QuantTrees: Histograms for Monitoring Multivariate Data Streams

Giacomo Boracchi

https://boracchi.faculty.polimi.it/

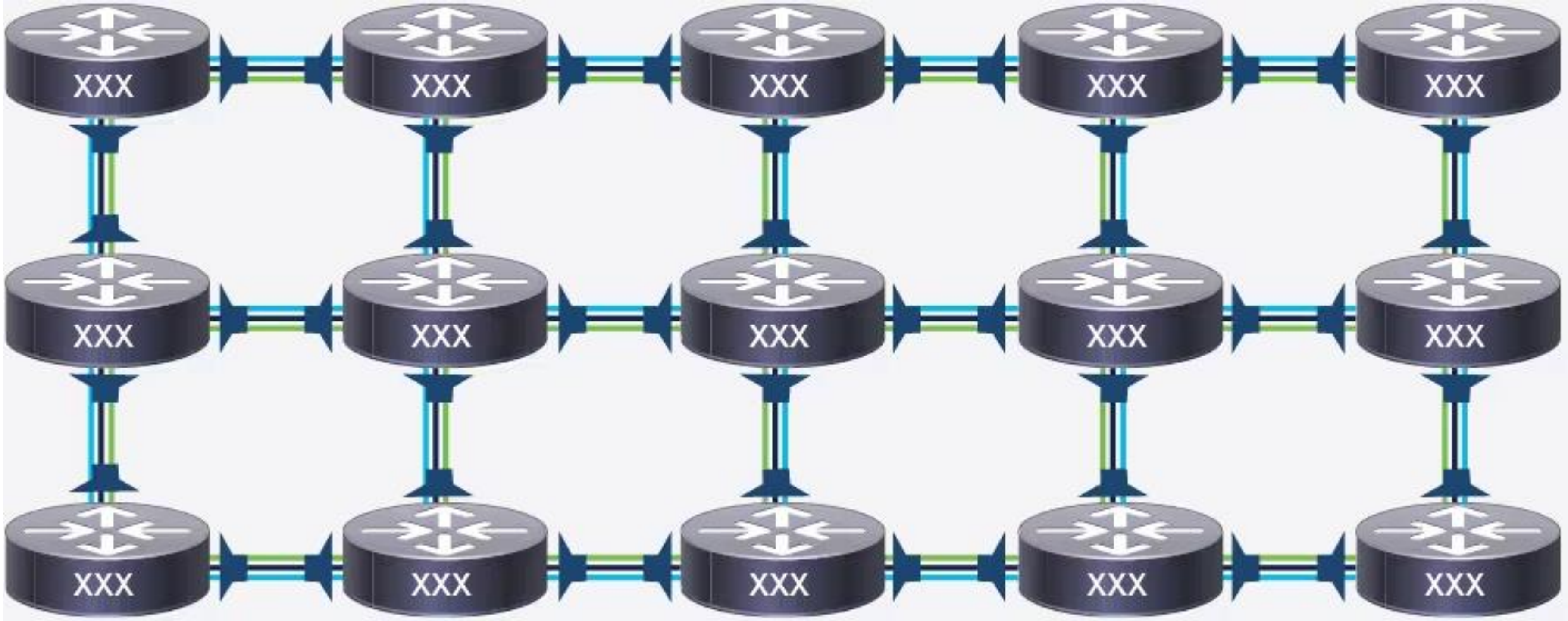
June 29th, 2025

International Symposium on Change Detection

EANN, Limassol Cyprus

POLITECNICO MILANO 1863

Use Case: Routed Optical Networks

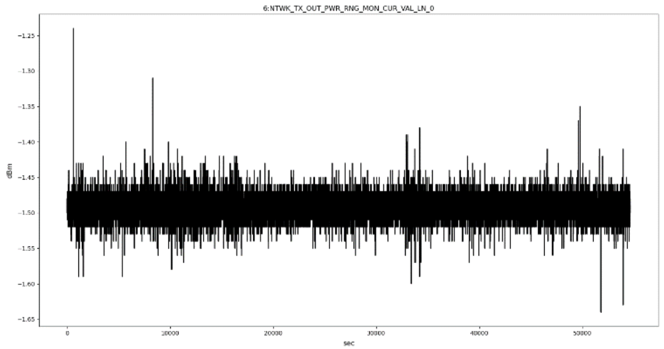


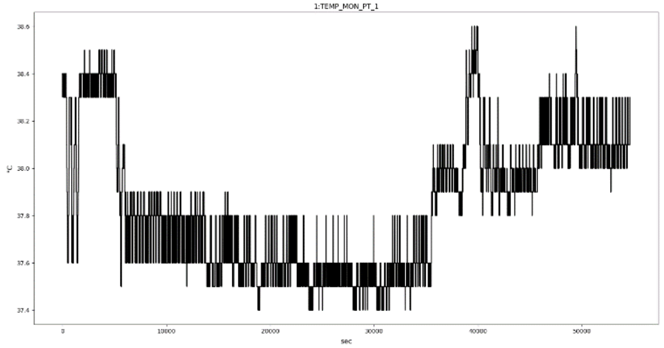
In collaboration with Cisco Photonics

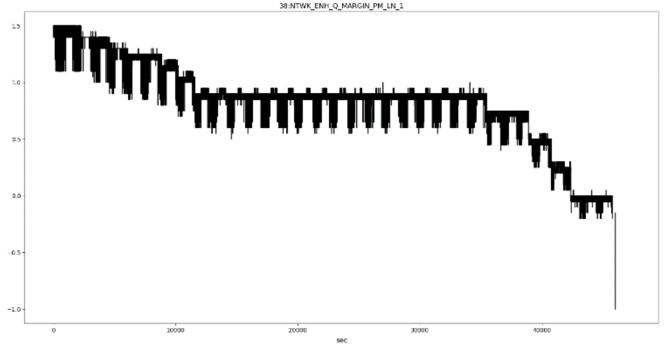
https://www.cisco.com/c/en/us/solutions/collateral/service-provider/routed-optical-networking/at-a-glance-c45-744217.html

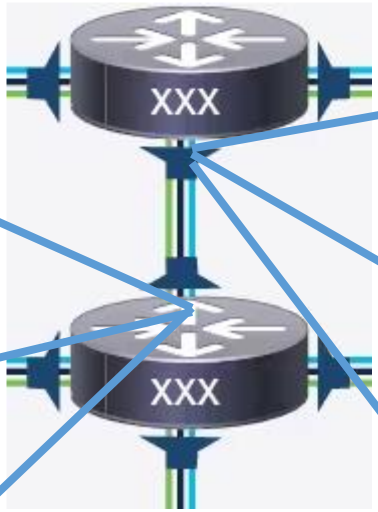
Routed Optical Networks

Datastreams @Router 1

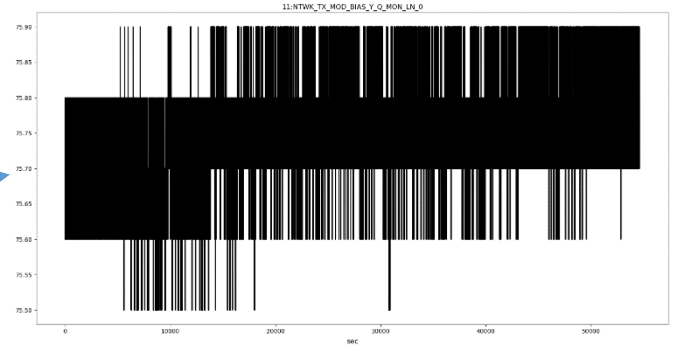


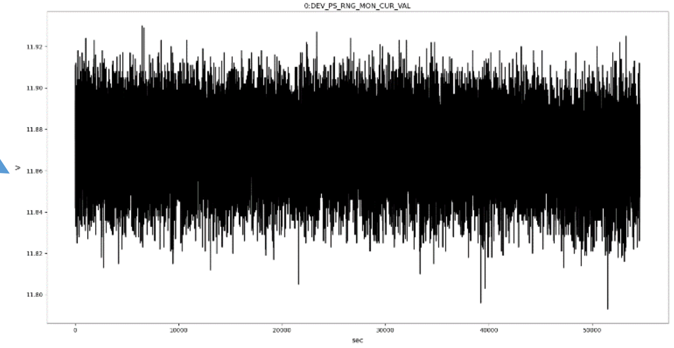


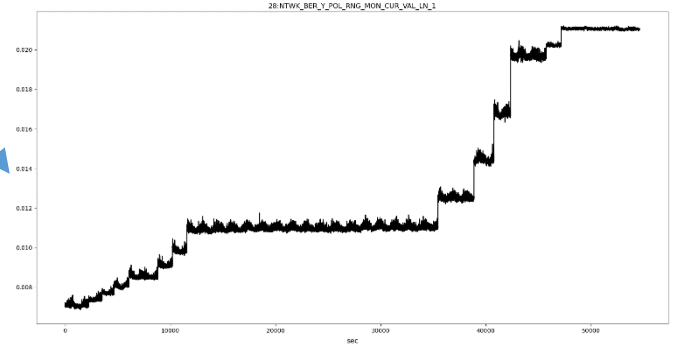




Datastreams @Router 2







In order to define the best routing strategies, each router needs to autonomously assess the transmission quality on each channel

https://www.cisco.com/c/en/us/solutions/collateral/service-provider/routed-optical-networking/at-a-glance-c45-744217.html

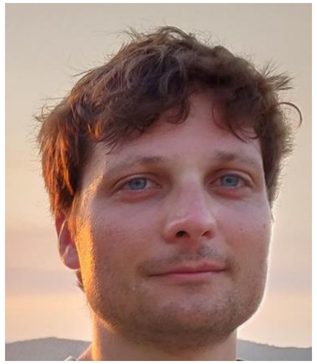
The “QuantTree Team”

Giacomo Boracchi

POLITECNICO MILANO 1863

Diego Carrera

POLITECNICO MILANO 1863 life.augmented



Luca Frittoli

POLITECNICO MILANO 1863

Diego Stucchi

POLITECNICO MILANO 1863 life.augmented

Olmo Notarianni

POLITECNICO MILANO 1863

Cristiano Cervellera

Consiglio Nazionale delle Ricerche

Danilo Macciò

Consiglio Nazionale delle Ricerche

Problem Formulation

Change Detection in Data Streams…

…and often also in time series… as the problem boils down to this, once having computed independent residuals

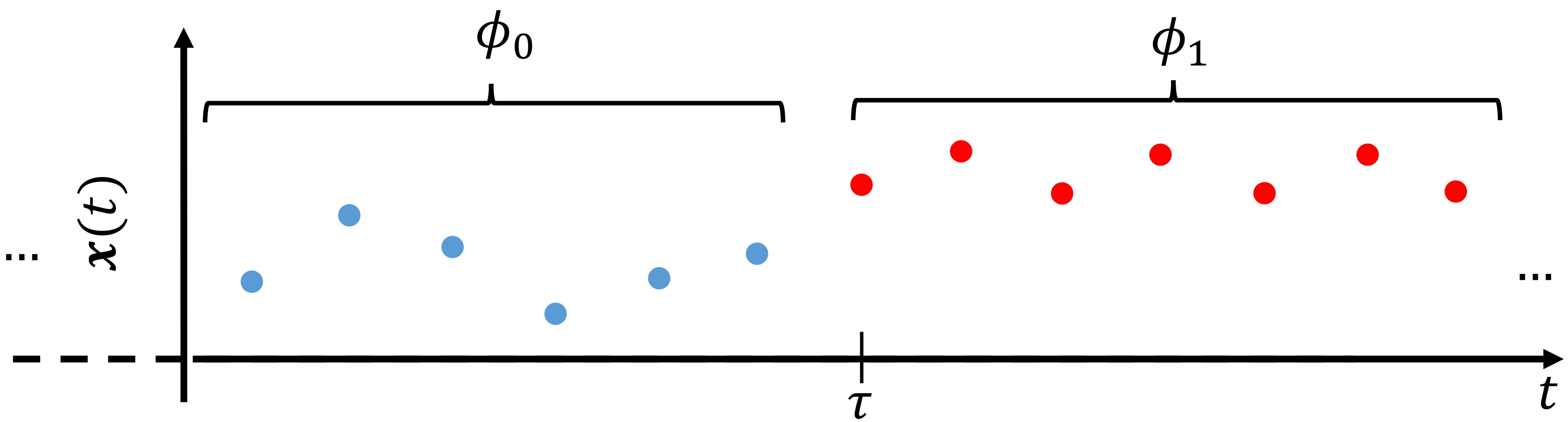
Change-Detection in a Statistical Framework

Monitor a **stream** $\{\boldsymbol{x}(t), t = 1, \dots\}$, $\boldsymbol{x}(t) \in \mathbb{R}^d$ of realizations of a **random variable**, and **detect the change-point** τ,

$$\boldsymbol{x}(t) \sim \begin{cases} \phi_0 & t < \tau & \text{in control state} \\ \phi_1 & t \geq \tau & \text{out of control state} \end{cases},$$

where $\{\boldsymbol{x}(t),\ t < \tau\}$ **are i.i.d.** and $\phi_0 \neq \phi_1$

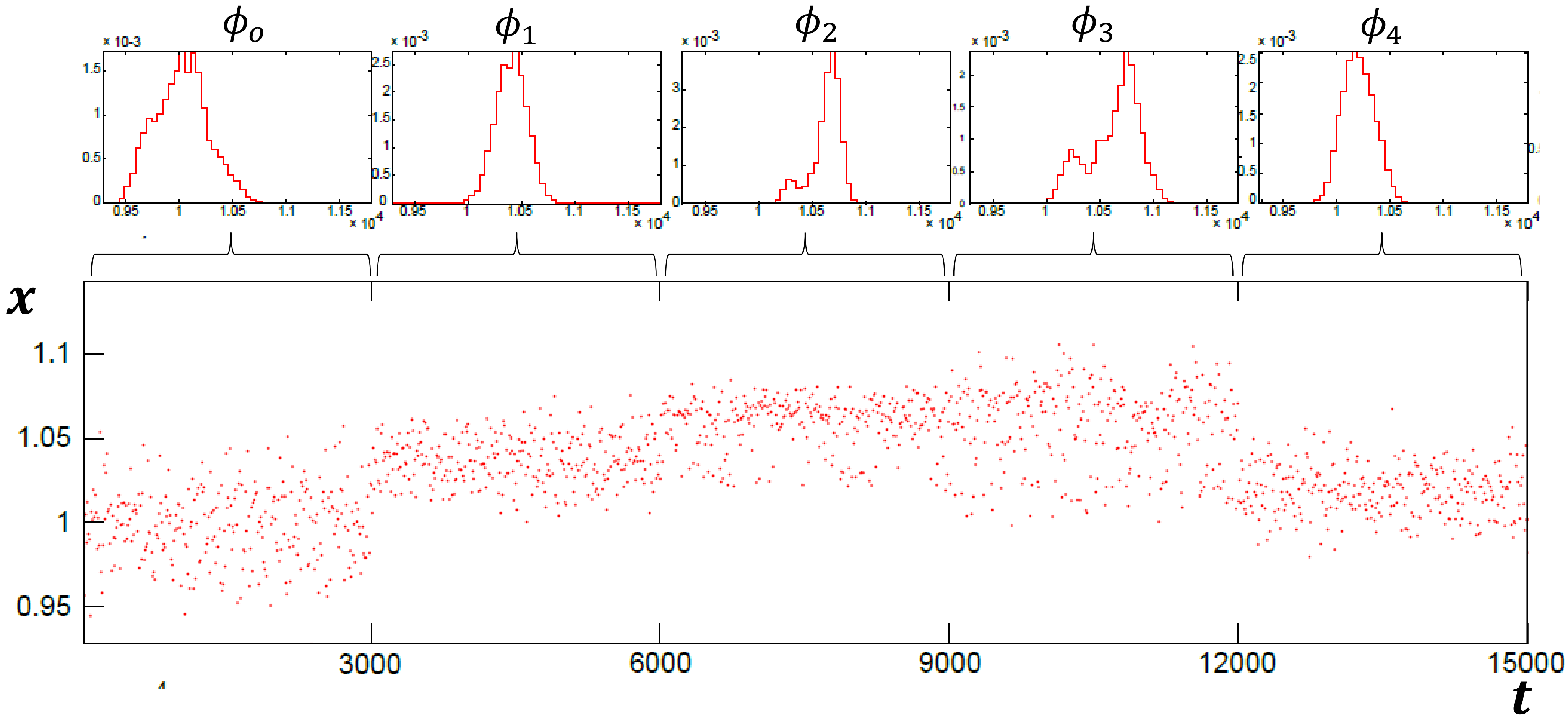
Typically, ϕ_1 is unknown and only $TR = \{\boldsymbol{x}(t) \sim \phi_0\}$ is given



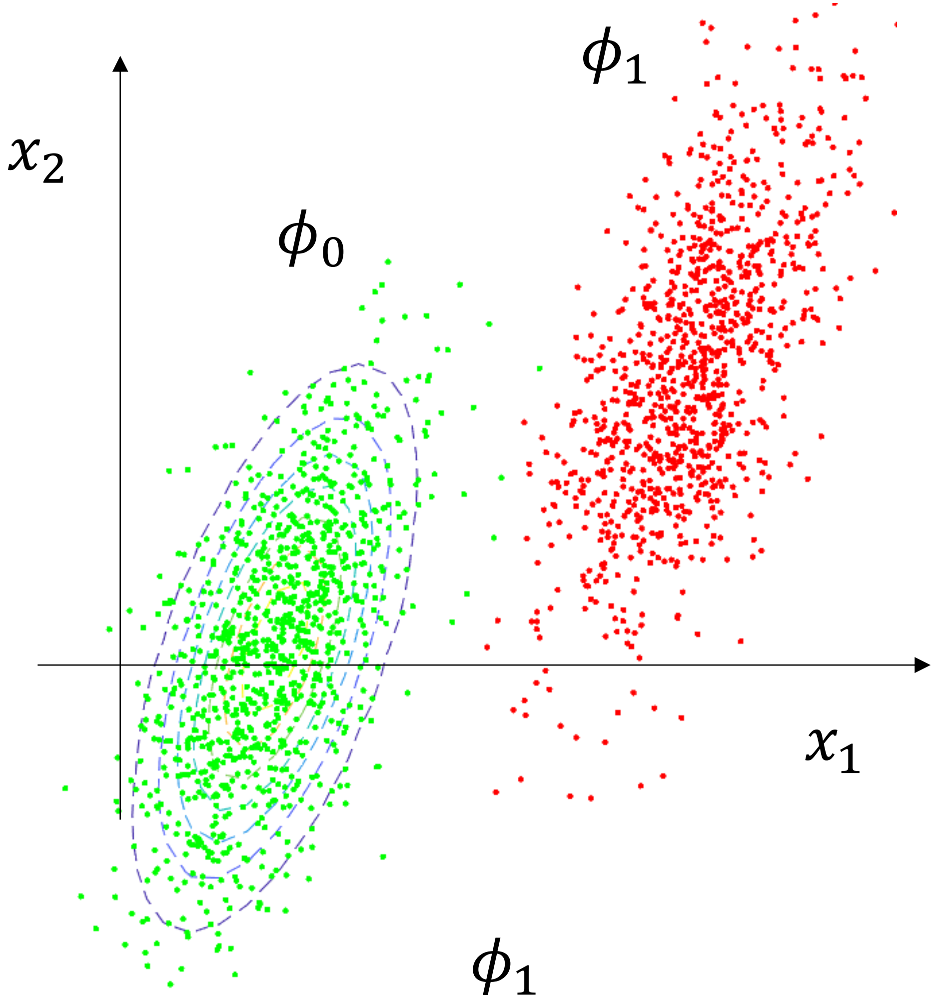
Change-Detection in a Statistical Framework

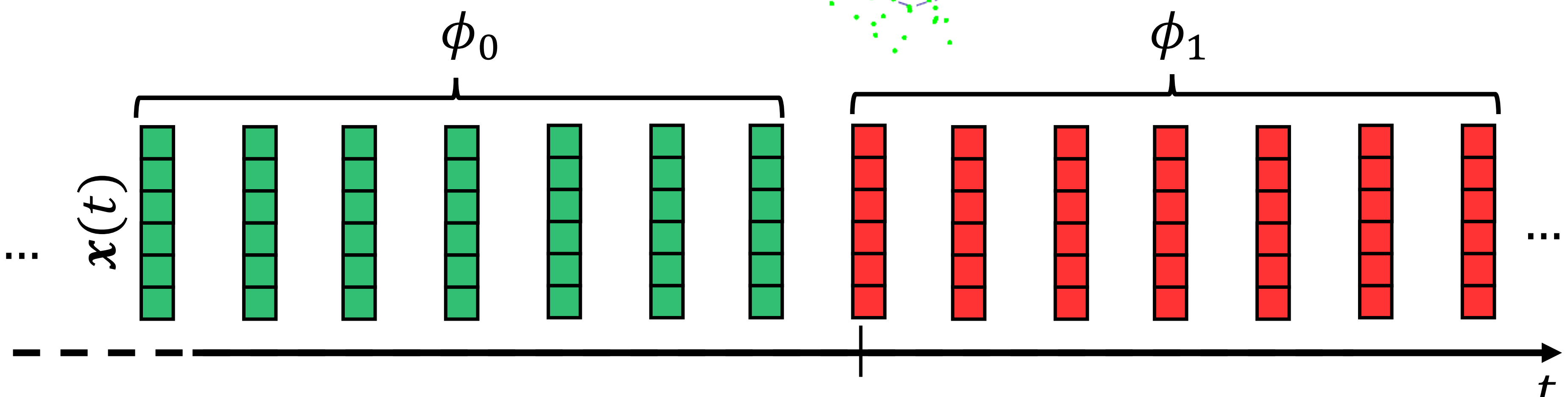
Here are data from an X-ray monitoring apparatus.

