Changepoint Detection via Subset Chains

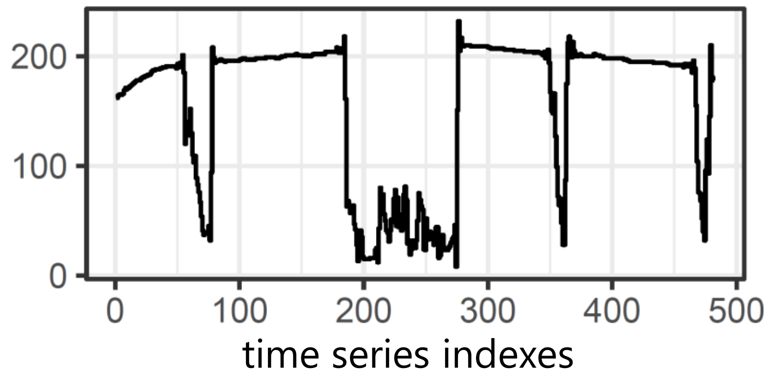
Alexis Huet, Jose Manuel Navarro, Dario Rossi

Huawei Technologies Co., Ltd.

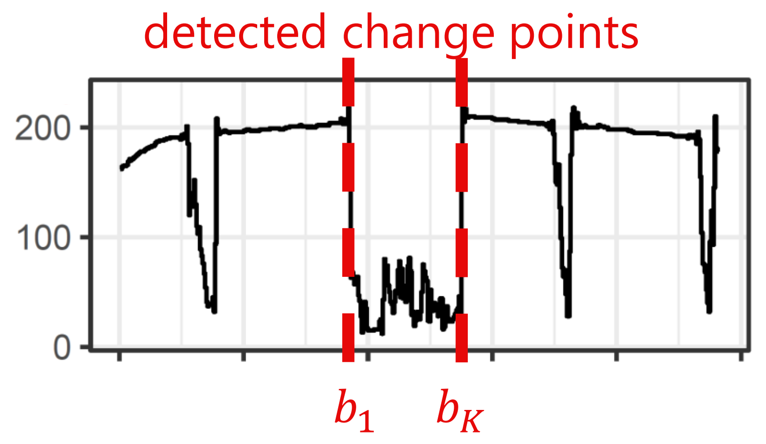
HUAWEI

Problem formulation

Given a univariate or multivariate time series, we would like to identify the change points:



some algorithm $\longrightarrow$



Segmentation $S = [1, b_1) \cup [b_1, b_K) \cup [b_K, T]$

K number of (detected) change points

Changepoint detection consists in selecting a segmentation, i.e. partitioning the indexes of a time series such that:

- each interval of the partition conserves a property of interest (e.g., the mean value),
- whereas consecutive intervals break it.

The property of interest is measured by a cost function C, mapping an interval to a numeric value.

E.g. the quadratic loss: $C_{\mathrm{L2}}(x_{[s,e)}) := \sum_{i=s}^{e-1} \|x_i - \bar{x}\|_2^2$, with $\bar{x}$ the mean (univariate of multivariate)

SOTA method – Penalty approach

Global optimization problem to find the segmentation $S = [1, b_1) \cup [b_1, b_2) \cup \cdots \cup [b_K, T]$, trading-off:

1. the total cost $\sum_{k=0}^{K} C(x_{[b_k, b_{k+1})})$ of the segmentation,
2. a penalty term increasing with number K of detected changes.

The penalty term should be selected:

– either automatically based on modeling assumptions,

– or manually set, e.g. of the form $\beta \log T$ with $\beta > 0$.

