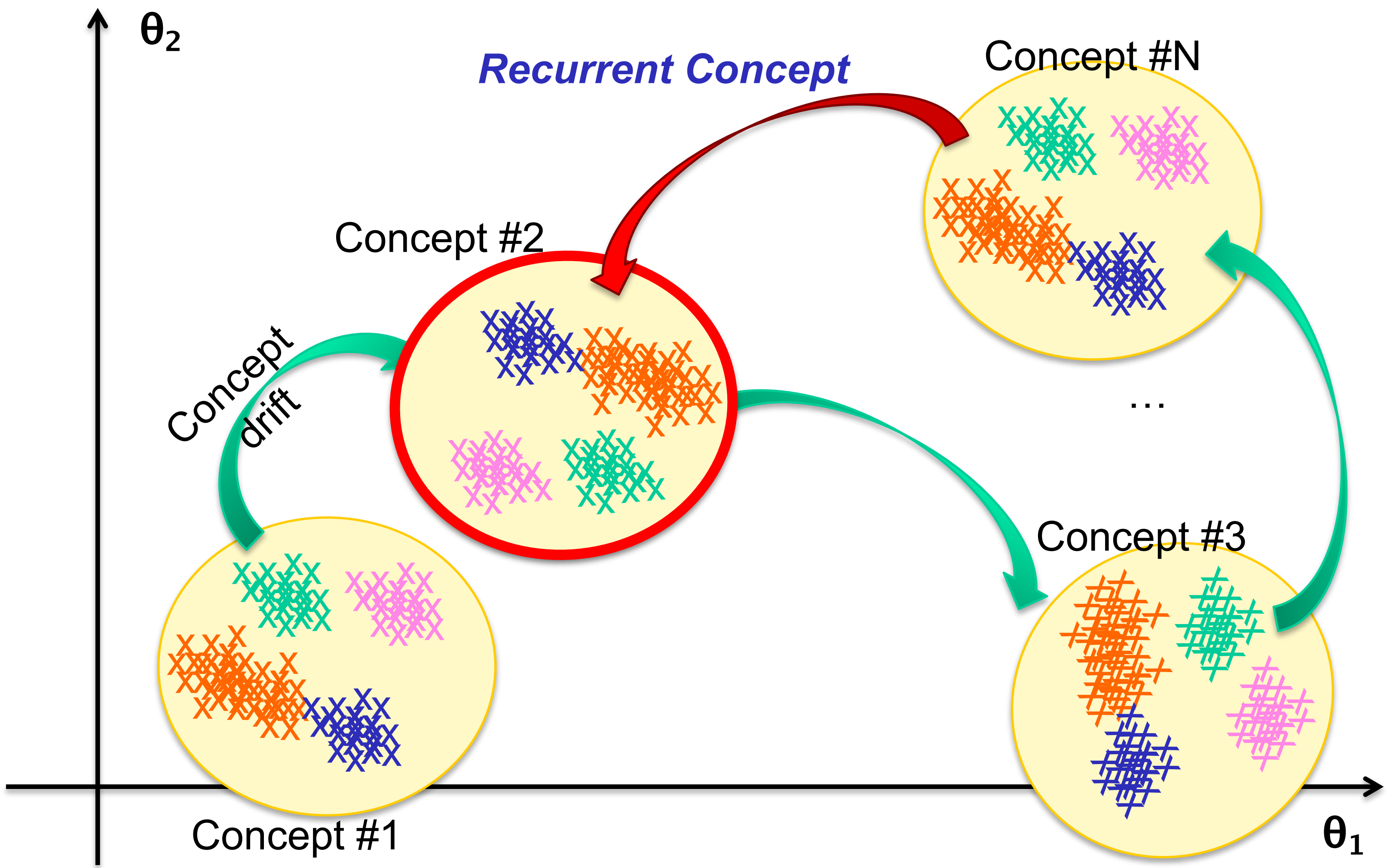
JIT adaptive classifiers grant asymptotic optimality when the process generating the data is affected by a sequence of abrupt concept drift