There are 4 changes $\phi_o \rightarrow \phi_1 \rightarrow \phi_2 \rightarrow \phi_3 \rightarrow \phi_4$ corresponding to different monitoring conditions and/or analyzed materials



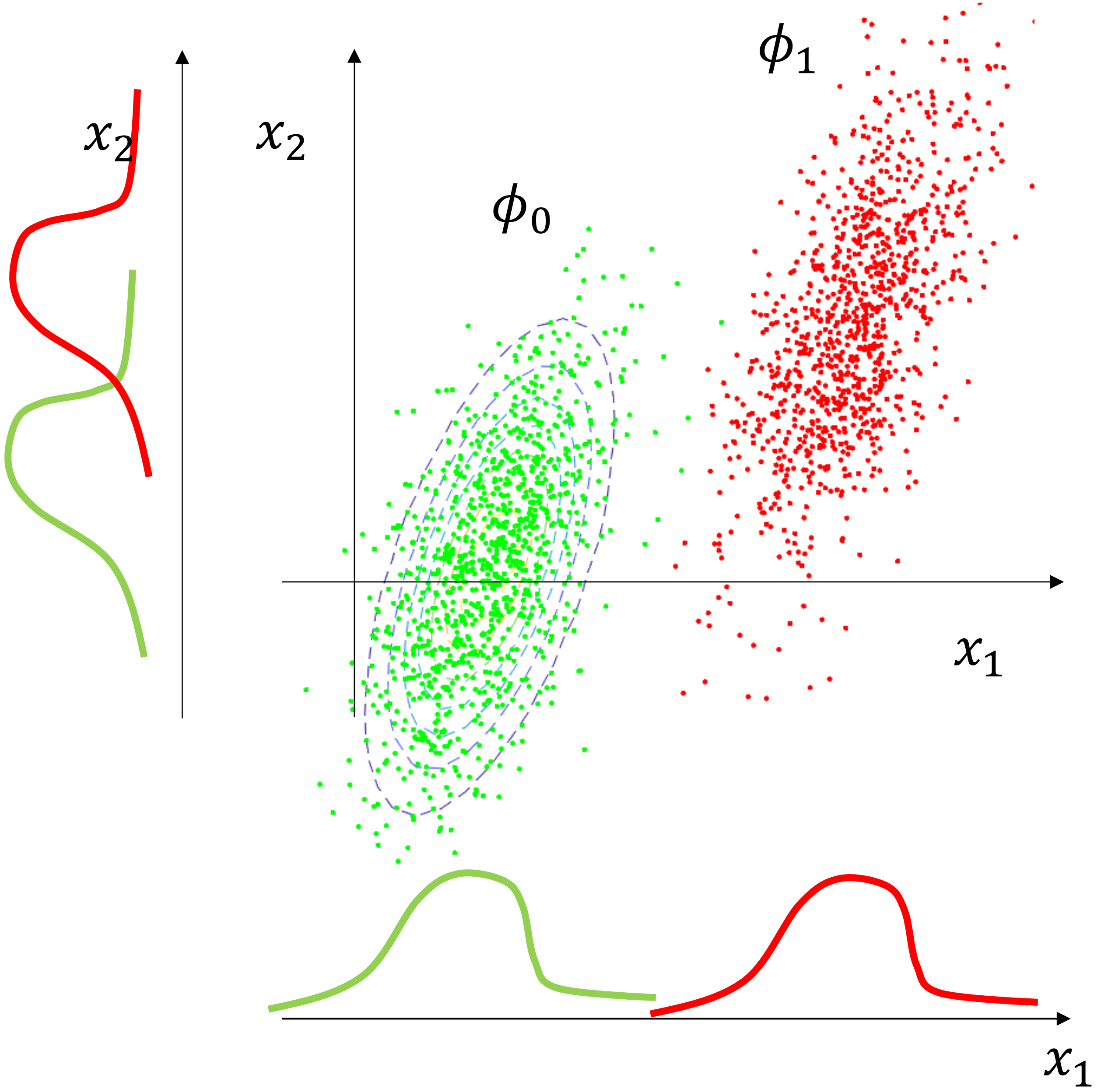
Multivariate Monitoring



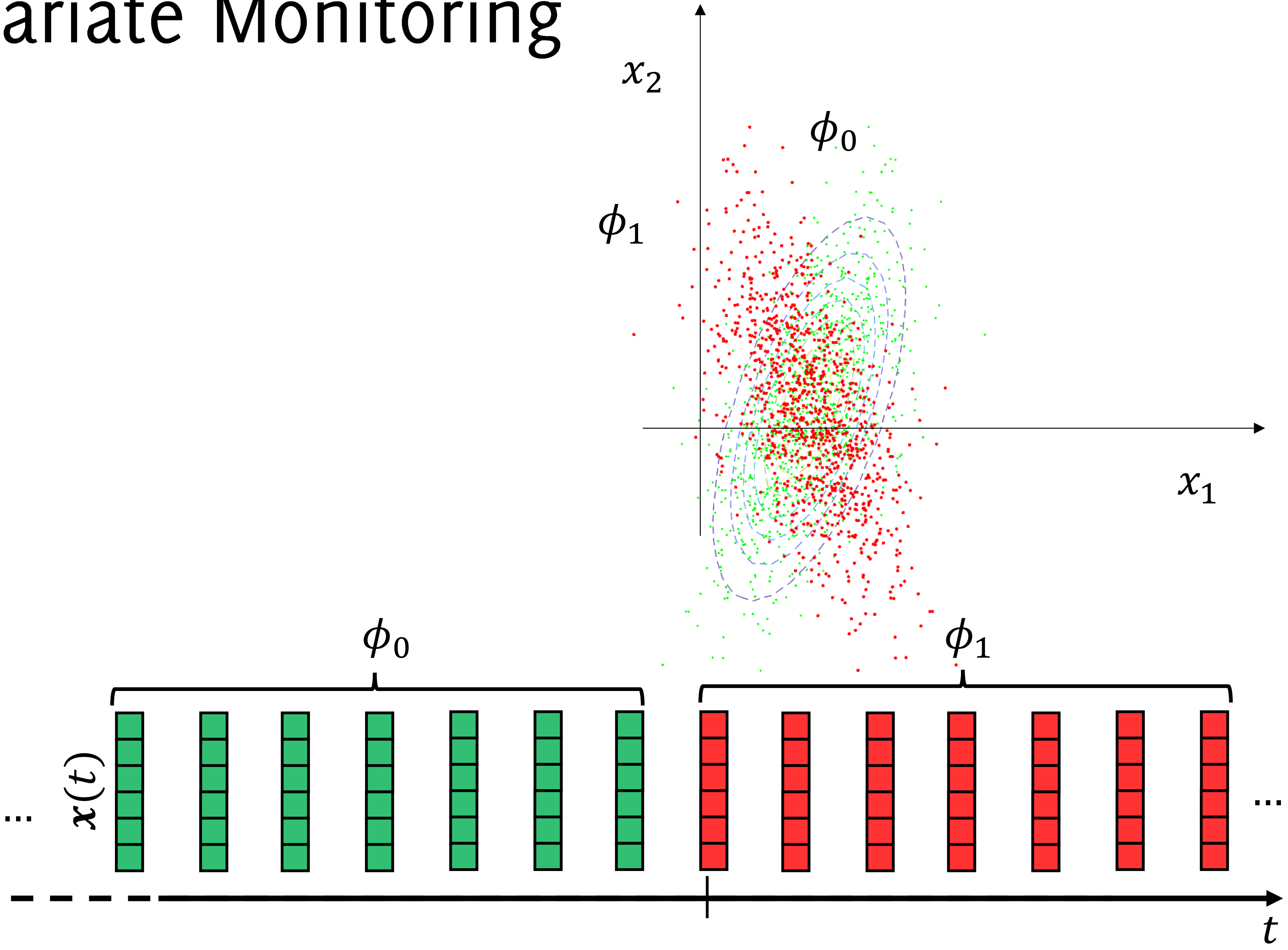


Multivariate Monitoring

Sometimes, monitoring the distribution of covariance is a viable option!



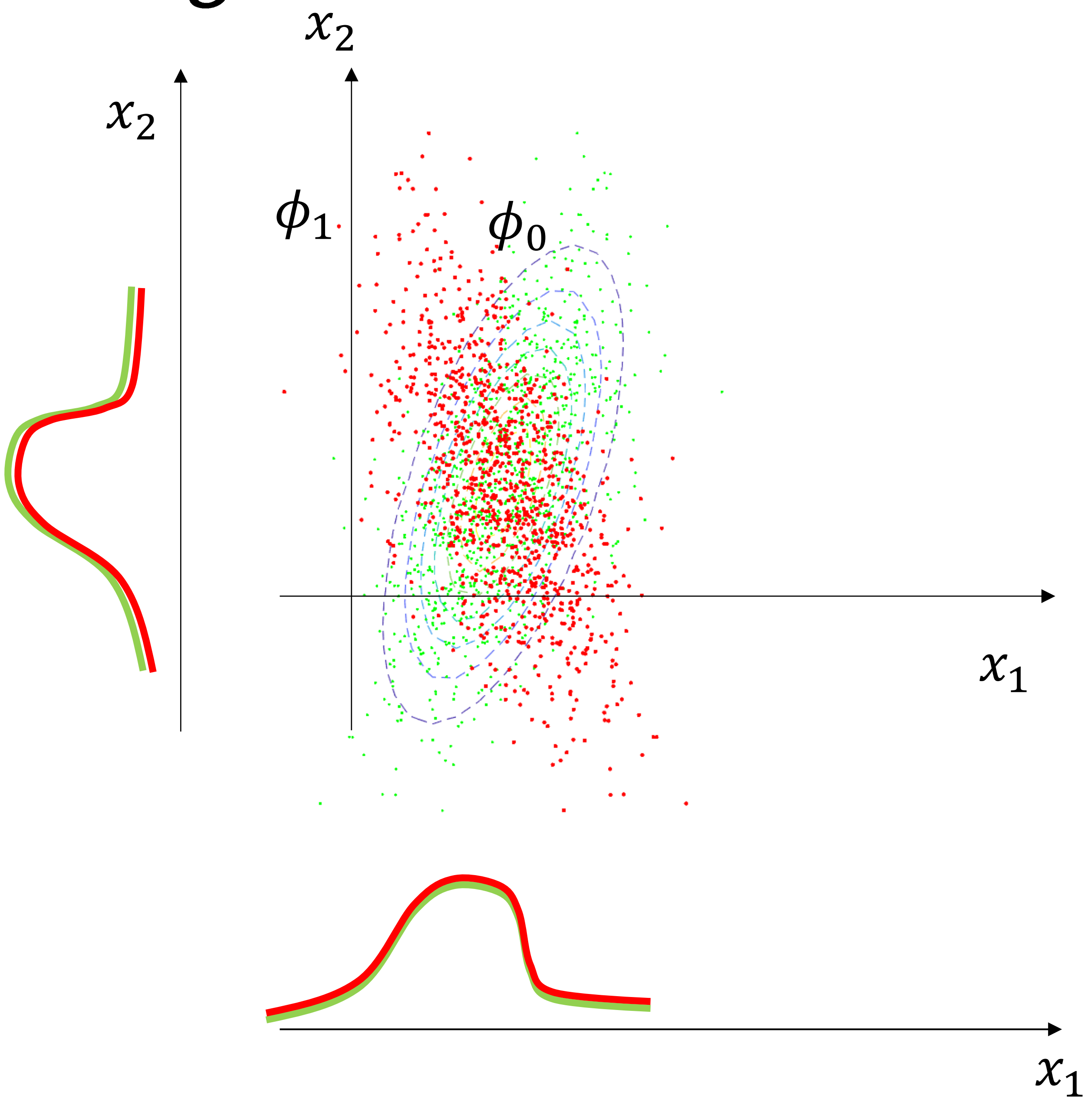
Multivariate Monitoring



Giacomo Boracchi

Multivariate Monitoring

Monitoring the distribution of covariance is not always a viable option



The Mainstream Change-Detection Approach

Three ingredients

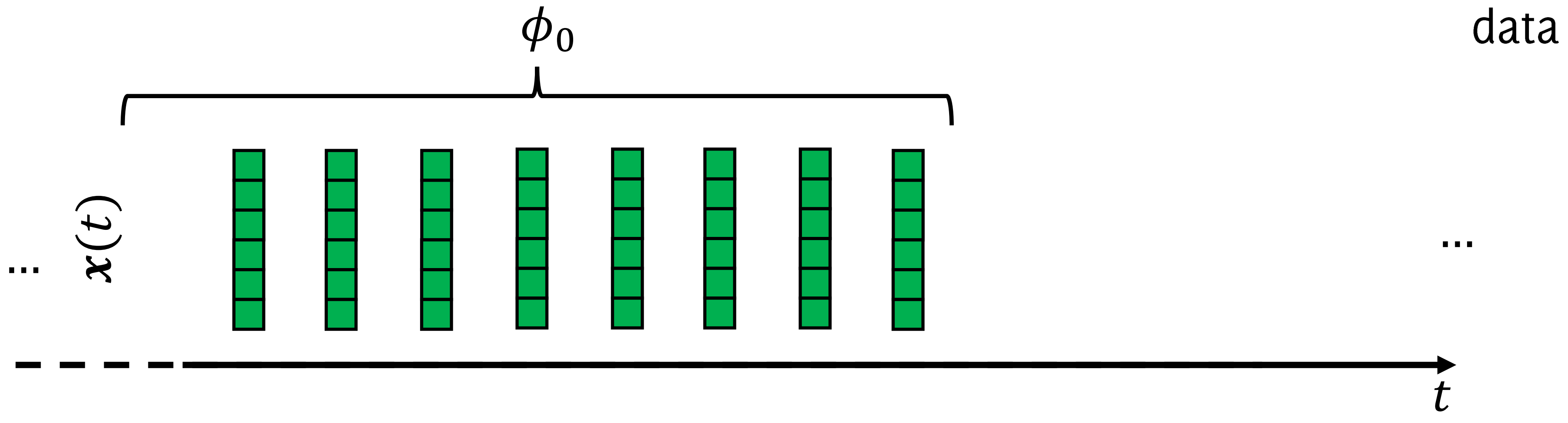
Most change-detection algorithm consists in

i. A model $\hat{\phi}_0$ describing ϕ_0

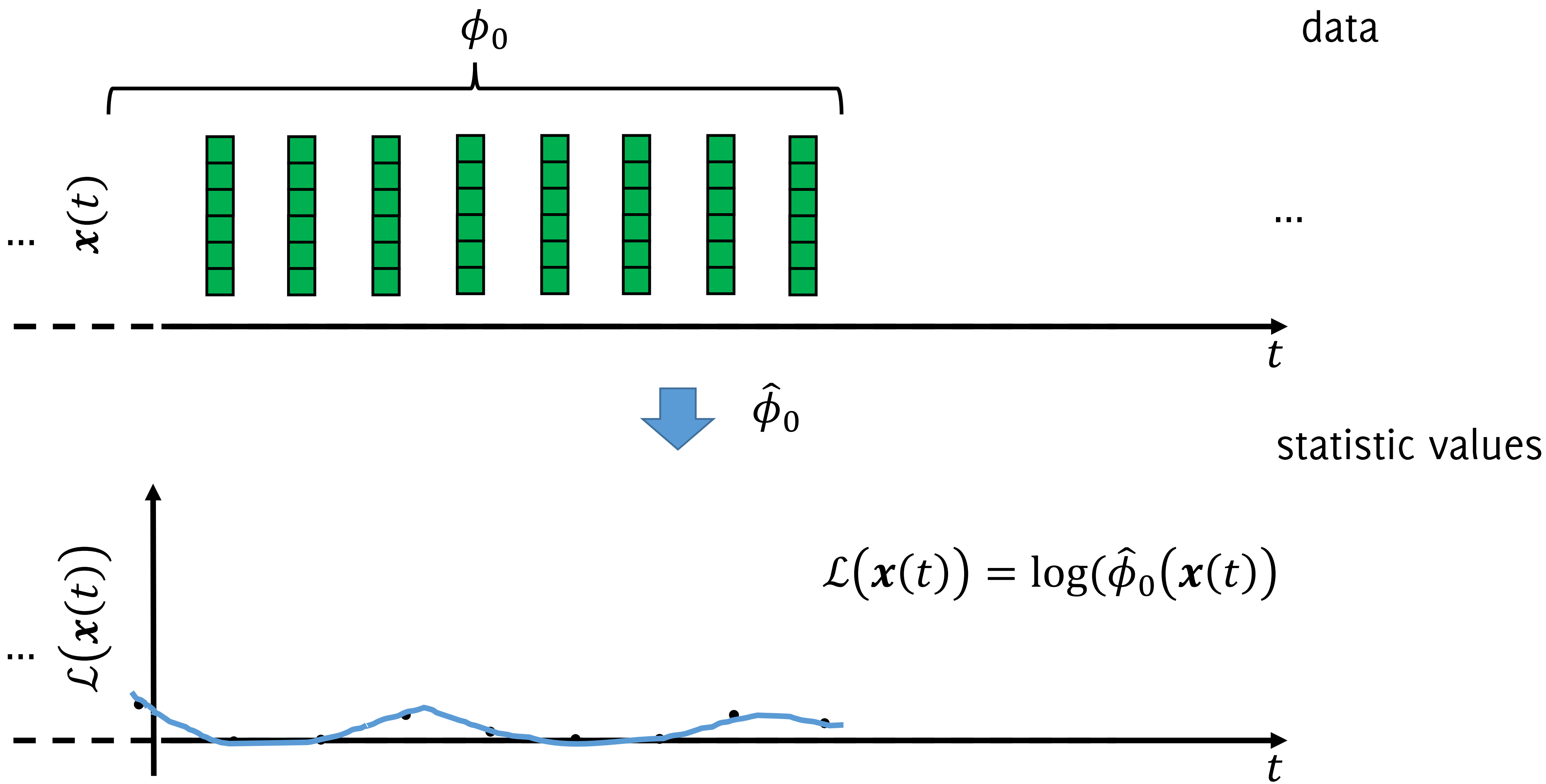
ii. A statistic $\mathcal{T}$ to test incoming data