Issue 1: Each penalty term leads to a different optimization problem

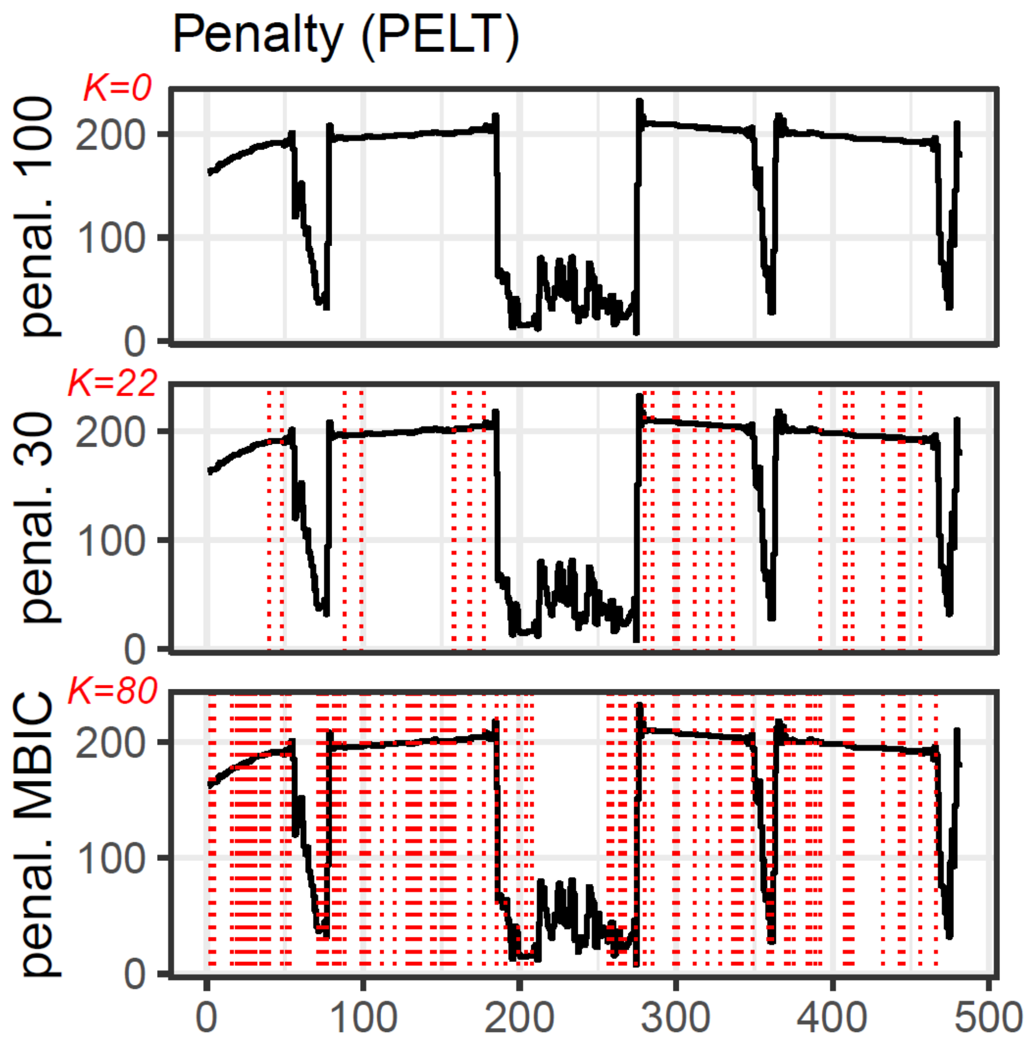
➔ Need for some flexibility

Issue 2: Difficult to interpret importance of the change

➔ Need for some interpretability

Issue 3: Except when modeling assumptions are realistic, automatic selection of penalty leads to many changes

➔ Need some good generic default parameters



SOTA method – Solution path approach

The idea of solution path approaches is to decouple:

1. the **ranking** of the potential change points and
2. the **selection** of a subset of the top ranked elements.

The idea is quite old with Binary Segmentation (BS).
Many recent algorithms explore both ranking and selection,
e.g. Wild Binary Segmentation 2 (WBS2).