Gaussian classes

Dealing with concept drift …



$$p(x|t) = p(\omega_1|t)p(x|\omega_1, t) + p(\omega_2|t)p(x|\omega_2, t)$$

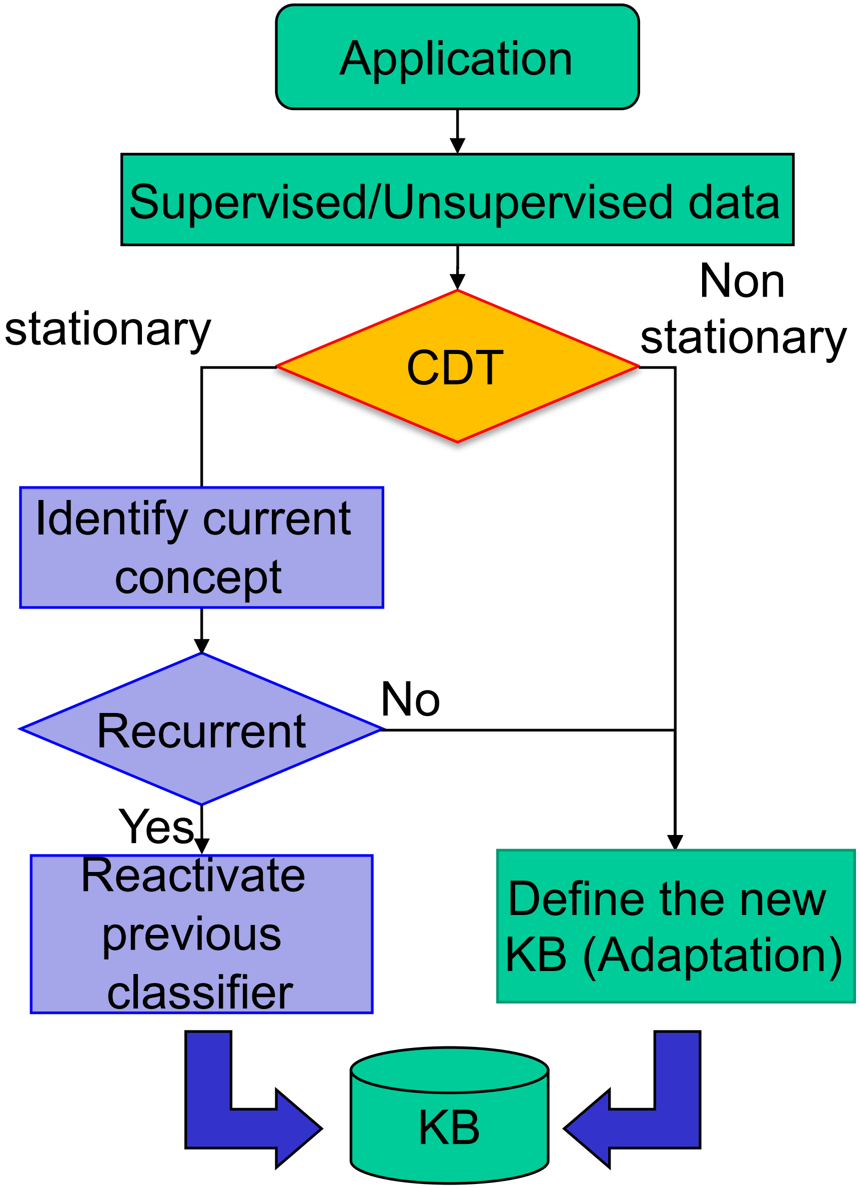
The novel idea: extending the JIT classifier

Two CDTs are to asses if:

- The **pdf** of the **input** is **stationary**
- the **classification error** is **stationary**

Adaptation phase consists in:

- **Isolation of the current concept**
- Identification of **recurrent concepts**
- Training the classifier by exploiting all the **available supervised information**



Some relevant remarks …

- ✓ Being acquainted with learning techniques is a plus in everybody's background
- ✓ Most of time the we can assume that the process generating the data is time invariant. When it is not we need to pay attention…
- ✓ Learning in a chaging environment must be considered and represents a key property intelligent systems should possess

Open Source Software and Available Benchmarks

- Many authors have made the **code and data** used in their publications available to the public
- **Code**:
  - Hierarchical ICI-based Change-Detection Tests
  - Learn++.NSE
  - (Scalable Advance) Massive Online Analysis
  - Online Nonstationary Boosting
- **Dataset (generator)**:
  - Minku & Yao’s / Kuncheva’s Concept Drift Generator
  - Kuncheva’s Concept Drift Generator
  - Airlines Flight Delay Prediction, Spam Classification, Chess.com
  - KDD Cup 1999: Collection of network intrusion detection data.
  - POLIMI Rock Collapse and Landslide Forecasting

Ditzler, G., Roveri, M., Alippi, C., & Polikar, R. (2015). Learning in nonstationary environments: a survey. *IEEE Computational Intelligence Magazine*, *10*(4), 12-25.

Topics of Future Interest

- Theoretical frameworks for learning
- Unstructured and heterogeneous data streams
- Transient concept drift and limited data
- Concept drift and Big Data

Ditzler, G., Roveri, M., Alippi, C., & Polikar, R. (2015). Learning in nonstationary environments: a survey. *IEEE Computational Intelligence Magazine*, *10*(4), 12-25.

Let’s play with MATLAB

- Download the examples related to Active and Passive solutions ….

Some References

- J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy, and A. Bouchachia, “A survey on concept drift adaptation,” ACM Computing Surveys, 2014.
- Ditzler, Gregory, Manuel Roveri, Cesare Alippi, and Robi Polikar. "Learning in nonstationary environments: A survey." *Computational Intelligence Magazine, IEEE* 10, no. 4 (2015): 12-25.
- Tsymbal, Alexey. "The problem of concept drift: definitions and related work." *Computer Science Department, Trinity College Dublin* 106 (2004).
- IEEE Transactions on Neural Networks and Learning Systems, “Special Issue on Learning in Nonstationary and Evolving Environments”, Jan. 2004