iii. A (sequential) decision rule that monitors $\mathcal{T}$ to detect changes

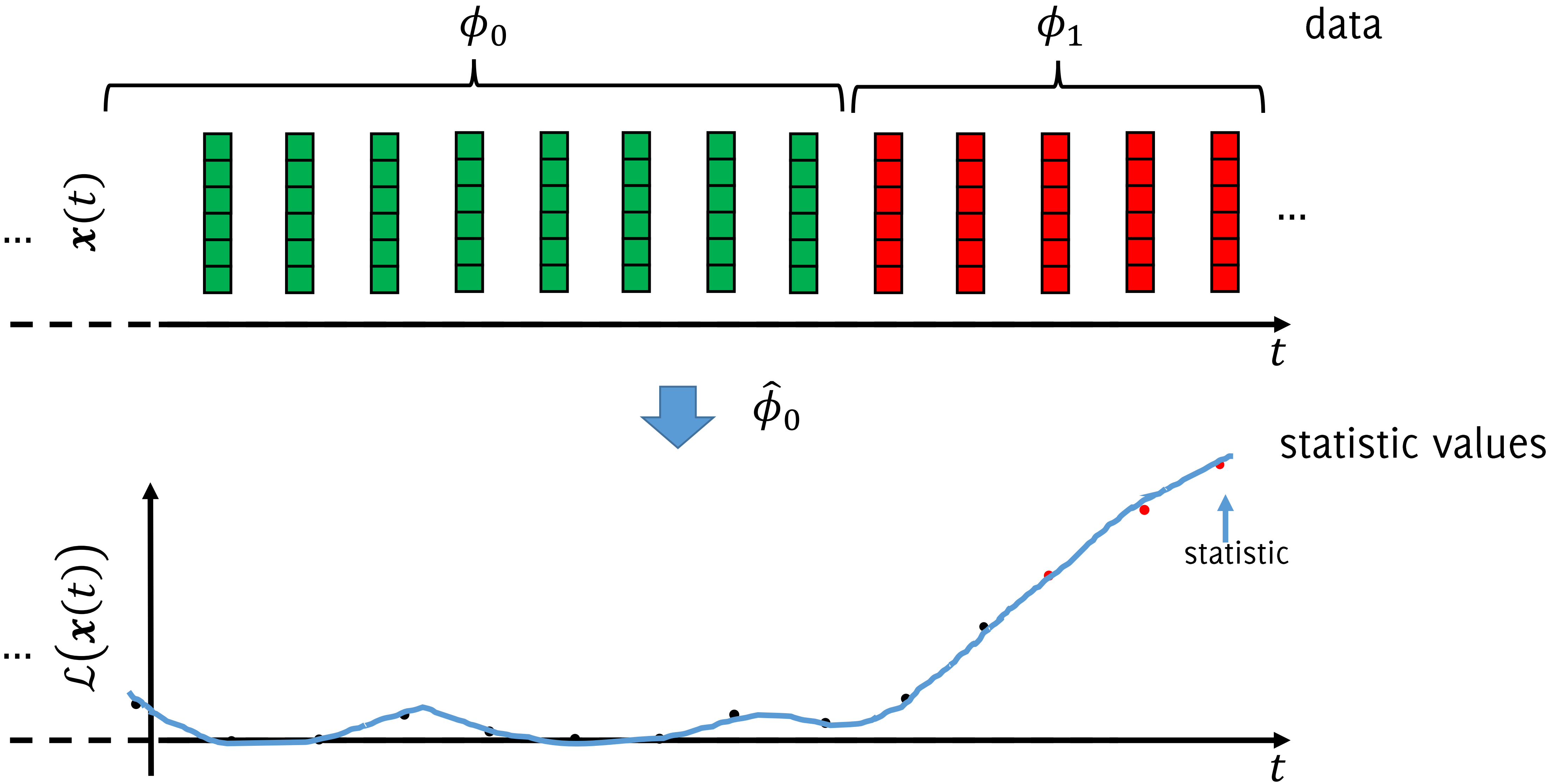
Illustration



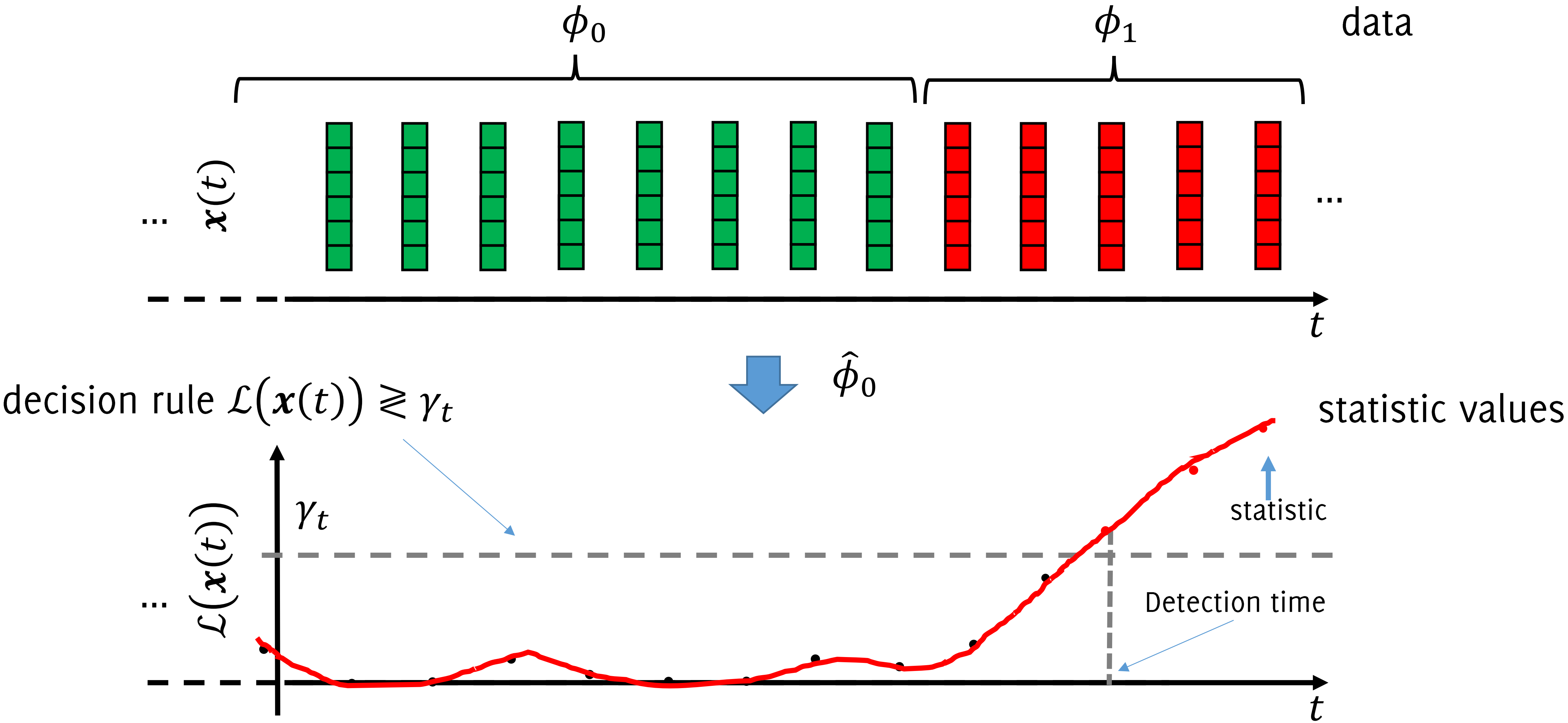
Illustration



Illustration



Illustration



Desiderata, Challenges and Goals

Desiderata in change detection

i. The model $\hat{\phi}_0$ describing ϕ_0 has to be:
   - general and simple
   - learnable from a training set

ii. The statistic $\mathcal{T}$ used to test incoming data has to:
   - provide a controlled response under ϕ_0
   - provide a different response under ϕ_1

iii. A decision rule that monitors $\mathcal{T}$ has to:
   - promptly detect changes and
   - control FPR (type I error in hypothesis testing) or ARL (averge run length in sequential monitoring)

The challenges we address

Most of the research has been devoted to **univariate** monitoring schemes:

- These are the historical settings in SPC
- Extension to monitoring classification / regression error are straightforward
- Nonparametric statistics (i.e., statistics that do not assume ϕ_0 known) are typically based on ranking, thus limited to $1d$ case.

- Parametric models $\hat{\phi}_0$ properly matching ϕ_0 are difficult to find
- Non-parametric models often require:
  - prohibitively large training sets
  - prohibitively long computing times

Our Goal

Build a model $\hat{\phi}_0$ and a truly multivariate monitoring scheme that:

- allows change detection in **multivariate**, possibly **high dimensional** data
- guarantees a **control over the false positives** without any assumption on ϕ_0
- requires only **little training data** for configuration
- it is **efficient** to test

Our Goal

Build a model $\hat{\phi}_0$ and a truly multivariate monitoring scheme that:

- allows change detection in **multivariate**, possibly **high dimensional** data
- guarantees a **control over the false positives** without any assumption on ϕ_0
- requires only **little training data** for configuration
- it is **efficient** to test

We adopt **histograms** to build the model $\hat{\phi}_0$ describing the distribution of stationary data.
There is a **lot of flexibility** in designing a histogram!

We've found a way to make change-detection easier: QuantTree

QuantTrees:
Histograms for Change Detection

A partitioning scheme specifically
designed for change detection

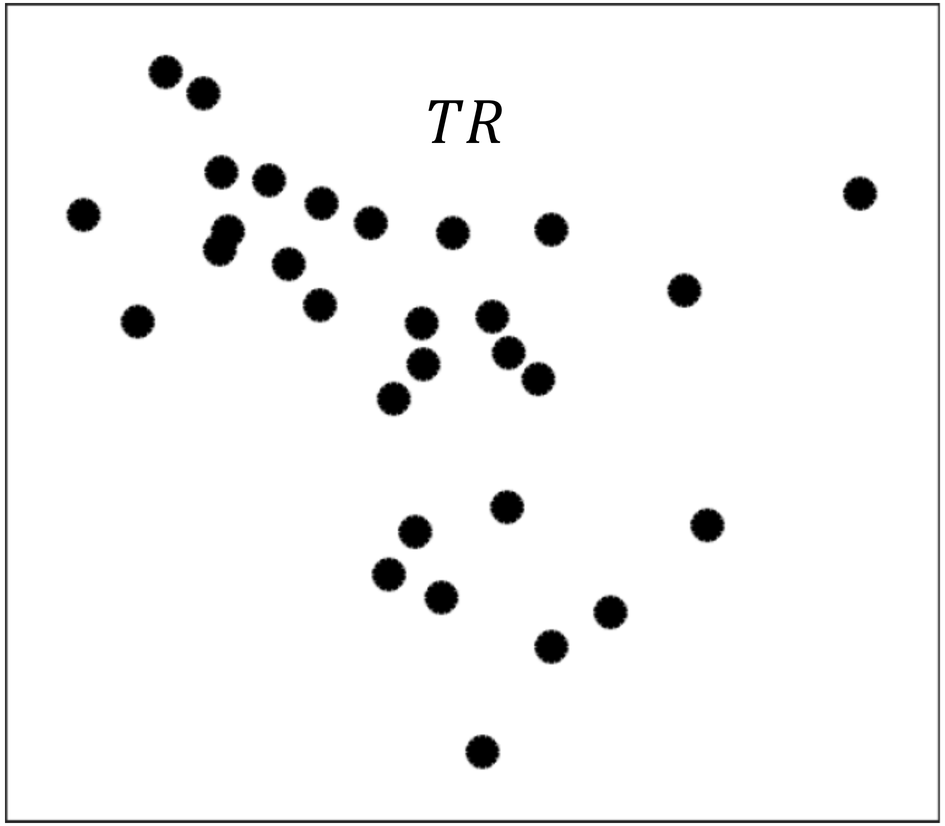
G. Boracchi, D. Carrera, C. Cervellera, D. Macciò *"QuantTree: Histograms for Change Detection in Multivariate Data Streams"* ICML 2018

QuantTree: Histograms for Change Detection in Multivariate Data Streams

Giacomo Boracchi [1] Diego Carrera [1] Cristiano Cervellera [2] Danilo Macciò [2]

QuantTrees: Histograms for change detection

Assume you are given a set of target probabilities $\{\pi_i\}_{i=1,..,K}$ and a training set TR



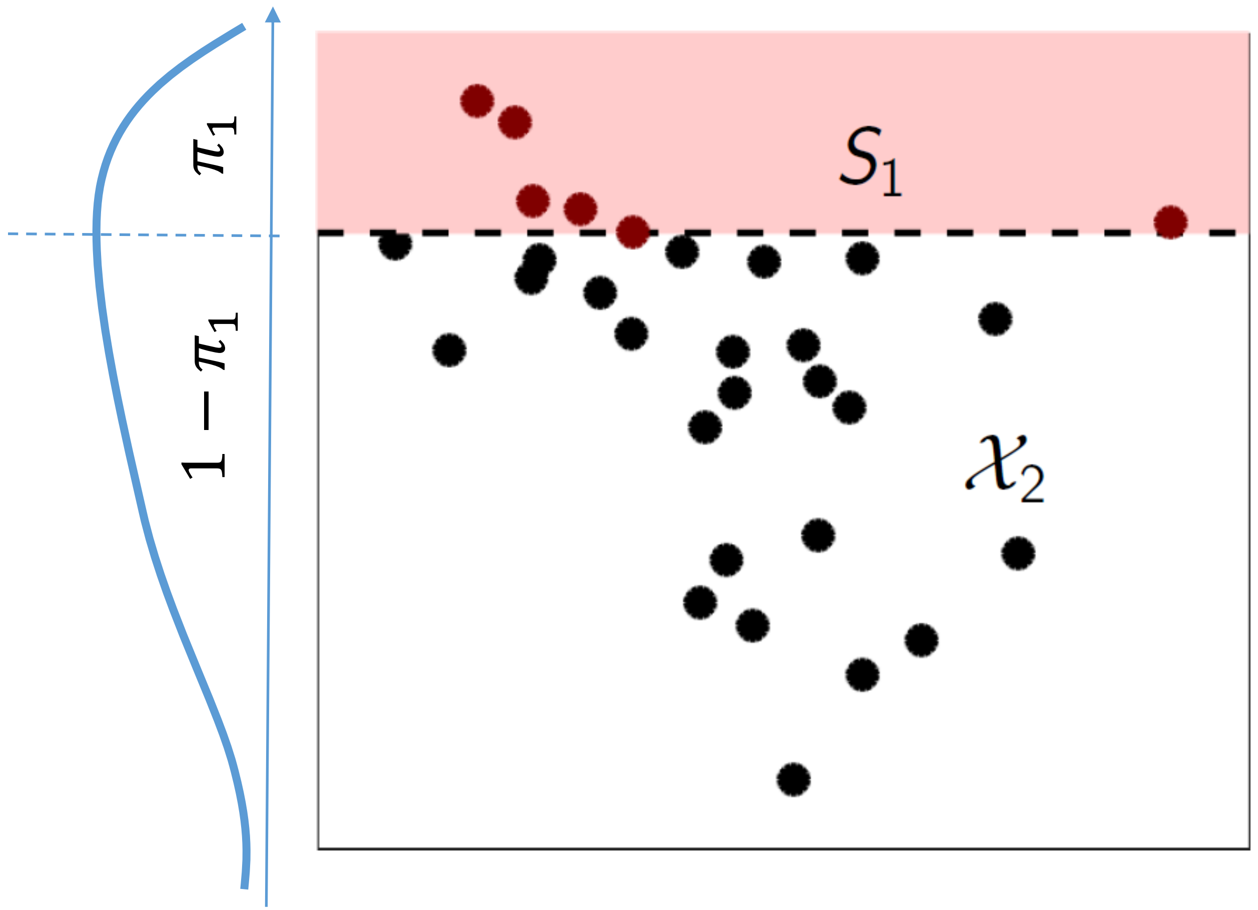
TR

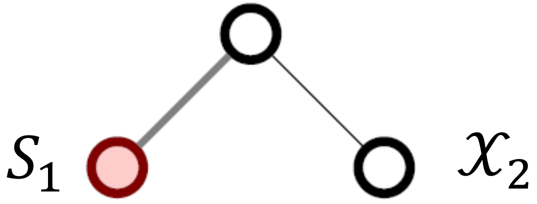
Inputs:
- K the number of bins,
- $\{\pi_i\}_{i=1,..,K}$ the bin probabilities

G. Boracchi, D. Carrera, C. Cervellera, D. Macciò *"QuantTree: Histograms for Change Detection in Multivariate Data Streams"* ICML 2018

QuantTrees: Histograms for change detection

Choose a dimension j at random, define the S_1 as the set containing the $1 - \pi_1$ quantile of the marginal distribution of training samples along j



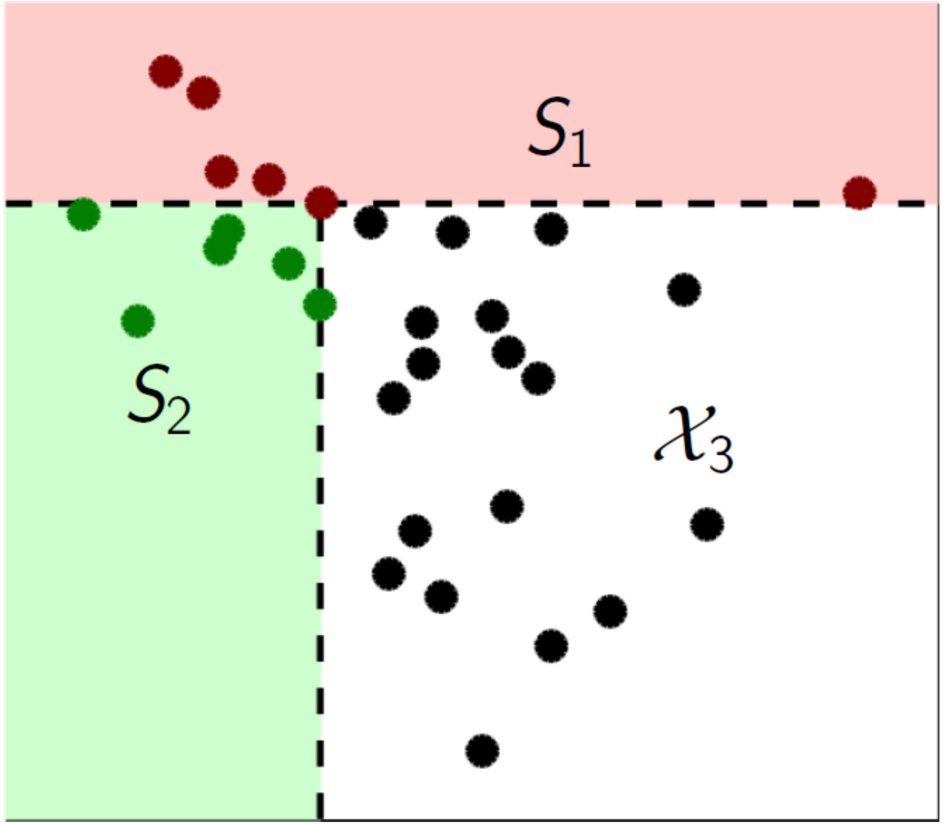


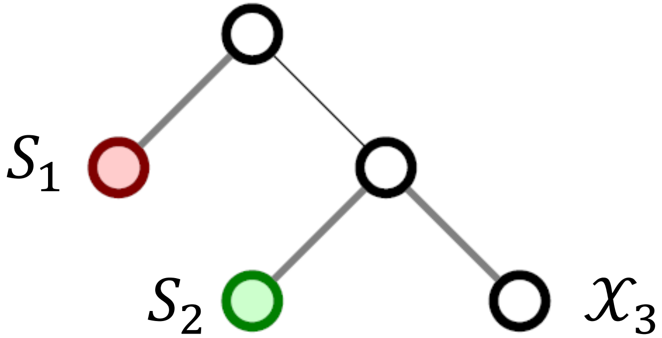
Call $\mathcal{X}_2$ the remaining samples

G. Boracchi, D. Carrera, C. Cervellera, D. Macciò *"QuantTree: Histograms for Change Detection in Multivariate Data Streams"* ICML 2018

QuantTrees: Histograms for change detection

The procedure is iterated on the training samples that have not been included in a bin.

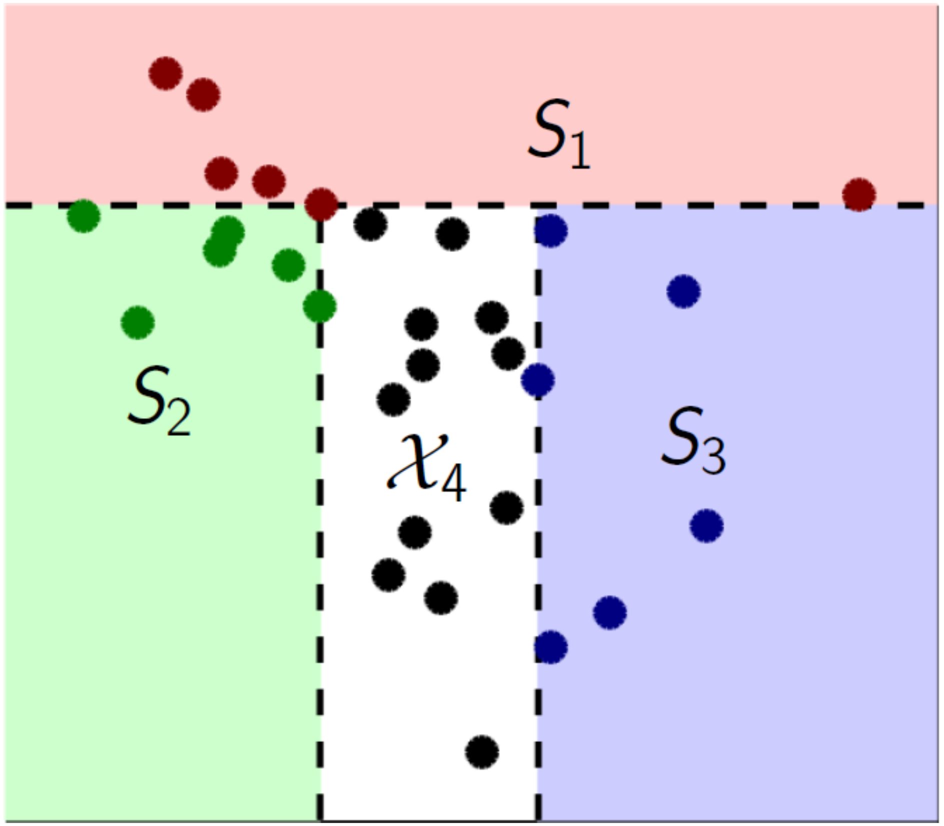


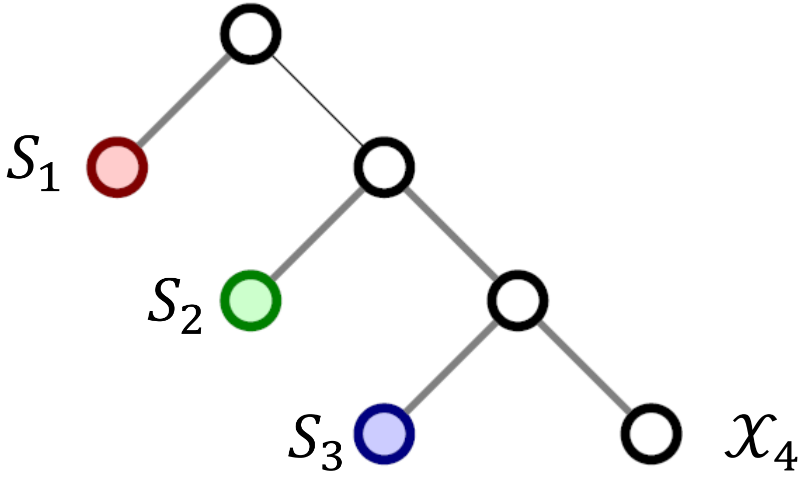


G. Boracchi, D. Carrera, C. Cervellera, D. Macciò *"QuantTree: Histograms for Change Detection in Multivariate Data Streams"* ICML 2018

QuantTrees: Histograms for change detection

The procedure is iterated on the training samples that have not been included in a bin.