Solution 1: Postpone selection at a latter stage

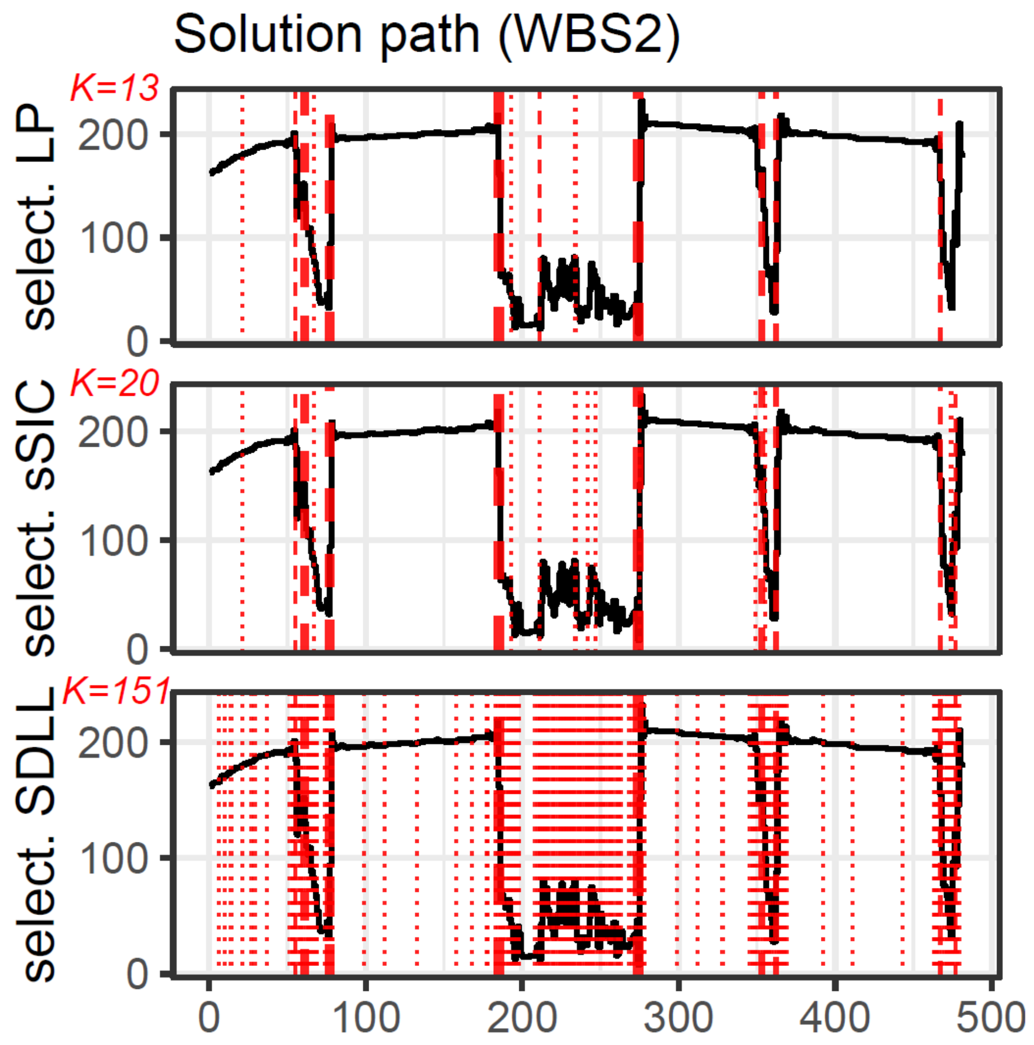
➔ Give some flexibility

Solution 2: Importance of the change given by the rank

➔ Give some interpretability

Issue 3: Automatic selection still leads to many changes

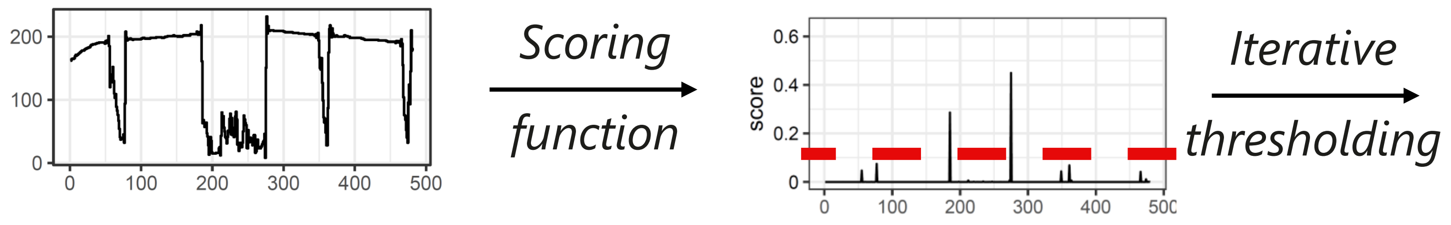
➔ Need some good generic default parameters



Our proposal: Subset chains

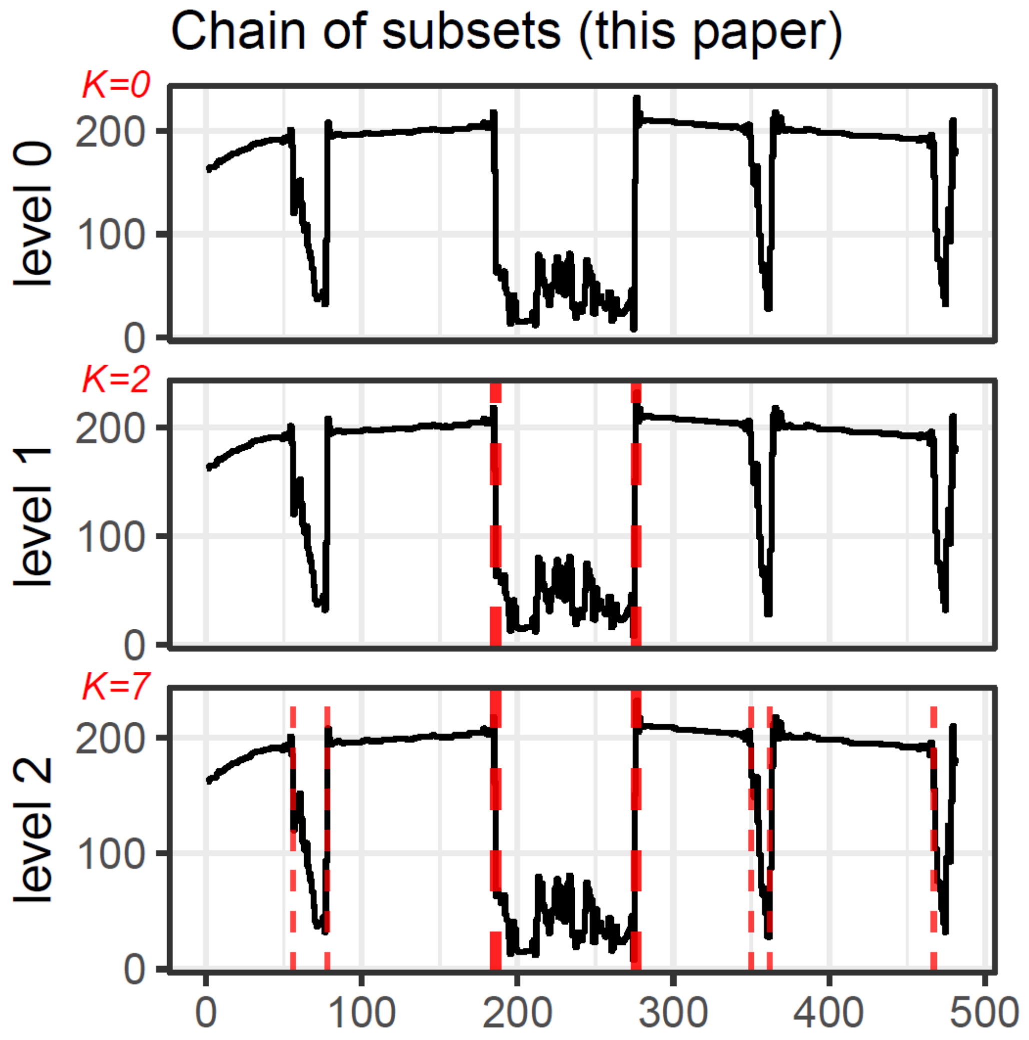
We took inspiration from the time series anomaly detection community to build our new proposal:

1. the **scoring** step gives each point a value indicating its importance as a potential change point,
2. the **thresholding** step detects changepoints iteratively, from major to minor, over multiple levels



Global problem

Local problem



Solution 1: Postpone selection at a latter stage

➔ Giving full flexibility through a single thresholding parameter

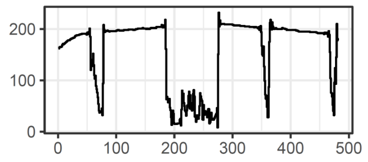
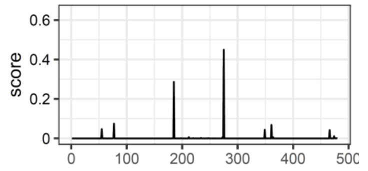
Solution 2: Importance of the change given by a **chain of subsets**

➔ Give interpretability with similar changes at the same level

Solution 3: Automatic selection of parameters give sparse detection

➔ Good default parameters robust over the tested datasets

Scoring function

Objective:  $\xrightarrow[\textit{function}]{\textit{Scoring}}$ 

Global problem Local problem

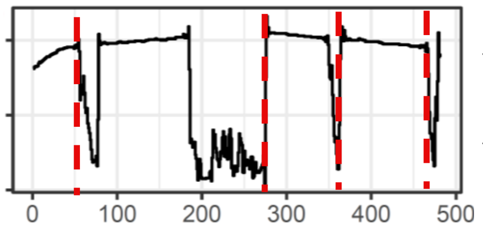
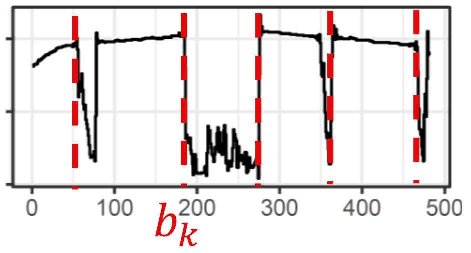
Segmentation $S = [1, b_1) \cup [b_1, b_K) \cup [b_K, T]$

K number of (detected) change points

How to change from a global to a local problem?