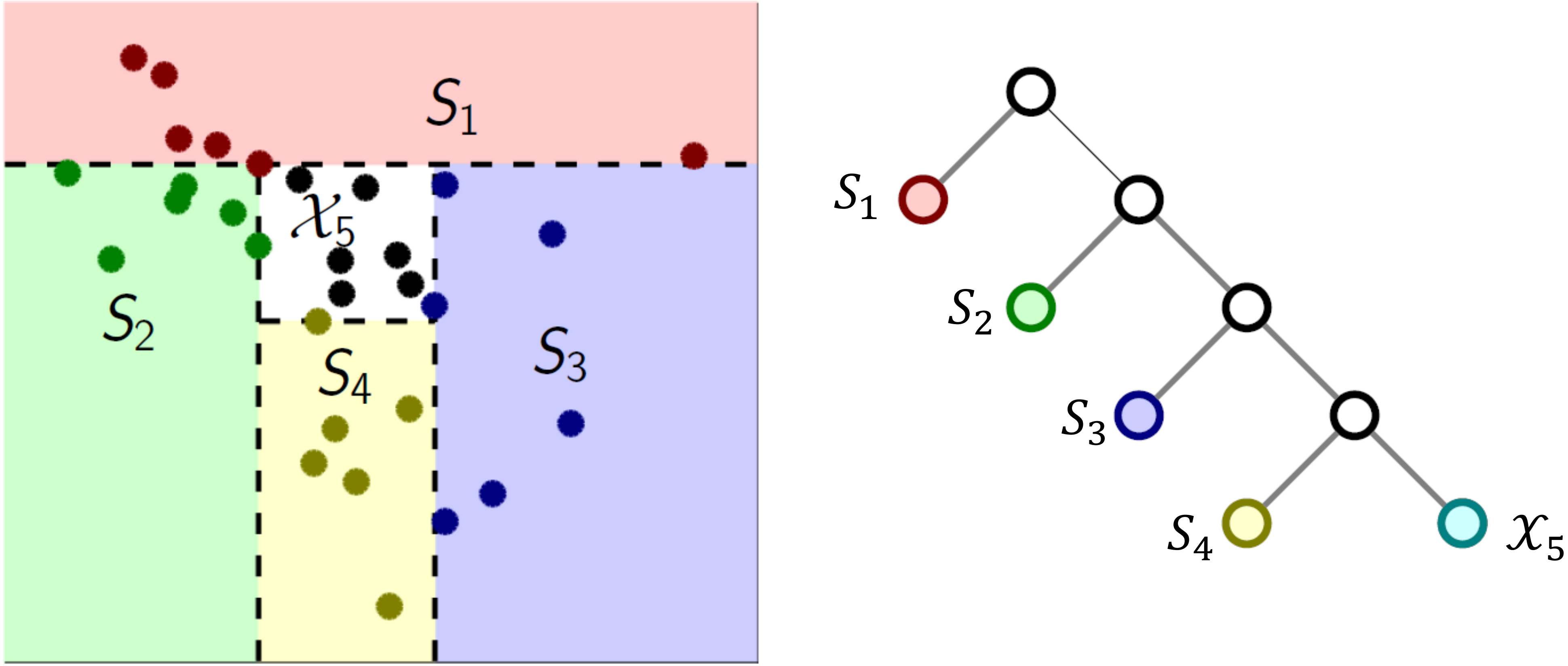




G. Boracchi, D. Carrera, C. Cervellera, D. Macciò *"QuantTree: Histograms for Change Detection in Multivariate Data Streams"* ICML 2018

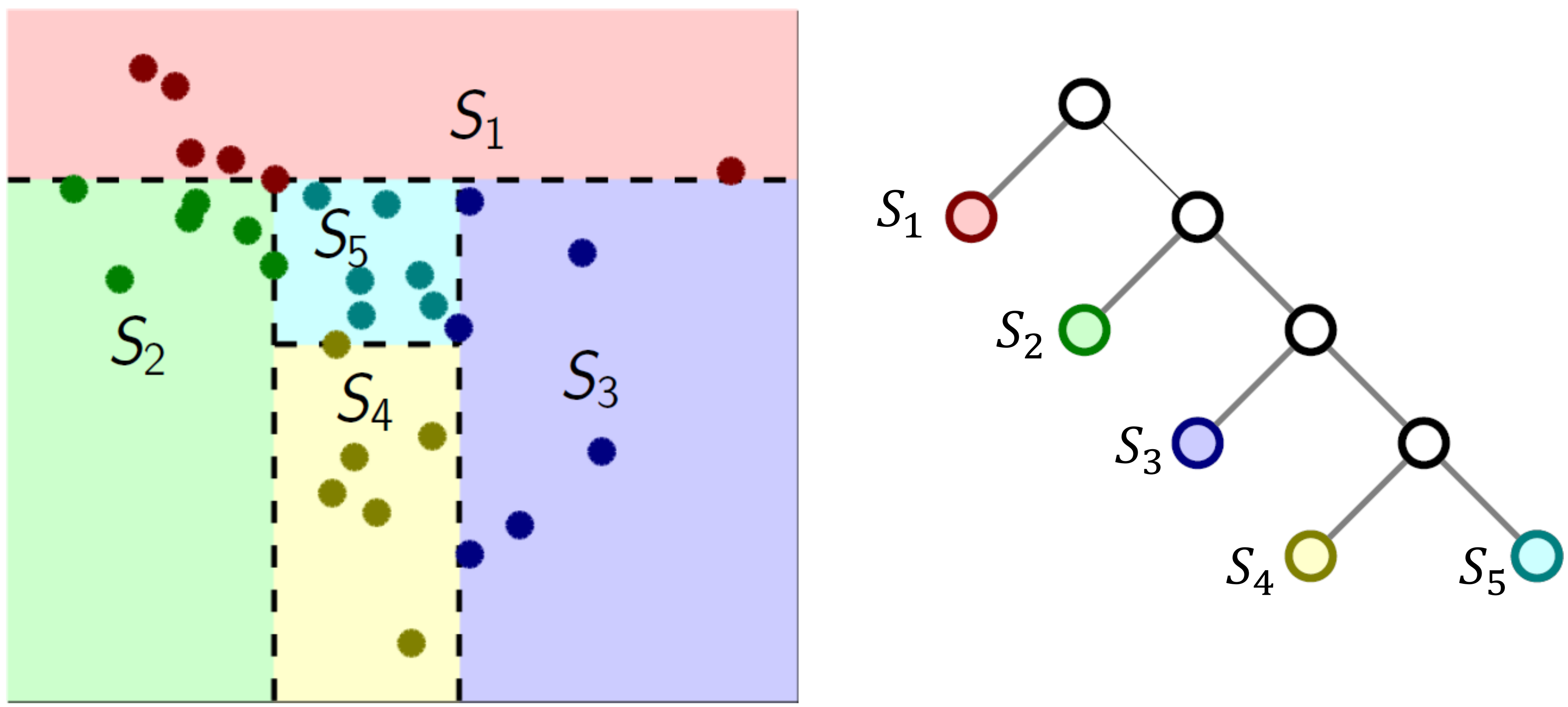
QuantTrees: Histograms for change detection

The procedure is iterated on the training samples that have not been included in a bin.



QuantTrees: Histograms for change detection

The procedure is iterated on the training samples that have not been included in a bin.



G. Boracchi, D. Carrera, C. Cervellera, D. Macciò *"QuantTree: Histograms for Change Detection in Multivariate Data Streams"* ICML 2018

QuantTree Construction

QuantTree **iteratively divides** the input space by **binary splits along a single covariate**, where the cutting points are defined by the **quantiles of the marginal distributions**

Algorithm 1 QuantTree

Input: Training set TR containing N stationary points in $\mathcal{X}$; number of bins K; target probabilities $\{\pi_k\}_k$.

Output: The histogram $h = \{(S_k, \widehat{\pi}_k)\}_k$.

Set $N_0 = N$, $L_0 = 0$.
for $k = 1, \dots, K$ **do**
  Set $N_k = N_{k-1} - L_{k-1}$, $\mathcal{X}_k = \mathcal{X} \setminus \bigcup_{j<k} S_j$, and $L_k = \text{round}(\pi^k N)$.
  Choose a random component $i \in \{1, \dots, d\}$.
  Define $z_n = [\mathbf{x}_n]_i$ for each $\mathbf{x}_n \in \mathcal{X}_k$.
  Sort $\{z_n\}$: $z_{(1)} \leq z_{(2)} \leq \dots z_{(N_k)}$.
  Draw $\gamma \in \{0, 1\}$ from a Bernoulli(0.5).
  if $\gamma = 0$ **then**
    Define $S_k = \{\mathbf{x} \in \mathcal{X}_k \quad [\mathbf{x}]_i \leq z_{(L_k)}\}$.
  else
    Define $S_k = \{\mathbf{x} \in \mathcal{X}_k \quad [\mathbf{x}]_i \geq z_{(N_k - L_k + 1)}\}$.
  end if
  Set $\widehat{\pi}_k = L_k / N$.
end for

G. Boracchi, D. Carrera, C. Cervellera, D. Macciò *"QuantTree: Histograms for Change Detection in Multivariate Data Streams"* ICML 2018

QuantTree Construction

QuantTree **iteratively divides** the input space by **binary splits along a single covariate**, where the cutting points are defined by the **quantiles of the marginal distributions**

The QuantTree construction is randomized by the random selection of the component for each split and whether to take the π_i or $1 - \pi_i$ quantile

Algorithm 1 QuantTree

Input: Training set TR containing N stationary points in $\mathcal{X}$; number of bins K; target probabilities $\{\pi_k\}_k$.

Output: The histogram $h = \{(S_k, \widehat{\pi}_k)\}_k$.

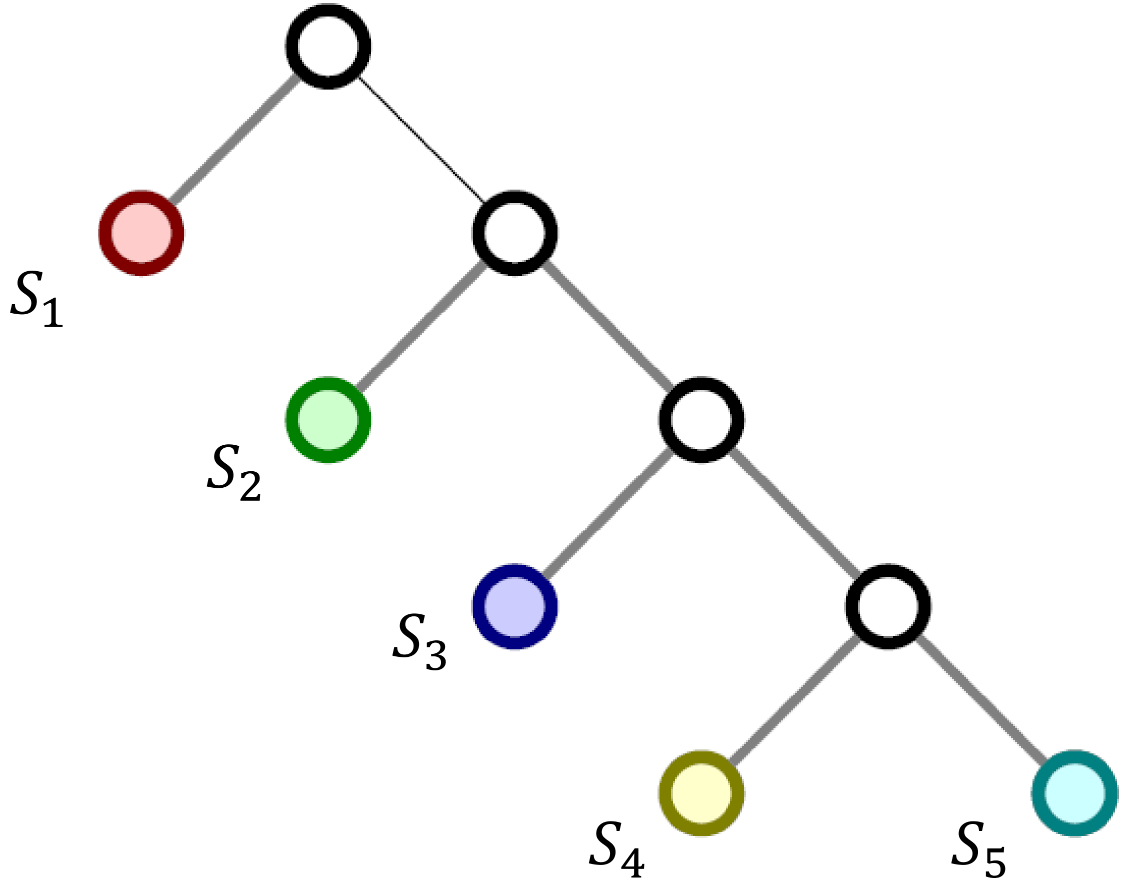
```
Set N_0 = N, L_0 = 0.
for k = 1, ..., K do
    Set N_k = N_{k-1} - L_{k-1}, X_k = X \ ∪_{j<k} S_j, and L_k = round(π^k N).
    Choose a random component i ∈ {1, ..., d}.
    Define z_n = [x_n]_i for each x_n ∈ X_k.
    Sort {z_n}: z_(1) ≤ z_(2) ≤ ... z_(N_k).
    Draw γ ∈ {0, 1} from a Bernoulli(0.5).
    if γ = 0 then
        Define S_k = {x ∈ X_k   [x]_i ≤ z_(L_k)}.
    else
        Define S_k = {x ∈ X_k   [x]_i ≥ z_(N_k - L_k + 1)}.
    end if
    Set π̂_k = L_k/N.
end for
```

QuantTree Partitioning

Each QuantTree produces a partitioning of the input domain $\mathcal{X}$

$$\{S_k, \hat{\pi}_k\}$$

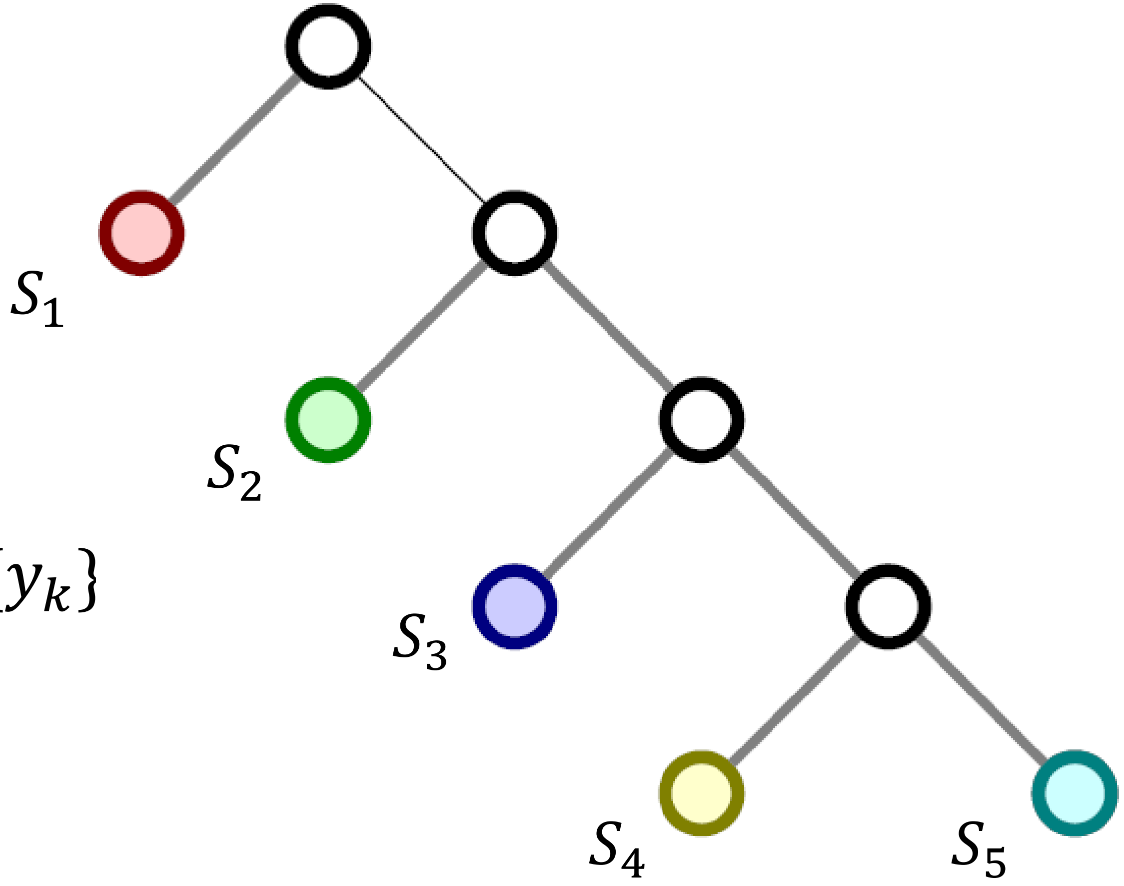
Where $\hat{\pi}_k$ are the probabilities estimated from TR, can slightly depart from the target $\{\pi_k\}$ (they match when $\pi_k N$ is an integer)



QuantTrees Test Statistic

Batch-wise change detection

1. Monitor a batch of ν test samples
$$W = \{x(t), \dots, x(t+\nu)\}$$
2. Dispatch samples in bins $\{S_k\}$ and compute the number of samples in each bin $\{y_k\}$
3. Compute **any test statistic** depending on $\{y_k\}$
$$e.g., \qquad \mathcal{T}_h(W) = \sum_{k=1}^{K} \frac{(y_k - \nu\pi_k)^2}{\nu\pi_k}$$
4. Compare it against a **threshold** γ
$$\mathcal{T}_h(W) > \gamma$$

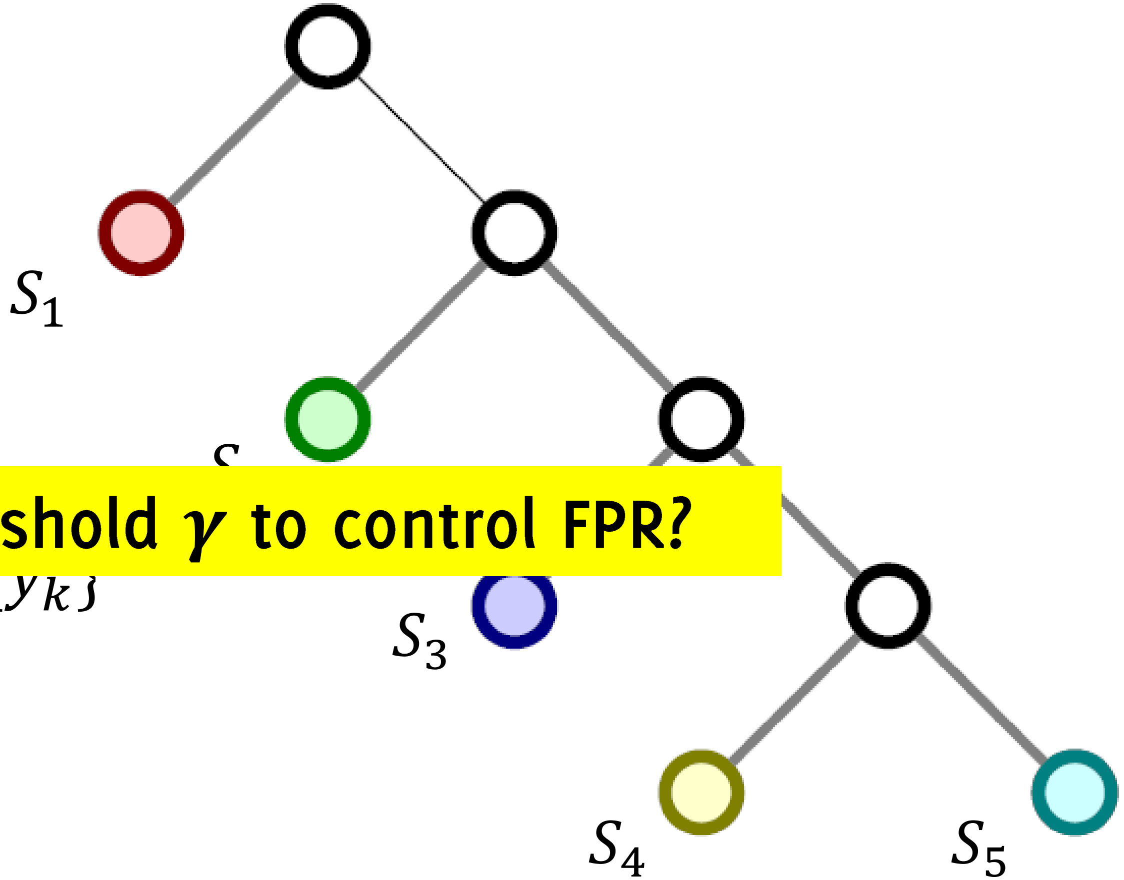


G. Boracchi, D. Carrera, C. Cervellera, D. Macciò *"QuantTree: Histograms for Change Detection in Multivariate Data Streams"* ICML 2018

QuantTrees Test Statistic

Batch-wise change detection

1. Monitor a batch of ν test samples
$$W = \{x(t), \dots, x(t+\nu)\}$$
2. Dispatch samples in bins $\{S_k\}$ and compute the number of samples in each bin $\{y_k\}$
3. Compute **any test statistic** depending on $\{y_k\}$
$$e.g., \qquad \mathcal{T}_h(W) = \sum_{k=1}^{K} \frac{(y_k - \nu\pi_k)^2}{\nu\pi_k}$$
4. Compare it against a **threshold** γ
$$\mathcal{T}_h(W) > \gamma$$



How can we set the detection threshold γ to control FPR?

G. Boracchi, D. Carrera, C. Cervellera, D. Macciò *"QuantTree: Histograms for Change Detection in Multivariate Data Streams"* ICML 2018

QuantTrees Statistics

Theorem (ICML18)

Let $T_h(\cdot)$ be a statistic defined over the bin probabilities of a histogram h computed by QuantTree.

For any stationary batch $W \sim \phi_0$, the distribution of $T_h(W)$ depends only on:

- *the number of training samples $N = \#TR$,*
- *the batch size W,*
- *the expected probabilities in each bin $\{\pi_i\}_{i=1,\dots,K}$*

Implications

In histograms constructed by QuantTrees, **test statistics do not depend on ϕ_0, nor data dimension d.** No need of bootstrap, small TR viable.

Detection threshold γ can be numerically computed from synthetic data:

1. Generating data according to a 1D ψ_0 (e.g., ψ_0 is uniform $[0,1]$)
2. Define a QT histogram $h = \{S_k, \pi_k\}$ on TR
3. Generate stationary test batches $W \sim \psi_0$, the test statistic
4. Compute the threshold γ from the empirical distribution of $T_h(W)$

α	Pearson		Total Variation			
	$K = 32$	$K = 128$	$K = 32$	$K = 128$	N	ν
0.001	64	192	25	43	4096	64
	62.75	187	52	85	16384	256
0.01	54	172	23	42	4096	64
	53.25	171	47	81	16384	256
0.05	46	156	21	41	4096	64
	45.75	157	44	78	16384	256

Example of Thresholds γ

G. Boracchi, D. Carrera, C. Cervellera, D. Macciò *"QuantTree: Histograms for Change Detection in Multivariate Data Streams"* ICML 2018

Implications

In histograms constructed by QuantTrees, the bin probabilities do not depend on ϕ_0, nor data dimension d.

Thus, **thresholds** of tests statistics **can be numerically computed from univariate data** that have been synthetically generated yet guaranteeing a controlled false positive rate.