At the global side, it's easy to measure the cost of a segmentation S: $\sum_{k=0}^{K} C(x_{[b_k, b_{k+1})})$.

We can measure, for this segmentation S, the importance or "gain" in doing the cut b_k:

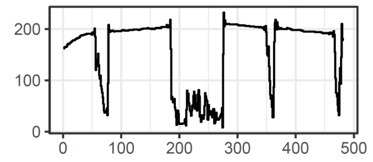
$G(b_k, S) :=$ Cost () – Cost ()

$$= C(x_{[b_{k-1}, b_{k+1})}) - \left(C(x_{[b_{k-1}, b_k)}) + C(x_{[b_k, b_{k+1})})\right).$$

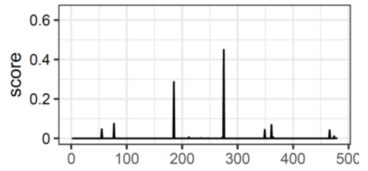
We retrieve the well-known "discrepancy function".

Scoring function

Objective:


Global problem

$\xrightarrow[\textit{function}]{\textit{Scoring}}$

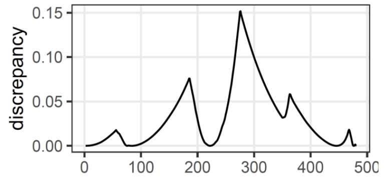


Local problem

Segmentation $S = [1, b_1) \cup [b_1, b_K) \cup [b_K, T]$

K number of (detected) change points

Direct maximization of the discrepancy function (BS) to get the scores **doesn't** have desired properties:



– incorrect magnitude of scores in that case,

– may also lead to inappropriate detected changepoints

We instead use a bottom-up approach to create the score function:

– We start from a fully segmented set,

– The score is defined as the ***maximum* gain observed so far along the constructed path**

– The worst score is removed from the segmentation at each step

Using maximum gain instead of gain allows to get two properties:

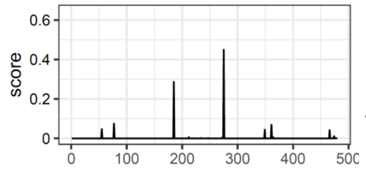
(i) the scores are always decreasing w.r.t. the solution path, and

(ii) given an existing path produced by the scores, it is possible to reconstruct the exact same scores by simply following the path

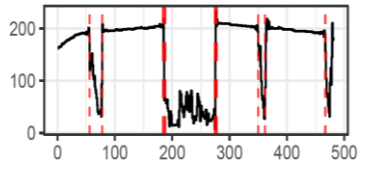
Computations are kept linear with size of the time series

Thresholding function

Objective:



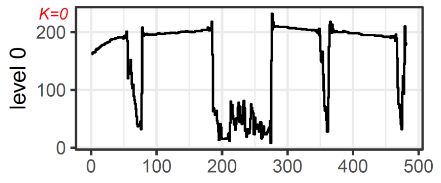
$\xrightarrow[\textit{thresholding}]{\textit{Iterative}}$



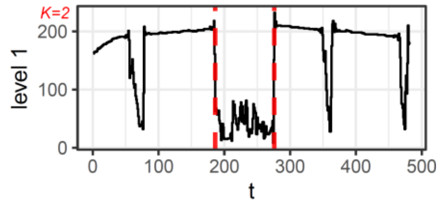
Create changepoints iteratively over multiple levels, from major to minor changes.

Level 0: no change detected

Level 1: simple thresholding by selecting a single value, e.g. 0.1, as a threshold.



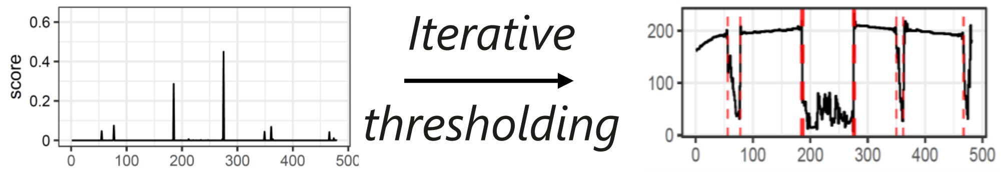
$\longrightarrow$



- By construction of the scores, decreasing the threshold can only increase the set of detected changes.
- The threshold value can be tuned without any increased computational cost (scoring already computed).
- Interpretation in terms of gain: a change point is selected whenever there exists a step (in the solution path) that has a gain greater than this threshold.

Thresholding function

Objective:

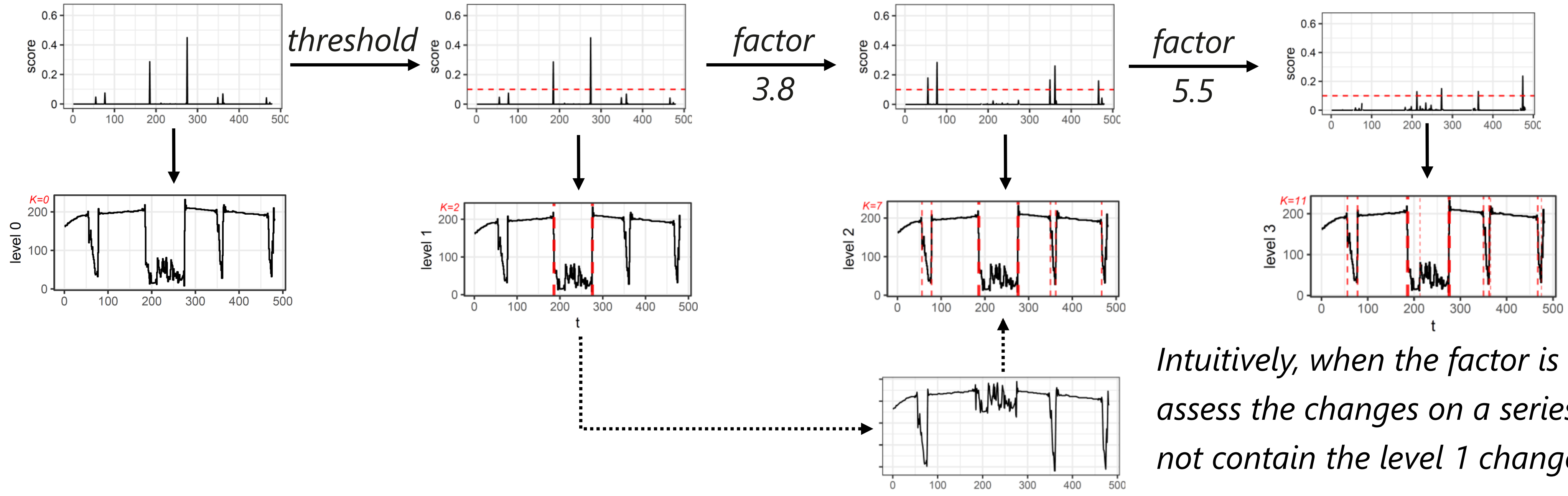


After level 1, we know that globally there is no gain greater than 0.1.

Level 2: For detecting the lower granularity changes, we apply a factor on the score function

The factor is the total cost (i.e. cost at level 0) divided by the cost of the segmentation retrieved at level 1.

Finally, we apply again the same threshold of 0.1 on those updated scores

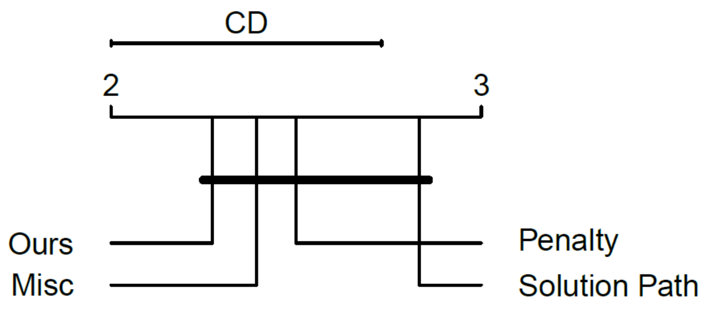


Intuitively, when the factor is applied, we assess the changes on a series that does not contain the level 1 changes

Evaluation

- *Datasets*: Publicly available Van den Burg's benchmark with 42 classic uni- and multivariate datasets
- *Labels*: The benchmark includes labels from 5 trained humans per dataset
- *Performance metric*: F1-score computed with a margin of error of $M = 5$
- *Algorithms and grid search*:
  - Our proposal with 2 cost functions and thresholds from 0.03 to 1,
  - Classic penalty approaches PELT, non-parametric, anomaly resilient RFPOP, …
  - Solution path approaches with WBS, WBS2, ChangeForest, …
- *Evaluation settings*:
  - **Oracle setting: the best parameters are selected for each dataset,**
  - Best-single setting: the best parameters are selected once for all the datasets