	$d > 1$	$d = 1$
Training	$O(KN \log N)$	$O(N \log N)$
Test	$O(K)$	$O(\log K)$

Change Detection By QuantTrees

Training:

- Define a QT $h = \{S_k, \hat{\pi}_k\}$ from TR with target probabilities $\{\pi_i\}_{i=1,\dots,K}$
- Compute threshold γ on synthetic data using $\{\hat{\pi}_k\}_{i=1,\dots,K}$, ν, $N = \#TR$

Testing

- Gather a batch of test samples W
- Compute the test statistic

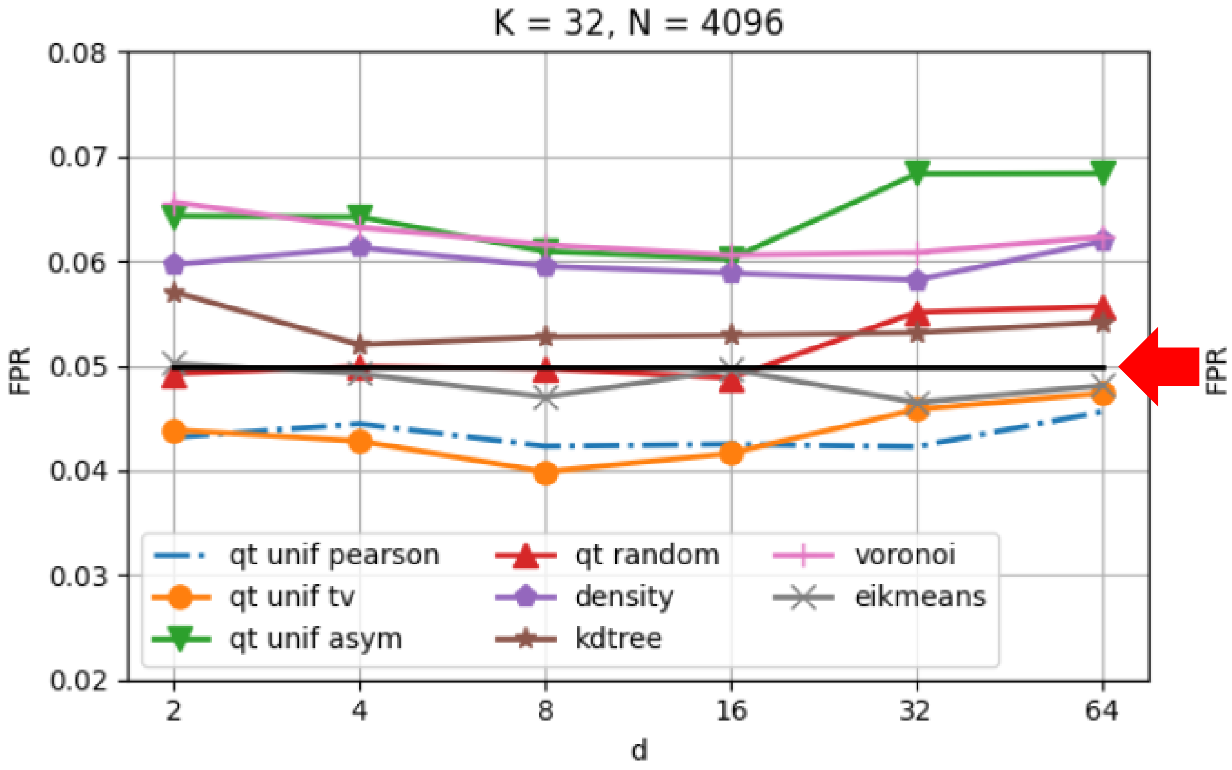
$$\mathcal{T}_h(W) = \sum_{k=1}^{K} \frac{(y_k - \nu\pi_k)^2}{\nu\pi_k}$$

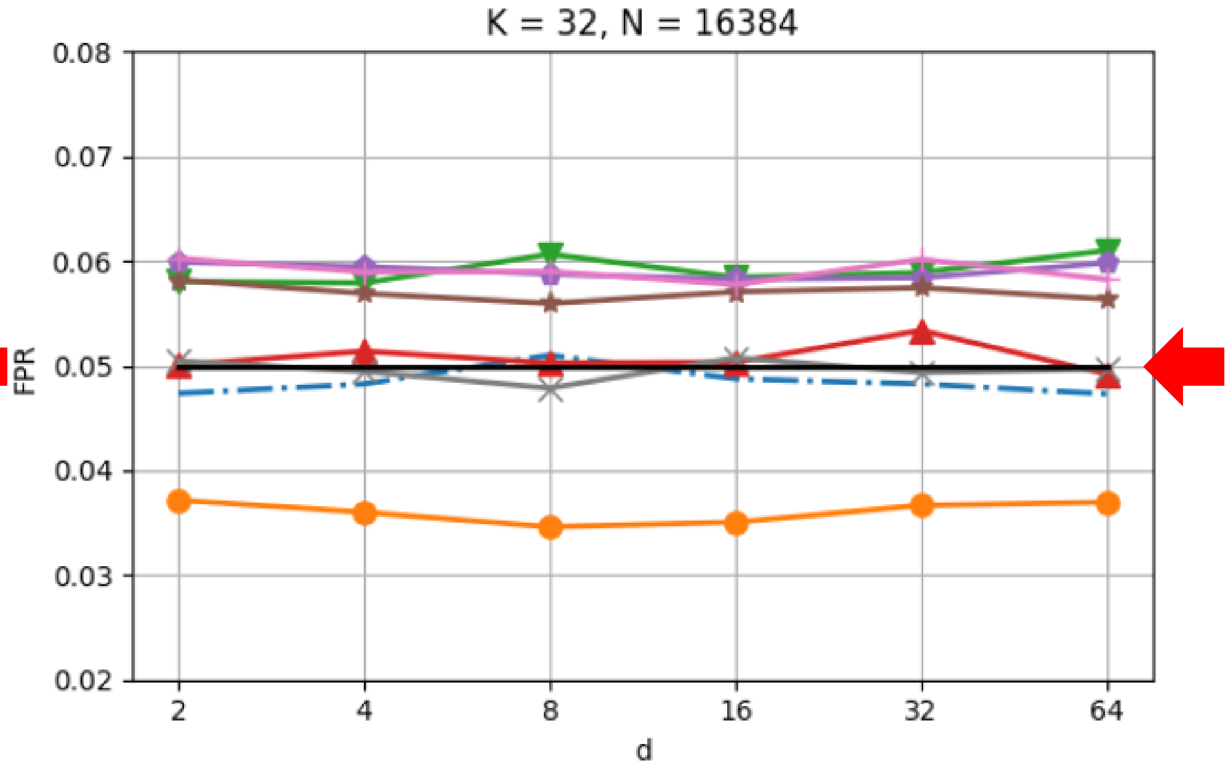
- Detect a change when $\mathcal{T}_h(W) > \gamma$

G. Boracchi, D. Carrera, C. Cervellera, D. Macciò *"QuantTree: Histograms for Change Detection in Multivariate Data Streams"* ICML 2018

Experiments on False Positive Control

QT algorithms can control FPR (target $\alpha = 0.05$) without resorting to bootstrap and better than asymptotic approximation

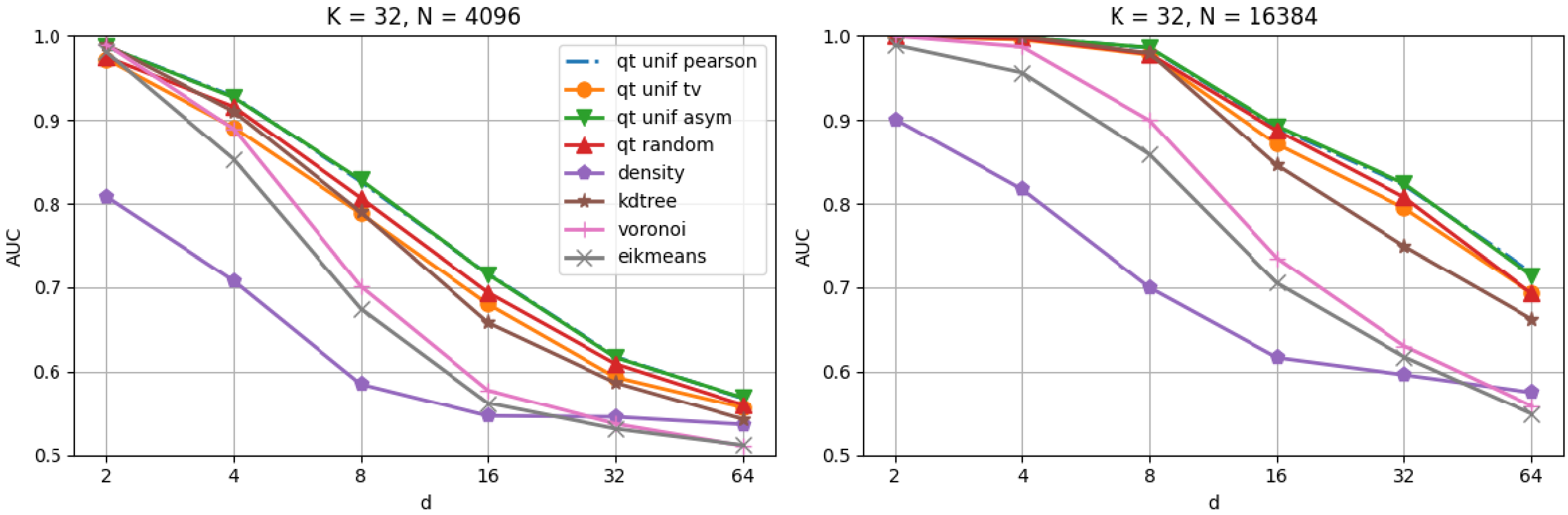




Test on synthetic data ϕ_0 is a Gaussian. High dispersion in statitsics from random bin probabilities $\{\pi_k\}$

Experiments on Detection Power (AUC)

- QT with Pearson Statistics are among the most powerful CD algorithms
- Uniform bin probabilities $\pi_k = 1/K$ are better than random probabilities



Test on synthetic data such as $sKL(\phi_0, \phi_1) = 1$

Experiments on Real World Datasets

dataset		qt unif pearson	qt unif asym	qt unif tv	kdtree	voronoi	density	qt random	eikmeans
particle	FPR	0.042	0.065	0.044	0.053	0.063	0.057	0.054	0.049
	AUC	0.876	**0.886**	0.865	0.841	0.530	0.529	0.842	0.512
protein	FPR	0.046	0.064	0.046	0.055	0.065	0.059	0.050	0.047
	AUC	**0.978**	**0.978**	0.972	0.969	0.564	0.591	0.962	0.527
credit	FPR	0.045	0.064	0.046	0.051	0.060	0.061	0.054	0.049
	AUC	0.800	**0.810**	0.781	0.788	0.532	0.721	0.753	0.515
sensorless	FPR	0.043	0.063	0.044	0.053	0.058	0.059	0.055	0.050
	AUC	**1.000**	**1.000**	**1.000**	**1.000**	0.517	0.627	**1.000**	0.503
nino	FPR	0.041	0.063	0.042	0.053	0.064	0.058	0.050	0.047
	AUC	**0.833**	0.825	0.811	0.819	0.558	0.546	0.802	0.543
spruce	FPR	0.042	0.067	0.041	0.056	0.065	0.058	0.052	0.050
	AUC	**1.000**	**1.000**	**1.000**	**1.000**	0.560	**1.000**	**1.000**	0.509
lodgepole	FPR	0.043	0.061	0.045	0.053	0.066	0.062	0.052	0.051
	AUC	**1.000**	**1.000**	**1.000**	**1.000**	0.580	**1.000**	**1.000**	0.517
insects	FPR	0.042	0.063	0.043	0.052	0.062	0.058	0.051	0.049
	AUC	0.912	0.910	0.892	0.854	0.897	**0.994**	0.877	0.854

Table 2: Results for the QuantTree algorithm on real datasets for $N = 4096$, $K = 32$. For each dataset, the FPR and AUC are repoted, averaged over 100 runs for each method.

Also on real world datasets, QT can control the FPR and is very powerful!

Another practical result about QuantTree Threshold

QuantTree Statistics

Theorem (TKDE22)

Let $h = \{S_k, \pi_k\}$ *be a partitioning of the input domain in* K *bins built using the QuantTree algorithm with target probabilities* $\{\pi_k\}_{k=1,\dots,K}$.

Let p_k *be the expected probability of* S_k *under* ϕ_0*, namely* $p_k = P_{\phi_0}(S_k)$.

Then, the probabilities $(p_1, \dots, p_K)$ *follow a Dirichlet distribution*

$$(p_1, \dots, p_K) \sim D\left(\pi_1 N, \pi_2 N, \dots, \left(1 - \sum_{j=1}^{K-1} \pi_j\right) N + 1\right)$$

QuantTree Statistics

Theorem (TKDE22)

Let $h = \{S_k, \pi_k\}$ be a partitioning of the input domain in K bins built using the QuantTree algorithm with target probabilities $\{\pi_k\}_{k=1,\dots,K}$.

Let p_k be the expected probability of S_k under ϕ_0, namely $p_k = P_{\phi_0}(S_k)$.

Then, the probabilities $(p_1, \dots, p_K)$ follow a Dirichlet distribution

$$(p_1, \dots, p_K) \sim D\left(\pi_1 N, \pi_2 N, \dots, \left(1 - \sum_{j=1}^{K-1} \pi_j\right) N + 1\right)$$

The probabilities of the K bins of a QuantTree under any ϕ_0

The QuantTree parameters

QuantTree Statistics

Theorem (TKDE22)

Let $h = \{S_k, \pi_k\}$ *be a partitioning of the input domain in* K *bins built using the QuantTree algorithm with target probabilities* $\{\pi_k\}_{k=1,\dots,K}$.

Let p_k *be the expected probability of* S_k *under* ϕ_0, *namely* $p_k = P_{\phi_0}(S_k)$.

Then, the probabilities $(p_1, \dots, p_K)$ *follow a Dirichlet distribution*

$$(p_1, \dots, p_K) \sim D\left(\pi_1 N, \pi_2 N, \dots, \left(1 - \sum_{j=1}^{K-1} \pi_j\right) N + 1\right)$$

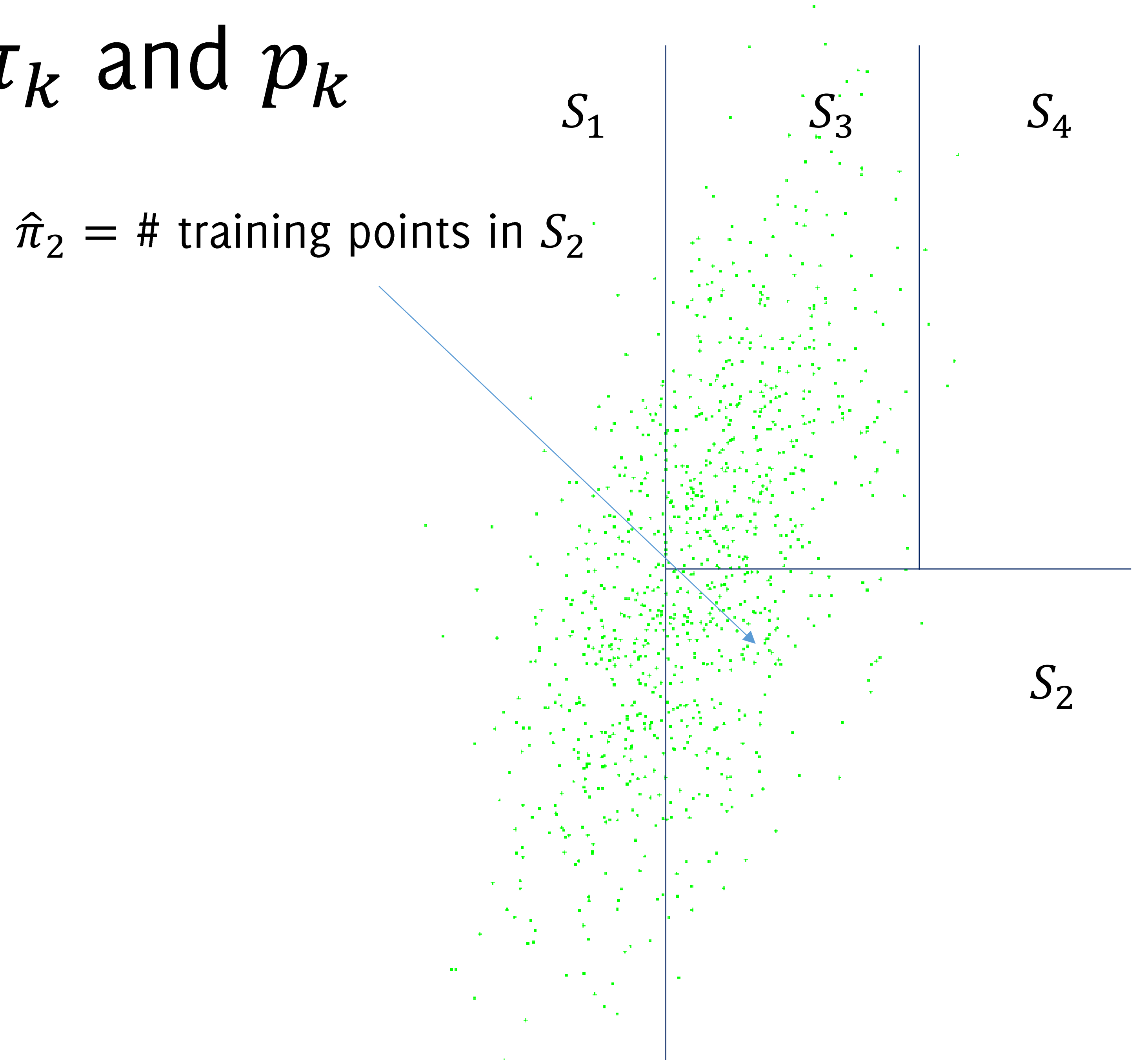
No need to sample points, no need to construct histograms!
We can directly draw the bin probabilities a QuantTree would produce!

L. Frittoli, D. Carrera, G. Boracchi *"Nonparametric and Online Change Detection in Multivariate Datastreams using QuantTree"* IEEE TKDE 2022

Differences between π_k and p_k

π_k and $\hat{\pi}_k$ represent the empirical frequency of points in the bin S_k. Sometimes they do coincide (often we assume they do)

These are used to construct the QuantTree histogram, but might not corresponds to the true bin probabilities

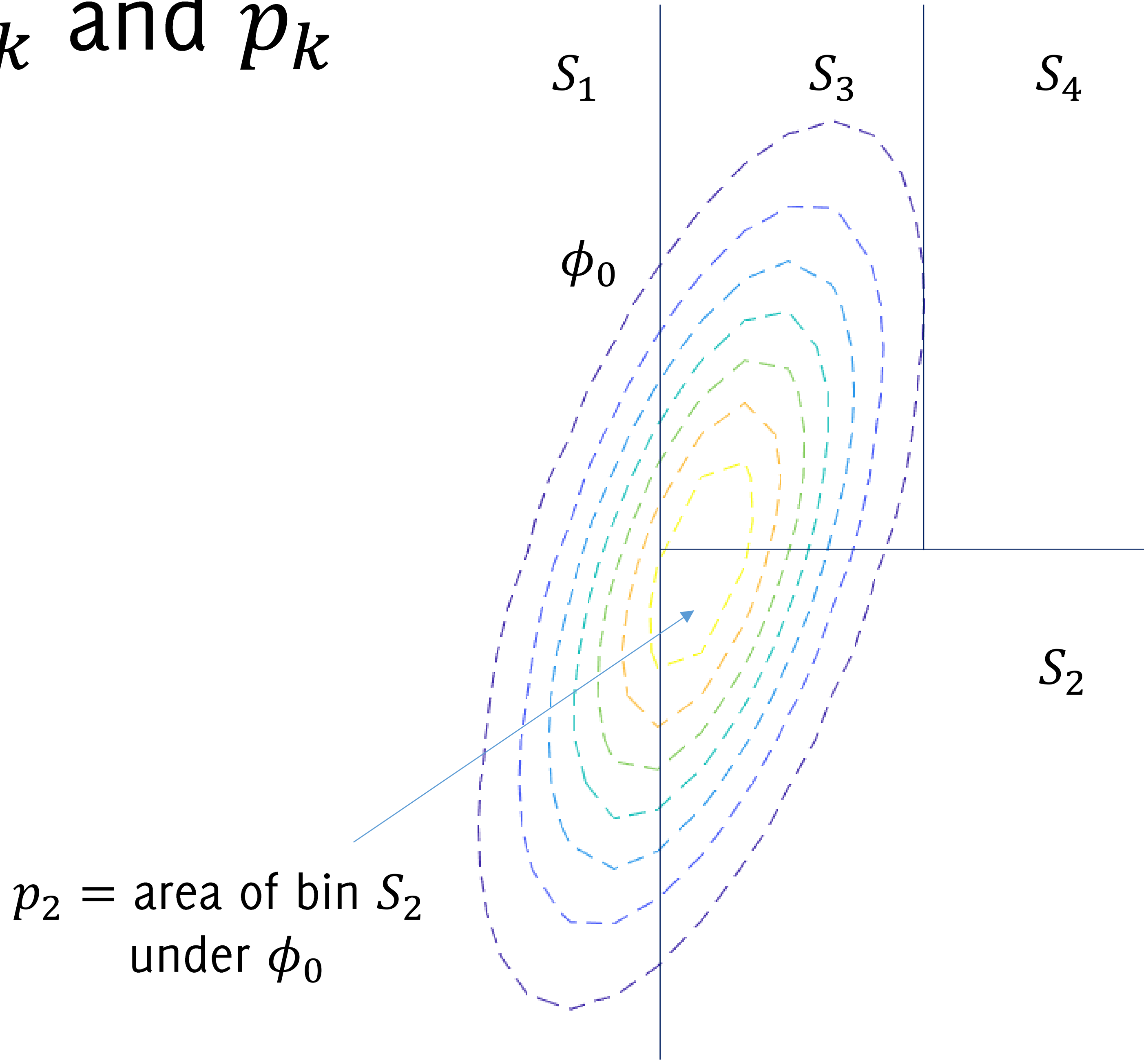


Differences between π_k and p_k

$\{p_k\}$ are the true bin probabilities. Thus, each p_k is the area of the bin S_k under the unknown ϕ_0. The true bin probabilities $\{p_k\}$ follow a Dirichlet distribution.

Given a batch W, the number of points falling in each bin $\{y_k\}$ is a realization of a multinomial distribution

$$\mathcal{M}(p_1, \ldots, p_K, \nu, K)$$



Implications

1. Draw the expected bin probabilities $(p_1, \dots, p_K)$ from the Dirichlet with parameters $\{\pi_k\}$
2. Draw the number of samples $(y_1, \dots, y_K)$ falling in each bin from a multinomial distribution having parameters $(p_1, \dots, p_K)$
$$(y_1, \dots, y_K) \sim \mathcal{M}(p_1, \dots, p_K, \nu, K)$$
3. Compute the values of test statistics $T_h(\cdot)$
4. Compute the threshold γ from the empirical distribution of $T_h(\cdot)$

MonteCarlo procedure to compute threshold γ without generating batches of data under ψ_0, without even constructing the QuantTree!

L. Frittoli, D. Carrera, G. Boracchi *"Nonparametric and Online Change Detection in Multivariate Datastreams using QuantTree"* IEEE TKDE 2022

Pros and Cons

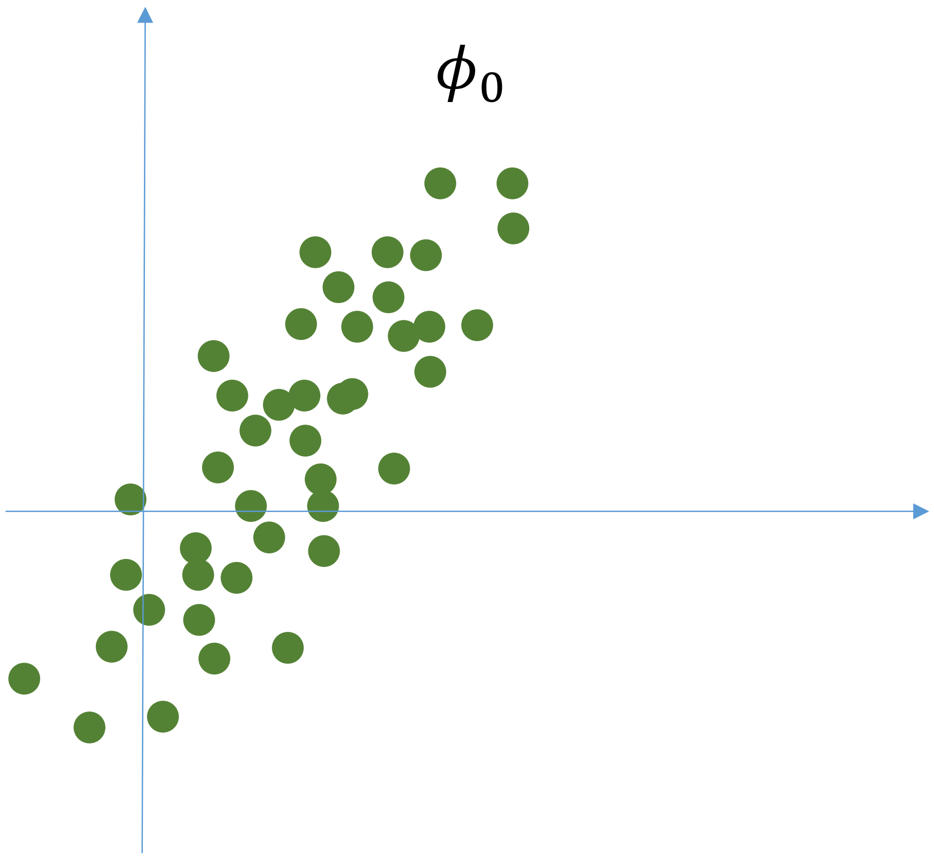
Advantages of QT

Provide a truly **multivariate** monitoring scheme that:

- Enables change detection in a **nonparametric manner** (no assumption on ϕ_0), possibly in high dimensional data d;
- **Guarantees control over the false positives** for any statistic $\mathcal{T}_h(W)$
- It requires **little training data** TR (while alternatives based on bootstrap do);
- It is rather **efficient to use**, compared to other schemes.

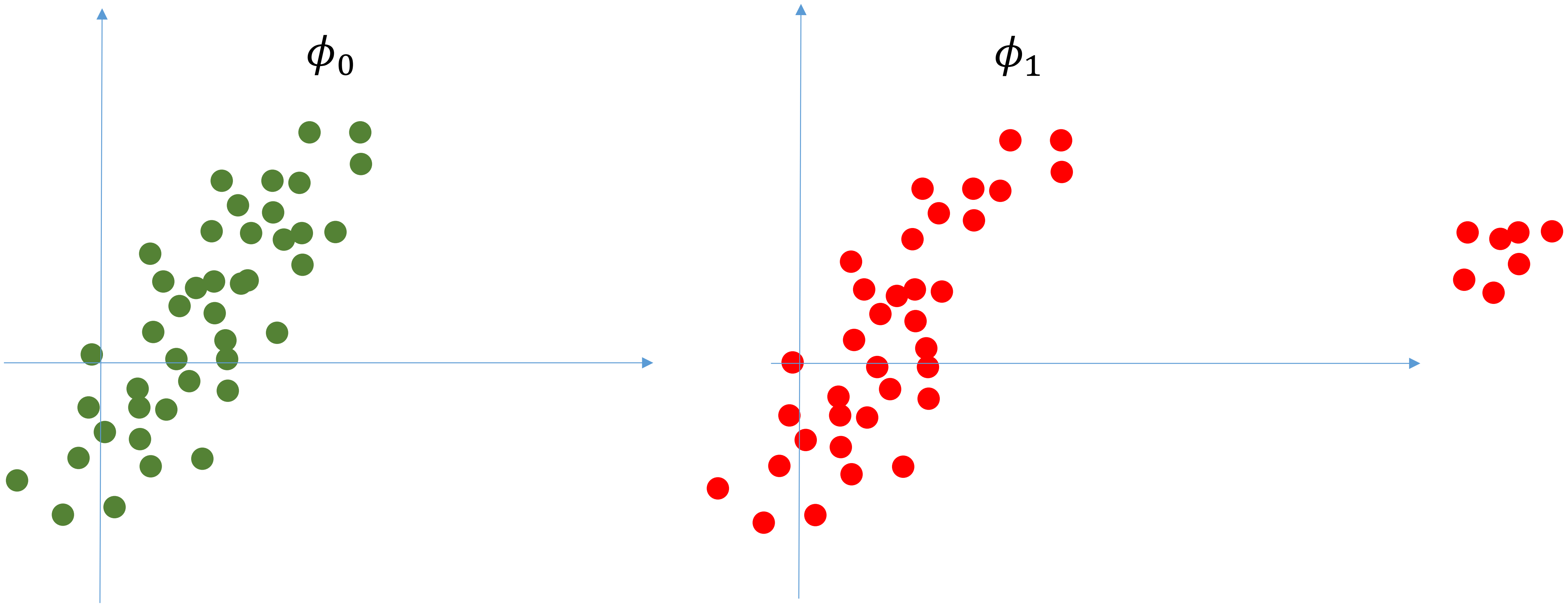
Limitations

Like any test based on histograms, QT does not perceive distribution changes "within" a bin.

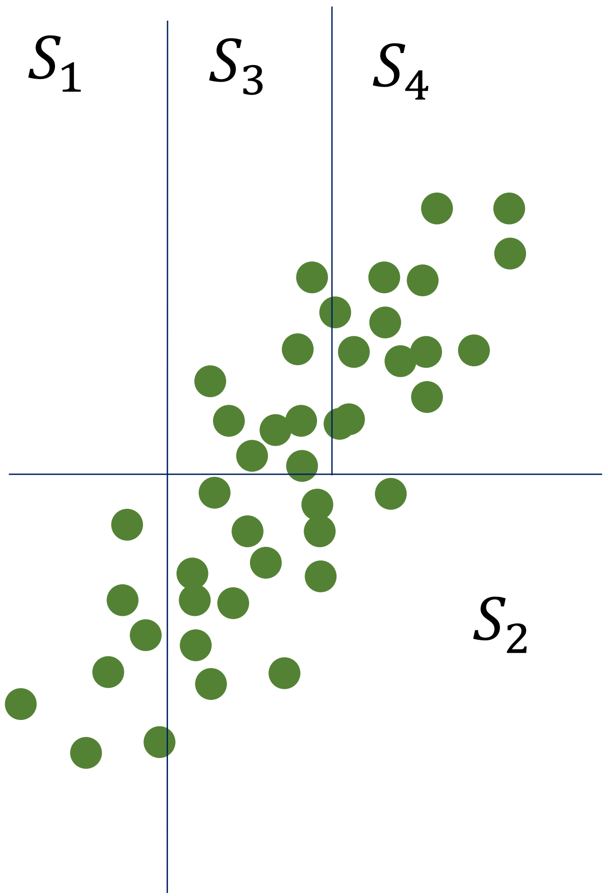


Limitations

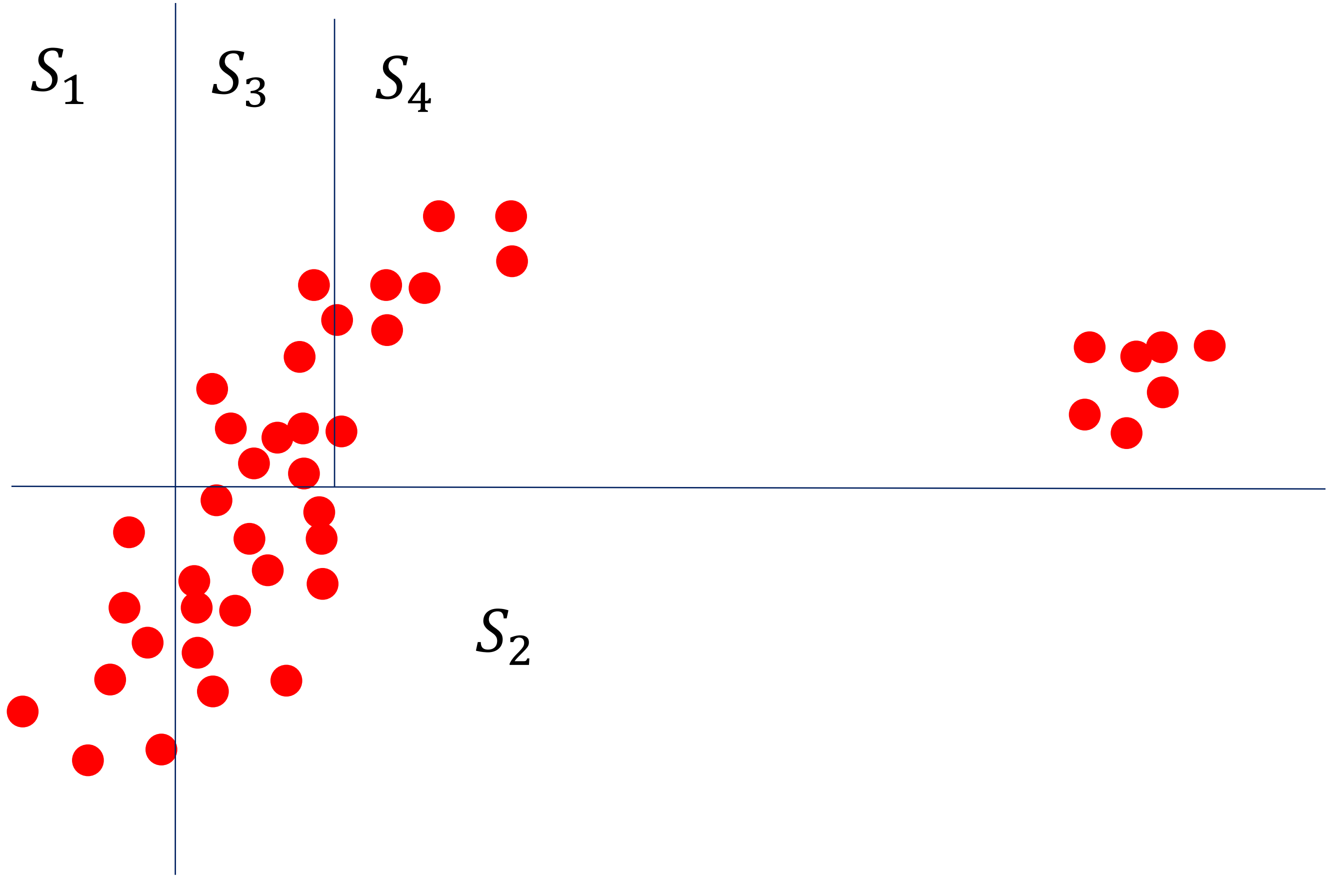
Depending on the bin partitioning, this apparent distribution change cannot be perceived by QuantTree!



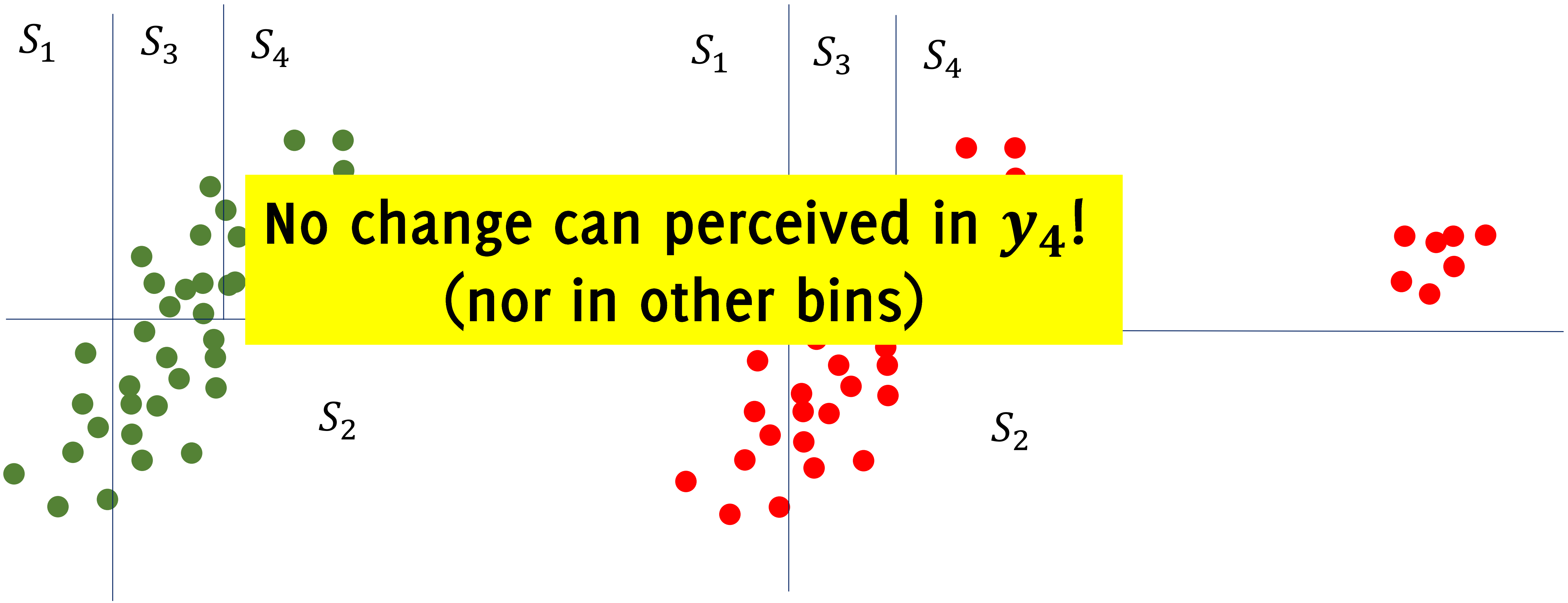
QuanTree Monitoring



QuanTree Monitoring



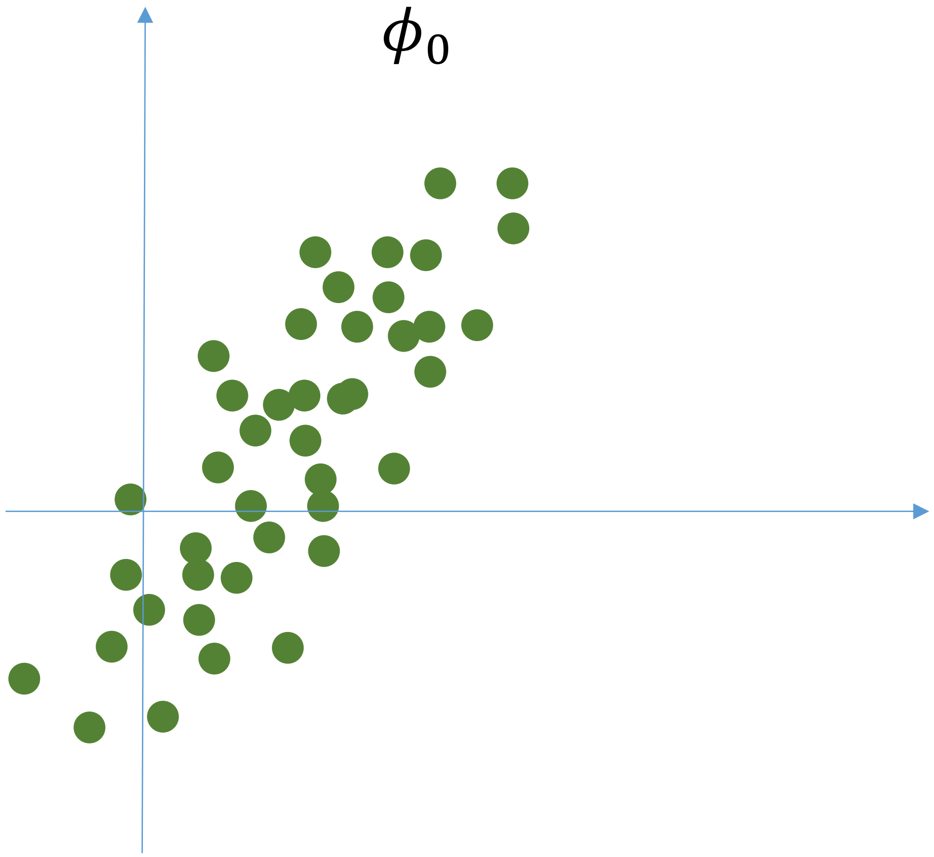
QuanTree Monitoring



No change can perceived in y_4!
(nor in other bins)

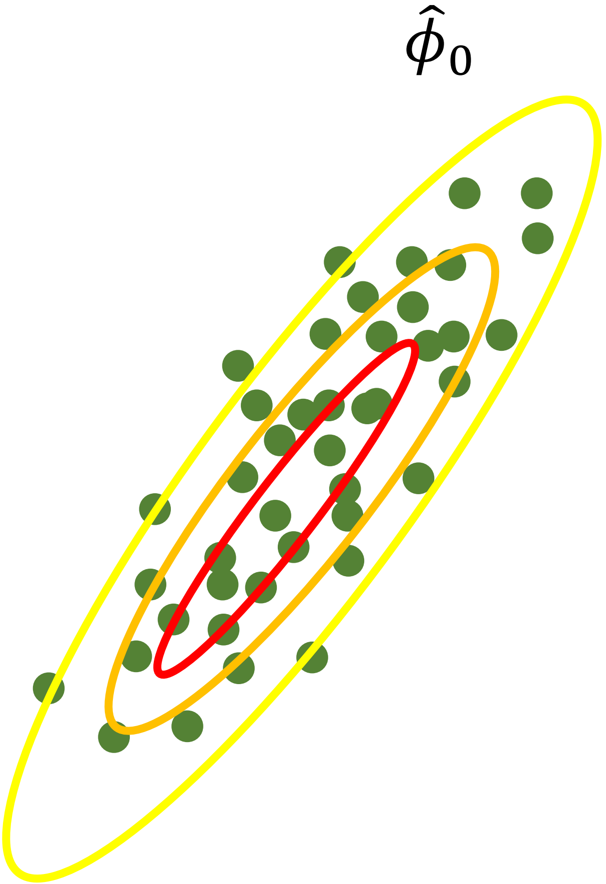
Likelihood- based monitoring

When we have an estimate of the "type" of ϕ_0, likelihood-based statistics are more powerful.