Family(#)	F1
Ours (48)	**0.87±0.20**
Misc (504)	*0.84±0.23*
Penalty (425)	0.81±0.23
Solution Path (113)	0.77±0.26
Zero	0.17±0.38

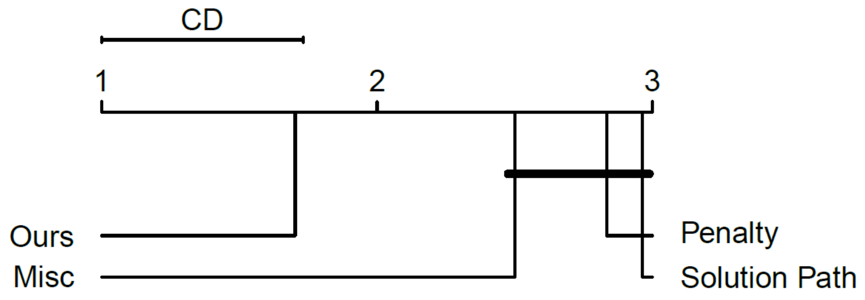


(a) Oracle setting

Evaluation

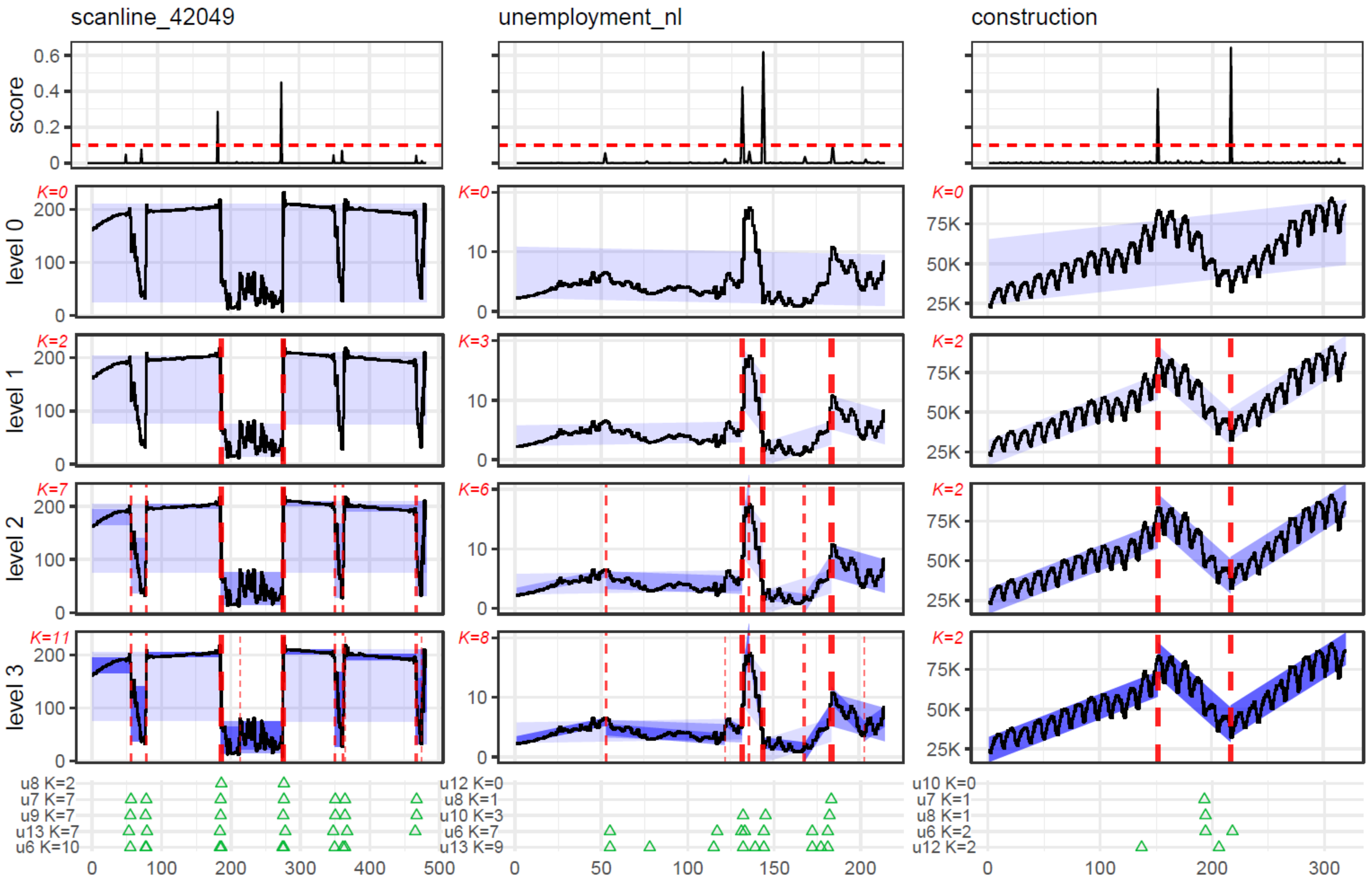
- *Datasets*: Publicly available Van den Burg's benchmark with 42 classic uni- and multivariate datasets
- *Labels*: The benchmark includes labels from 5 trained humans per dataset
- *Performance metric*: F1-score computed with a margin of error of $M = 5$
- *Algorithms and grid search*:
  - Our proposal with 2 cost functions and thresholds from 0.03 to 1,
  - Classic penalty approaches PELT, non-parametric, anomaly resilient RFPOP, …
  - Solution path approaches with WBS, WBS2, ChangeForest…
- *Evaluation settings*:
  - Oracle setting: the best parameters are selected for each dataset,
  - **Best-single setting: the best parameters are selected once for all the datasets**