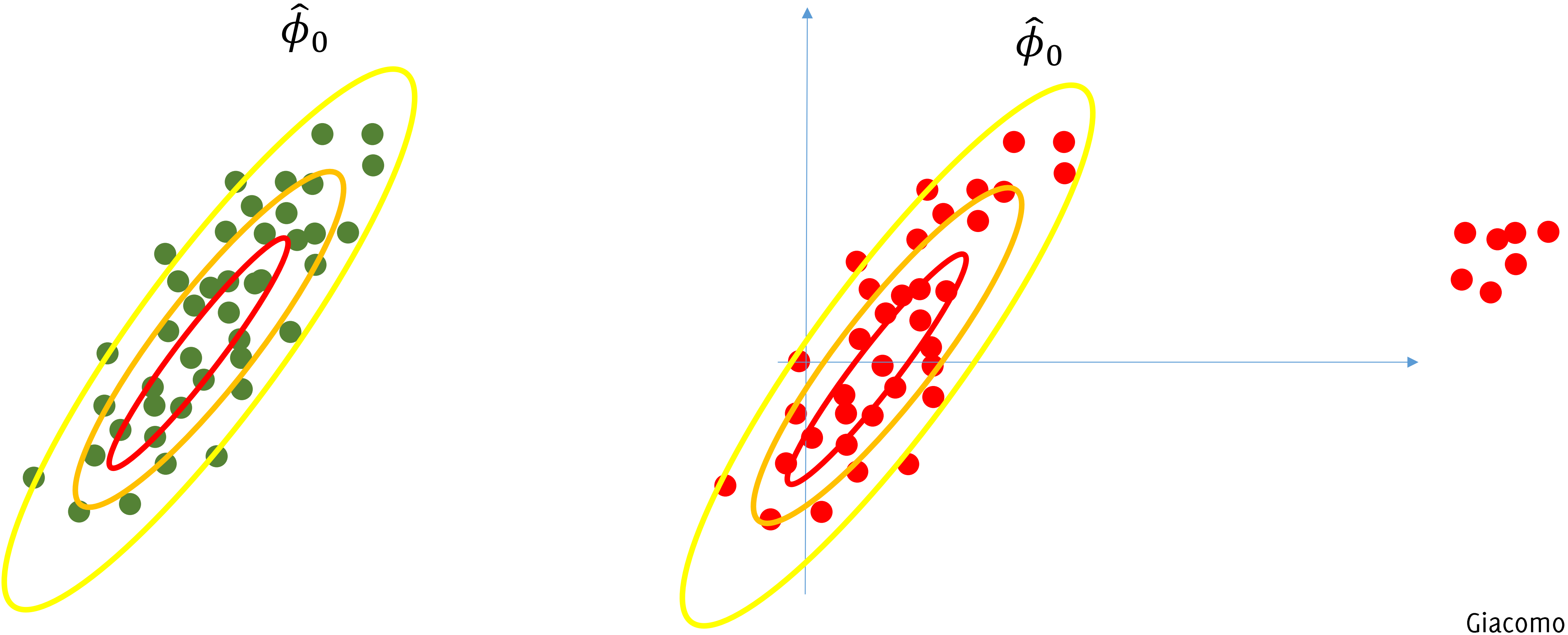
Likelihood- based monitoring

Fit $\hat{\phi}_0$ on stationary data



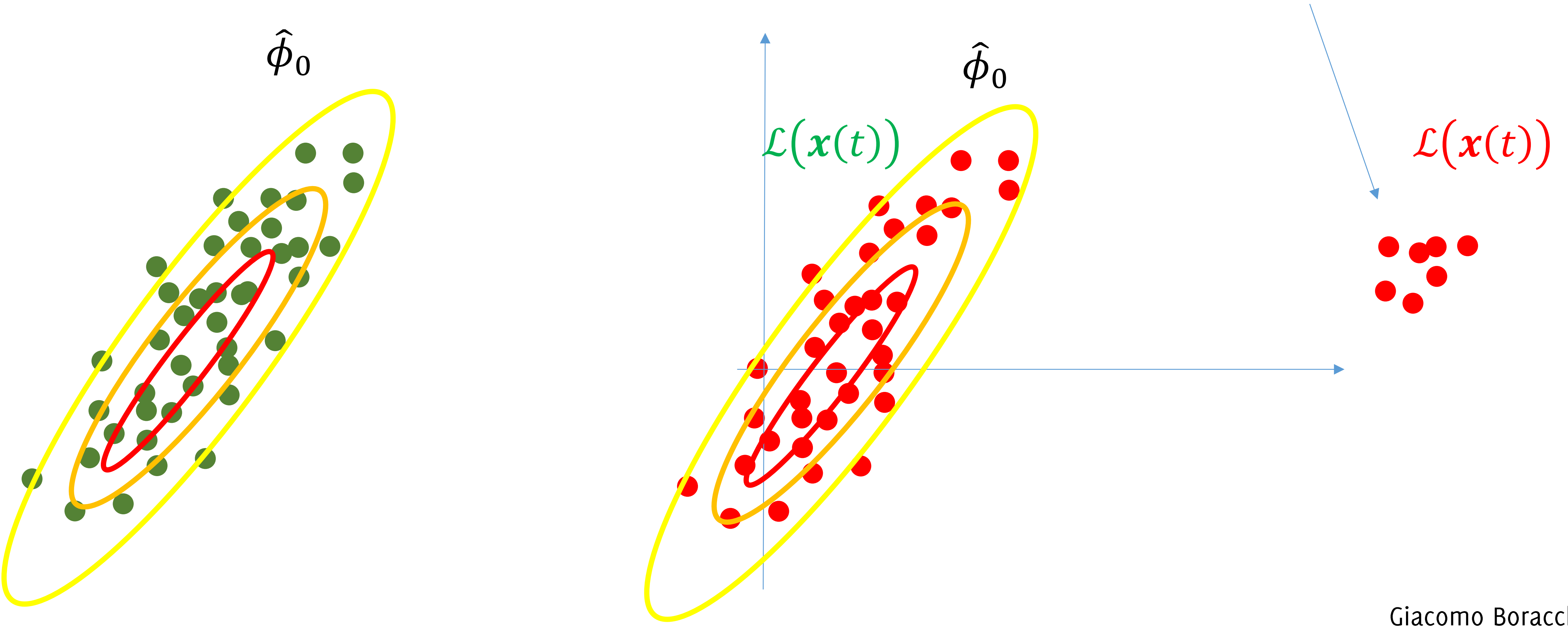
Likelihood- based monitoring

Compute $\mathcal{L}(\boldsymbol{x}(t)) = \log(\hat{\phi}_0(\boldsymbol{x}(t))$ on test data



Likelihood- based monitoring

These samples are very unusual w.r.t. $\hat{\phi}_0$
$\hat{\phi}_0(\boldsymbol{x})$ would be very low!



Limitations

- Like any test based on histograms, QT does not perceive distribution changes "within" a bin.
- Poor in efficiency compared to other tree structures (e.g., kdTrees that are balanced)
- Just an Hyoothesis Testing: it does not perform sequential monitoring

Limitations

- Like any test based on histograms, QT does not perceive distribution changes "within" a bin.
- Poor in efficiency compared to other tree structures (e.g., kdTrees that are balanced)
- Just an Hyoothesis Testing: it does not perform sequential monitoring

This can be fixed!

QT-Exponential Weighted Moving Average (QT-EWMA)

Sequential Monitoring by QuantTrees

L. Frittoli, D. Carrera, G. Boracchi *"Nonparametric and Online Change Detection in Multivariate Datastreams using QuantTree"* IEEE TKDE 2022

8328 IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 35, NO. 8, AUGUST 2023

Nonparametric and Online Change Detection in Multivariate Datastreams Using QuantTree

Luca Frittoli, Diego Carrera, and Giacomo Boracchi

Sequential Monitoring Settings

Online monitoring:

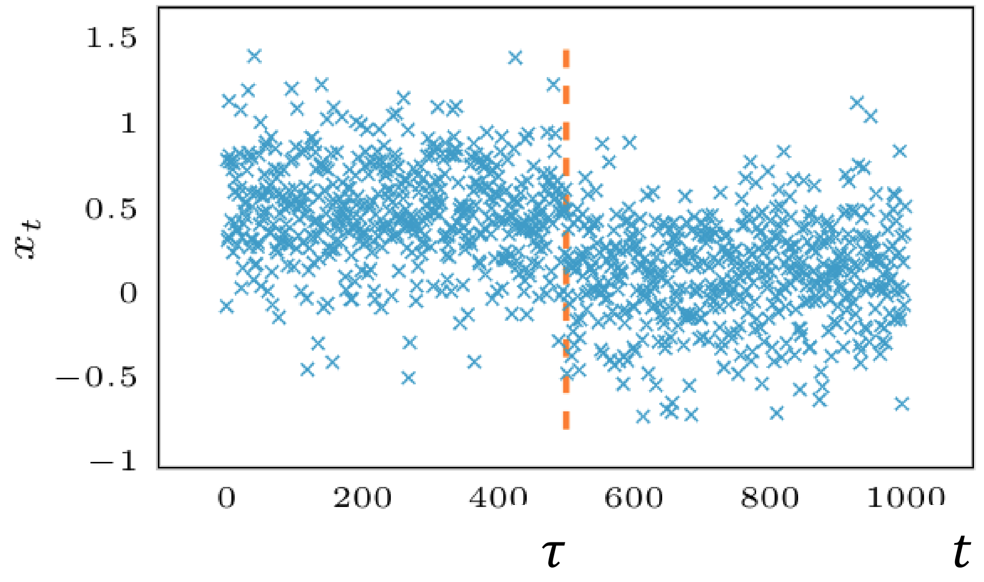
- At time t, a new sample $\boldsymbol{x}(t)$ arrive and a decision must be made

Sequential Monitoring Settings

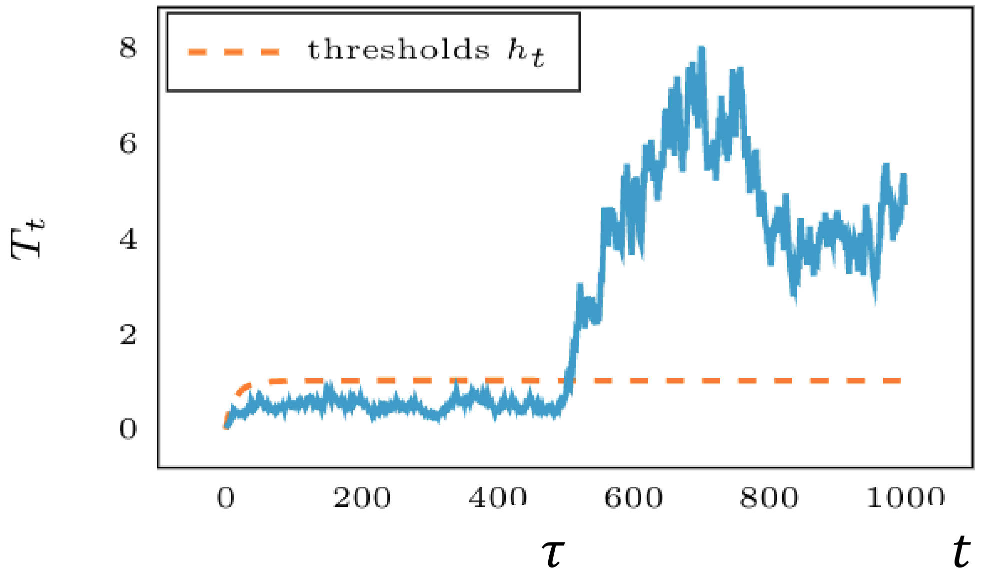
Online monitoring:

- At time t, a new sample $\boldsymbol{x}(t)$ arrive and a decision must be made
- After $\phi_0 \rightarrow \phi_1$, the evidence for a change increases and the test is expected to be more powerful

ϕ_0 ϕ_1



"nice" (sequential) test statistic



Sequential Monitoring Settings

Online monitoring:

- At time t, a new sample $\boldsymbol{x}(t)$ arrive and a decision must be made
- After $\phi_0 \to \phi_1$, the evidence for a change increases and the test is expected to be more powerful
- There is no clear notion of false alarm, rather measure **the expected time between false positive**, Average Run Length ARL_0
$$ARL_0 = \mathrm{E}_{\boldsymbol{x}}[\hat{\tau} | \boldsymbol{x} \sim \phi_0]$$
- Similarly, rather than the test power (TPR or AUC), measure **the expected detection delay**
$$ARL_1 = \mathrm{E}_{\boldsymbol{x}}[\hat{\tau} | \boldsymbol{x} \sim \phi_1]$$

Sequential Monitoring Challenges

Computational Challenges:

- Each decision should be made in constant time
- Impossible to store previously observed data as a reference

Theoretical Challenges:

- Difficult to define sequential statistics with multivariate data
- Difficult to define, for a target value of ARL_0, the corresponding threshold $\gamma = \gamma(ARL_0)$ which do not depend on ϕ_0
- Bootstrap is often not a viable alternative since we need to **consider temporal evolution** of the analysis

EWMA: Exponential Weighted Moving Average

EWMA is a standard sequential monitoring scheme for **1D** datastreams

We take inspiration from **ECDD for concept-drift** monitoring

$$Z_0 = 0, \qquad Z_t = (1-\lambda) Z_{t-1} + \lambda\, e_t$$

- $e_t \in \{0,1\}$ is the **classification error** of a classifier at time t
- $\lambda \in [0,1]$ is a parameter regulating test “reactiveness”

As a matter of fact

- Z_t is in stationary conditions tends to the average classification error
- After a change, Z_t moves towards the post-change classification error

G. J. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand *"Exponentially Weighted Moving Average Charts for Detecting Concept Drift"* Pattern Recogn. Lett. 33, 2 (Jan. 2012), 191–198 2012

EWMA: Detection Scheme

In ECDD it is possible to set a detection rule controlling ARL_0

$$Z_t > p_0 + L_t \sigma_{Z_t}$$

Defining the sequence $\{L_t\}_t$ is very complicated as these depend on $\hat{p}_{0,t}$ (the estimated classification error).

A «simple» problem to address via MonteCarlo simulation is, given a value L and p_0, to estimate the corresponding ARL_0

$$Montecarlo(L, p_0) \rightarrow ARL_0$$

It is also possible «to revert» this by setting up a suitable Montecarlo scheme such that, provided ARL_0 and p_0 one estimates L

This holds true because e_t follows a Bernoulli distribution

G. J. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand *"Exponentially Weighted Moving Average Charts for Detecting Concept Drift"* Pattern Recogn. Lett. 33, 2 (Jan. 2012), 191–198 2012

Sequential Monitoring by QT: Idea

Using EWMA over QuantTree bins:

- Replace e_t by statistics derived from an indicator functions defined on each single bin (this is also a binary quantity).

$$y_{k,t} = \mathbb{I}(x_t \in S_k) = \begin{cases} 0 & x_t \notin S_k \\ 1 & x_t \in S_k \end{cases}$$

note that $E_{\phi_0}[y_{k,t}] = p_k$ (the probability for a sample to fall in S_k)

L. Frittoli, D. Carrera, G. Boracchi *"Nonparametric and Online Change Detection in Multivariate Datastreams using QuantTree"* IEEE TKDE 2022

Sequential Monitoring by QT: Idea

Using EWMA over QuantTree bins:

- Replace e_t by statistics derived from an indicator functions defined on each single bin (this is also a binary quantity).
- **Compute a «*bin-wise*» EWMA statistic** corresponding proportion of samples falling in each bin. This is exactly the same as the classification error

$$Z_{k,0} = 0, \qquad Z_{k,t} = (1-\lambda)Z_{k,t-1} + \lambda\, y_{k,t} \quad \forall k = 1, \dots, K$$

Sequential Monitoring by QT: Idea

Using EWMA over QuantTree bins:

- Replace e_t by statistics derived from an indicator functions defined on each single bin (this is also a binary quantity).
- **Compute a *«bin-wise»* EWMA statistic** corresponding proportion of samples falling in each bin. This is exactly the same as the classification error
- **Aggregate all the EWMA statistics** in a *Pearson-like* statistic

$$\mathcal{T}_t = \sum_{k=1}^{K} \frac{\left(Z_{k,t} - \hat{\pi}_k\right)^2}{\hat{\pi}_k}$$

Which is the Pearson Statistics monitoring how much the bin-wise EWMA departs from $\hat{\pi}_k$

L. Frittoli, D. Carrera, G. Boracchi *"Nonparametric and Online Change Detection in Multivariate Datastreams using QuantTree"* IEEE TKDE 2022

Sequential Monitoring by QT: Idea

Using EWMA over QuantTree bins:

- Replace e_t by statistics derived from an indicator functions defined on each single bin (this is also a binary quantity).
- **Compute a «*bin-wise*» EWMA statistic** corresponding proportion of samples falling in each bin. This is exactly the same as the classification error
- **Aggregate all the EWMA statistics** in a *Pearson-like* statistic
- Compute a sequence of detection thresholds $\{\gamma_t\}$ **via the MonteCarlo procedure** described [Ross 2012], **but leveraging QT properties** to speed up simulations

L. Frittoli, D. Carrera, G. Boracchi *"Nonparametric and Online Change Detection in Multivariate Datastreams using QuantTree"* IEEE TKDE 2022

The QT-EWMA Algorithm

Algorithm 1: QT-EWMA

input : datastream $x_1, x_2, \ldots$, target $\{\pi_j\}_{j=1}^K$, thresholds $\{h_t\}_t$, TR
output : detection flag `ChangeDetected`, detection time t^*

`ChangeDetected` $\leftarrow$ False, $\quad t^* \leftarrow \infty$;
estimate QT histogram $\{(S_j, \pi_j)\}_{j=1}^K$ from TR and define $\{\hat{\pi}_j\}_{j=1}^K$ as in (4);
$Z_{j,0} \leftarrow \hat{\pi}_j \; \forall j = 1, \ldots, K$;
for $t = 1, \ldots$ **do**
  $y_{j,t} \leftarrow \mathbb{1}(x_t \in S_j)$;
  $Z_{j,t} \leftarrow (1-\lambda) Z_{j,t-1} + \lambda y_{j,t}, \quad j = 1 \ldots, K$;
  $T_t \leftarrow \sum_{j=1}^K (Z_{j,t} - \hat{\pi}_j)^2 / \hat{\pi}_j$;
  if $T_t > h_t$ **then**
    `ChangeDetected` $\leftarrow$ True, $\quad t^* \leftarrow t$;
    break;
  end
end
return `ChangeDetected`, t^*

find in which bin each sample falls

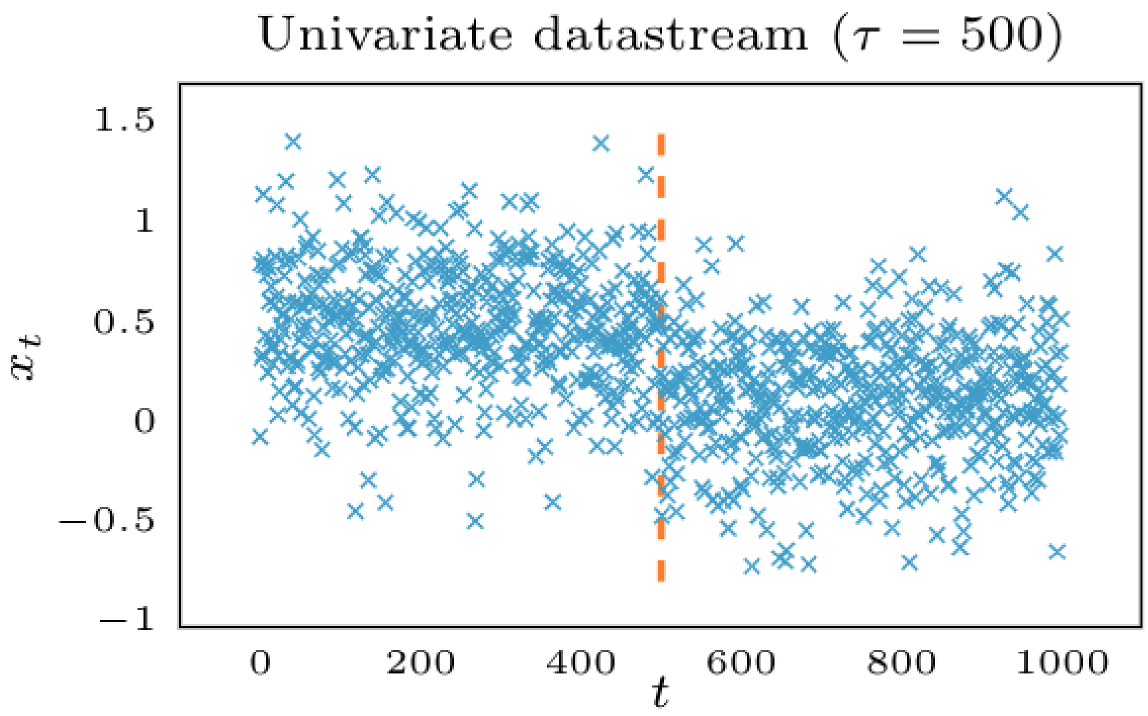
monitor the empirical bin probabilities by EWMA statistics $Z_{j,t}$

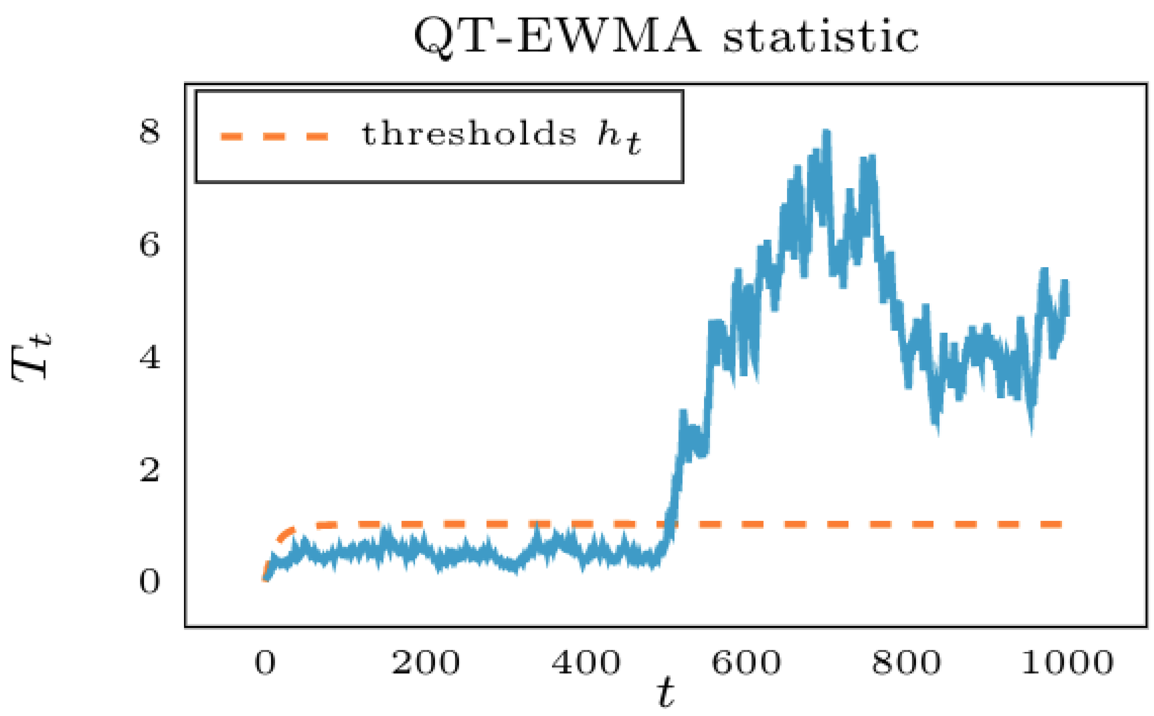
measure the deviation from the expected probabilities by $\mathcal{T}_t$

detect a change when $\mathcal{T}_t$ exceeds a threshold h_t

L. Frittoli, D. Carrera, G. Boracchi *"Nonparametric and Online Change Detection in Multivariate Datastreams using QuantTree"* IEEE TKDE 2022

Example





The **deviation** of the **bin probabilities** from their expected values measured by $\mathcal{T}_t$ **increases** after a **distribution change**

QT-EWMA: Thresholds $\{h_t\}$ computation

The theoretical properties of QuantTree **guarantee** that our statistics are **independent** from ϕ_0, and d.

Test statistics depends on N, the target ARL_0, the parameter λ and $\{\pi_k\}$

We set h_t to keep a constant probability of a false alarm at each time t

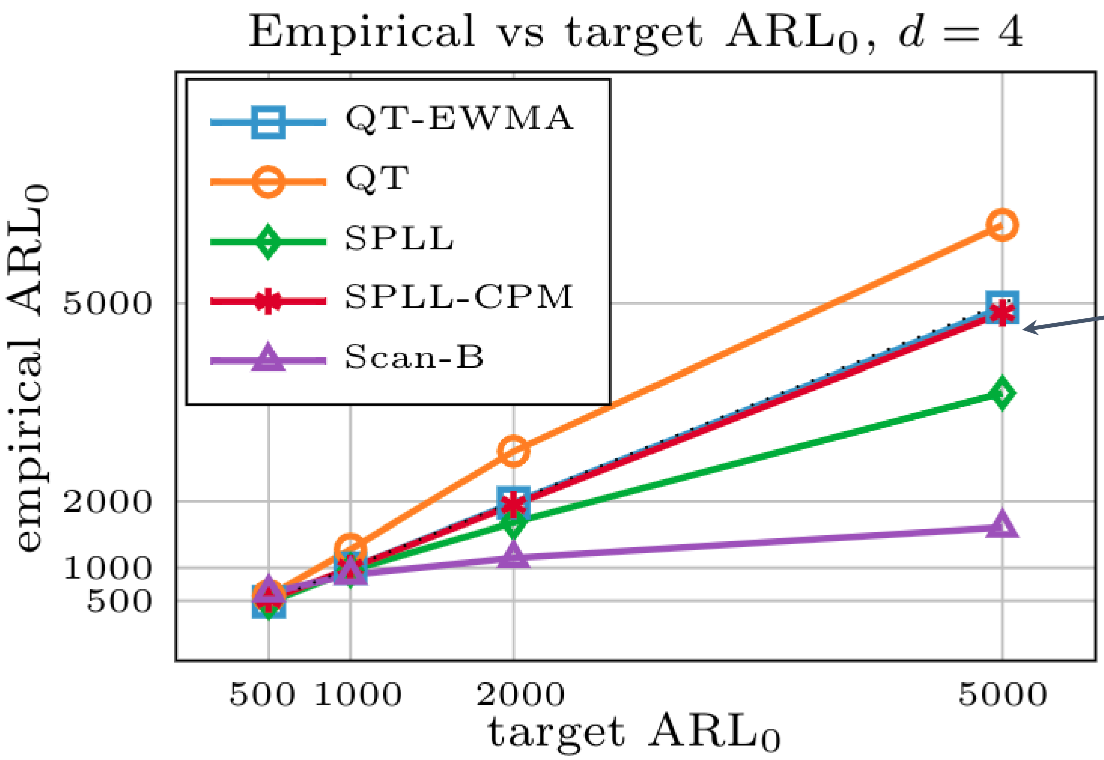
$$P(\mathcal{T}_t > \gamma_t \mid \mathcal{T}_\tau < \gamma_\tau\ , \forall \tau < t) = \alpha = \frac{1}{ARL_0}$$

We design an efficient **Monte Carlo scheme** to compute these thresholds using theoretical results from QT.

We regularize $\{\gamma_t\}$ by fitting a polynomial in t^{-1} to the empirical estimates

Experiments: synthetic Gaussian data

We set different ARL_0 values and measure the **empirical** ARL_0 of QT-EWMA and the other considered methods

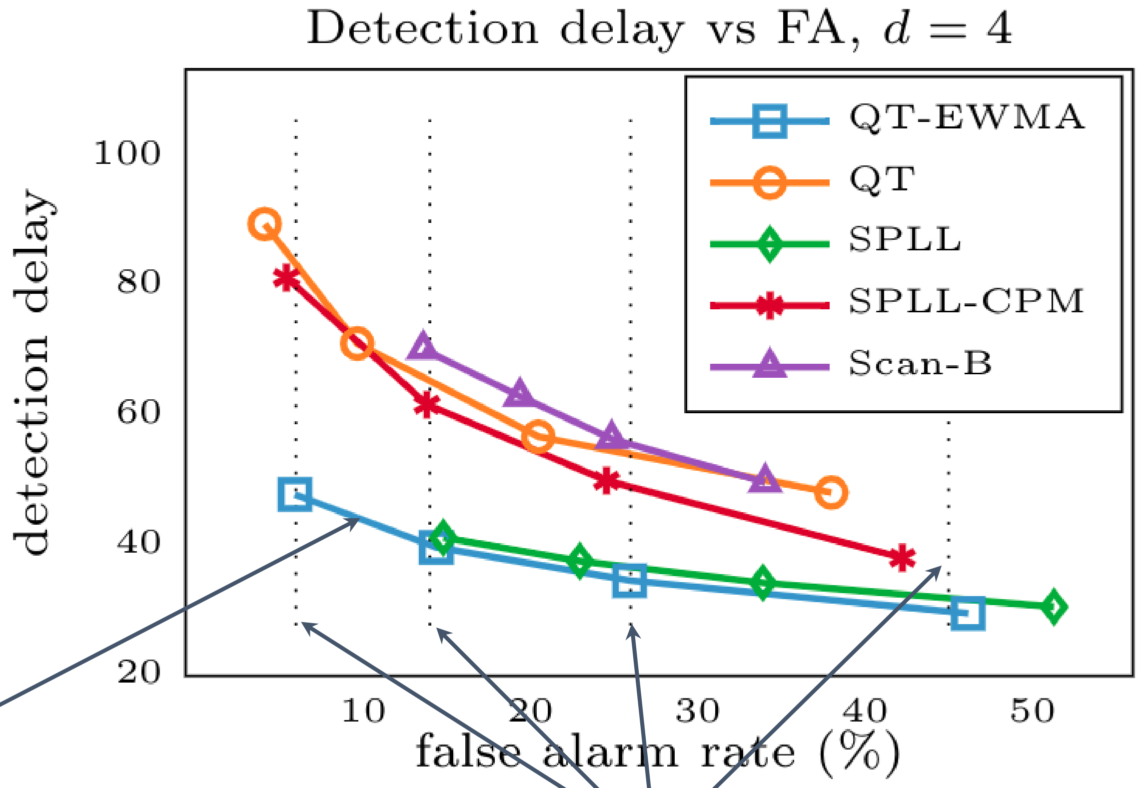


Goal: preserve the target ARL_0

[SPLL] L. Kuncheva "Change Detection in Streaming Multivariate Data Using Likelihood Detectors", IEEE TKDE, 2011
[Scan-B] S. Li et al. "M-Statistic for Kernel Change-Point Detection", Advances in Neural Information Processing Systems, 2015

Experiments: synthetic Gaussian data

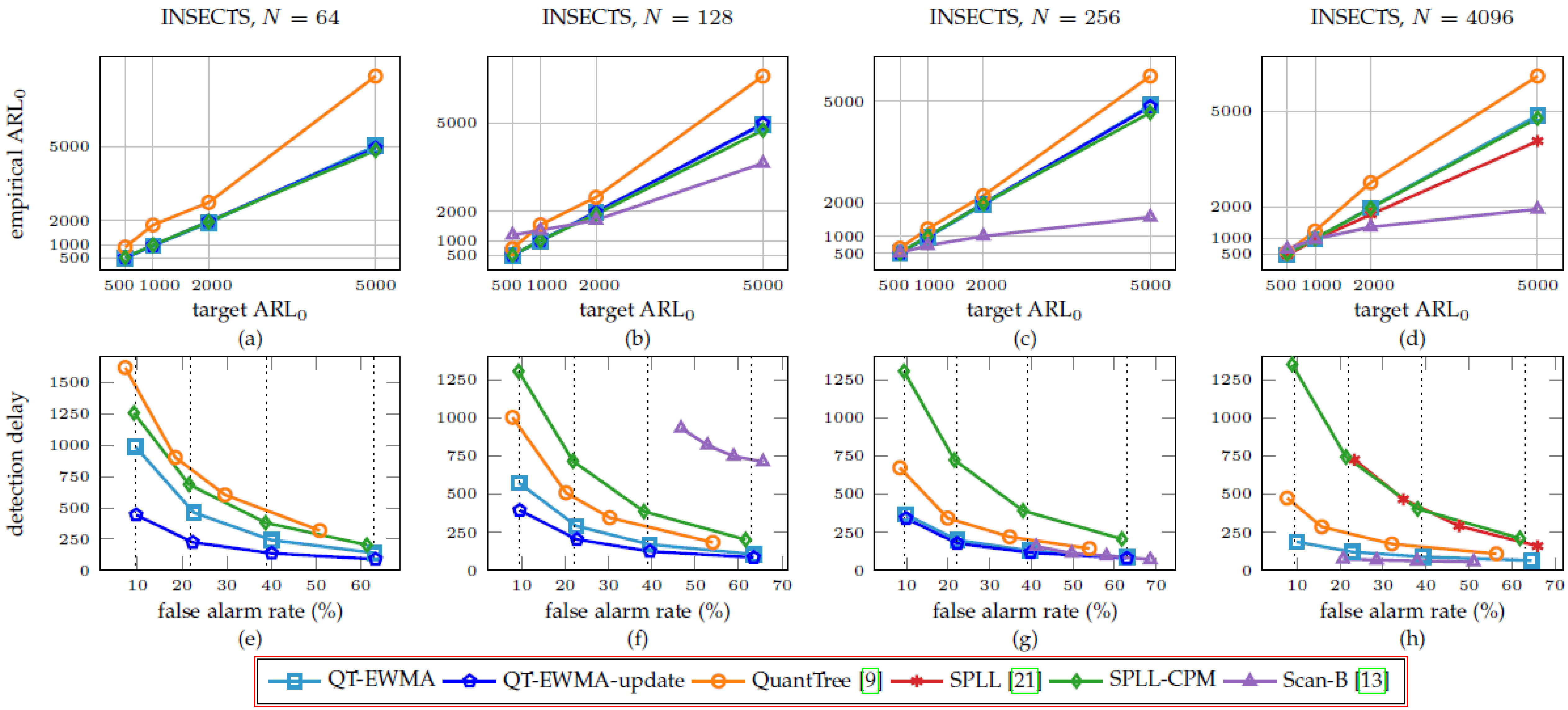
We set different ARL_0 values and observe the **trade-off** between **detection delay** and **false alarm rate**



Goal 1: minimize the detection delay

Goal 2: maintain the target false alarm rates depending on the target ARL_0

Experiments: Real data



What if we have very little training data?

QT-EWMA-update

When TR is very small, $\hat{\pi}_k$ are very far from the true probabilities, and

$$\mathcal{T}_t = \sum_{k=1}^{K} \frac{\left(Z_{k,t} - \hat{\pi}_k\right)^2}{\hat{\pi}_k}$$

Is not very powerful as a test statistic

Idea: update bin probabilities $\hat{\boldsymbol{\pi}}_{\boldsymbol{k}}$ as long as no change is detected

$$\hat{p}_{k,0} = \hat{\pi}_k, \qquad \text{and } \hat{p}_{k,t} = (1 - \omega_t)\hat{p}_{k,t-1} + \omega y_{k,t}$$

Where

$$\omega_t = \frac{1}{\beta(N + t)}$$

regulates the updating speed, and tends to 0 as t increases.

L. Frittoli, D. Carrera, G. Boracchi *"Nonparametric and Online Change Detection in Multivariate Datastreams using QuantTree"* IEEE TKDE 2022

QT-EWMA-update Updating Speed

The updating speed is regulated by β

$$\omega_t = \frac{1}{\beta(N+t)}$$

- High values or β are meant to prevent updating the bin probabilities after the change
- The updating speed β is a parameter of QT-EWMA-update.
- A sequence of detection thredsholds depending on β can be computed to grant nonparametric and sequential monitoring

L. Frittoli, D. Carrera, G. Boracchi *"Nonparametric and Online Change Detection in Multivariate Datastreams using QuantTree"* IEEE TKDE 2022

Concluding Remarks and Extensions

Concluding Remarks on QuantTree

- QuantTree is an **effective, theoretically grounded** monitoring scheme for **multivariate** datastreams.
- Our focus is to be **nonparametric** and **control "False Alarms"**.
- *Histograms are flexible, design them to yield a **comfortable monitoring**!*
- Enables new type of investigation (like class-wise distribution for change-detection).

Concluding Remarks on QuantTree

Extended Variants of QuantTrees:

- Kernel QuantTrees allow arbitrary-shaped bins, increasing detection power

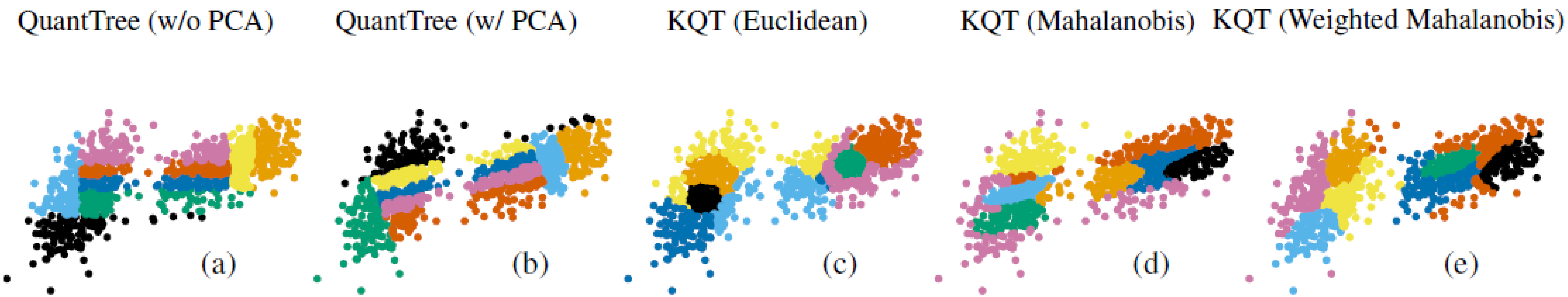


Figure 1. QuantTree generates bins as intersection of hyperplanes, performing cuts along the axis (a). After a preprocessing through PCA, the cuts are oriented along the principal directions (b). Kernel QuantTree generates bins that are subsets of d-dimensional spheres according to the underlying kernel functions, namely the Euclidean (c), Mahalanobis (d) and Weighted Mahalanobis (e) distances.

D. Stucchi, P. Rizzo, N. Folloni, G. Boracchi, "Kernel QuantTree" International Conference on Machine Learning, ICML 2023

Concluding Remarks on QuantTree

Extended Variants of QuantTrees:

- Kernel QuantTrees allow arbitrary-shaped bins, increasing detection power
- Multi-modal QuantTrees enable monitoring when ϕ_0 corresponds to a set of different distributions $\{\phi_{0,i}\}$. Batch-wise monitoring and identification of the generating modality.

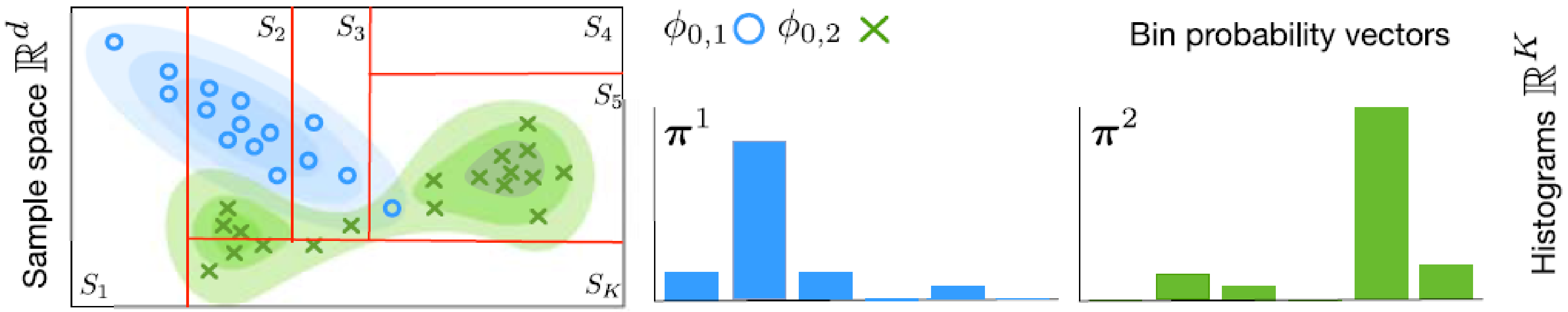
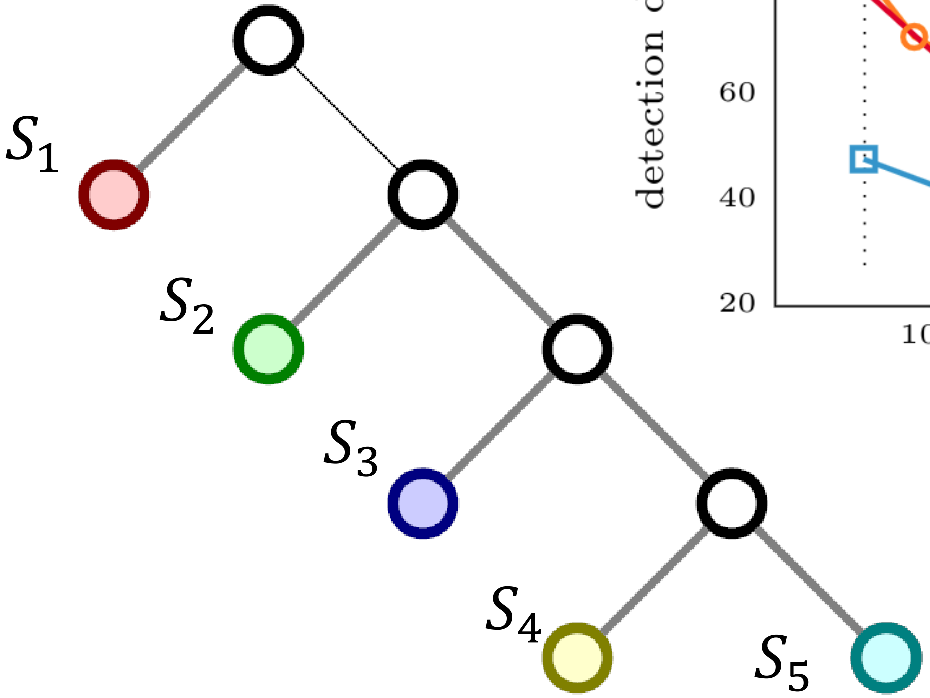
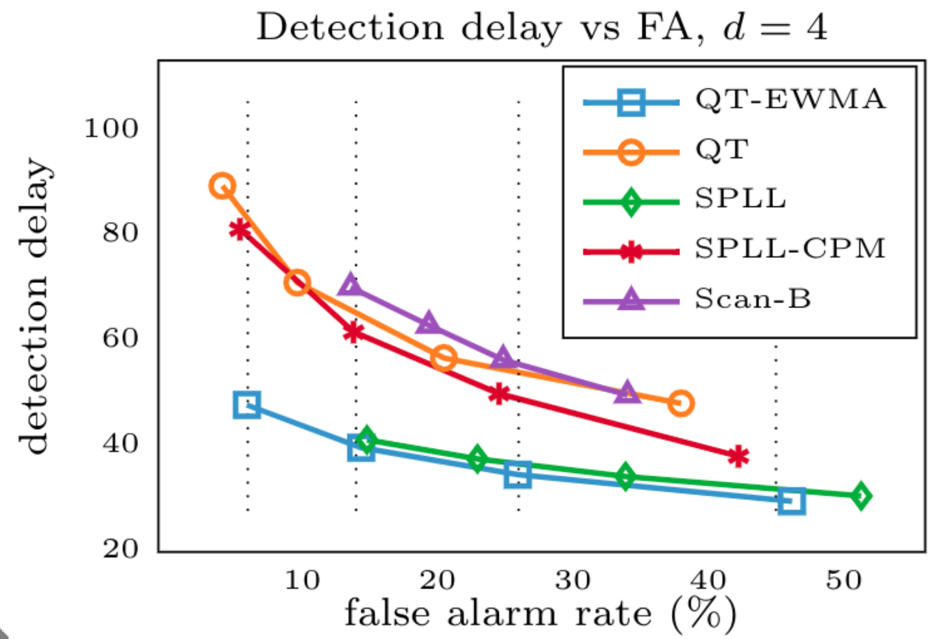


Fig. 2. Left: two stationary batches drawn from two modalities $\phi_{0,1}$ and $\phi_{0,2}$, and their contour plot. Note that here $\phi_{0,2}$ is non-Gaussian and multipeaked. A QuantTree partitioning is drawn in red lines. Right: corresponding bin-probability vectors π^1 and π^2. MMQT provides CD capabilities in a multimodal batch-wise setting, where any batch drawn from $\phi_{0,1}$ or $\phi_{0,2}$ is considered stationary.

D. Stucchi, L. Magri, D. Carrera, G. Boracchi, "Multimodal Batch-wise Change Detection" IEEE TNNLS 2023

Thank you! Questions?





https://github.com/diegocarrera89/quantTree

Repository where you can find all the resources available from download