Family(#)	F1	selected parameters
Ours (48)	**0.76±0.27**	linear cost; threshold=0.1
Misc (504)	*0.49±0.38*	OCP; $\lambda=100$; $a=10$; $b=k=1$
Penalty (425)	0.40±0.38	RFPOP; Huber cost; penalty=10
Solution Path (113)	0.43±0.35	NOT with M=10; quadratic cost; IC
Zero	0.17±0.38	none



(b) Best-single setting

Qualitative results with the single-best parametrization



Conclusion

Detect changes in two steps with **scoring and thresholding** independent functions:

➔ Giving full flexibility in refining the detection strategy

Provide leveled importance of the change given by a **chain of subsets**

➔ Give qualitative interpretability, with similar changes detected at the same level

A single default parametrization give good and sparse detection over the tested datasets

➔ New thresholding mechanisms can be explored

Code available: https://github.com/ahstat/scoth-segmentation

Demo available: https://huet.shinyapps.io/scoth-segmentation

Thank you

Detect changes in two steps with **scoring and thresholding** independent functions:

➔ Giving full flexibility in refining the detection strategy

Provide leveled importance of the change given by a **chain of subsets**

➔ Give qualitative interpretability, with similar changes detected at the same level

A single default parametrization give good and sparse detection over the tested datasets

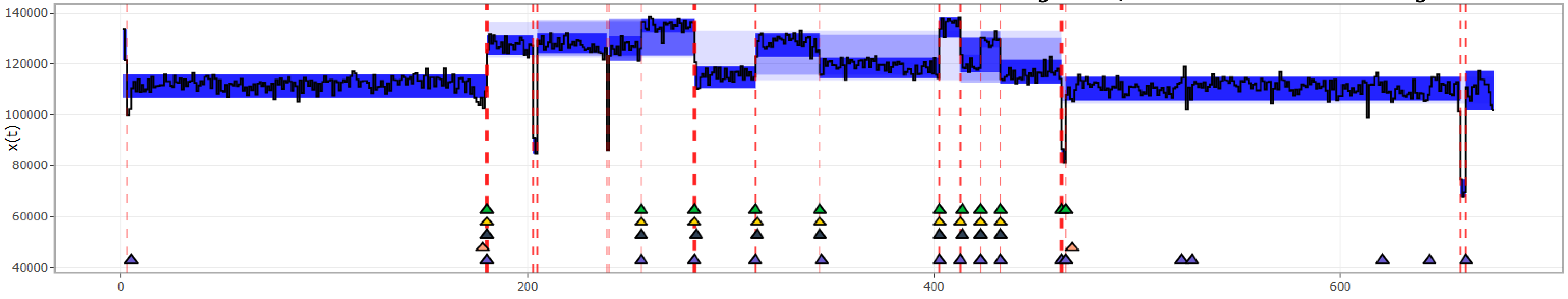
➔ New thresholding mechanisms can be explored

Code available: https://github.com/ahstat/scoth-segmentation

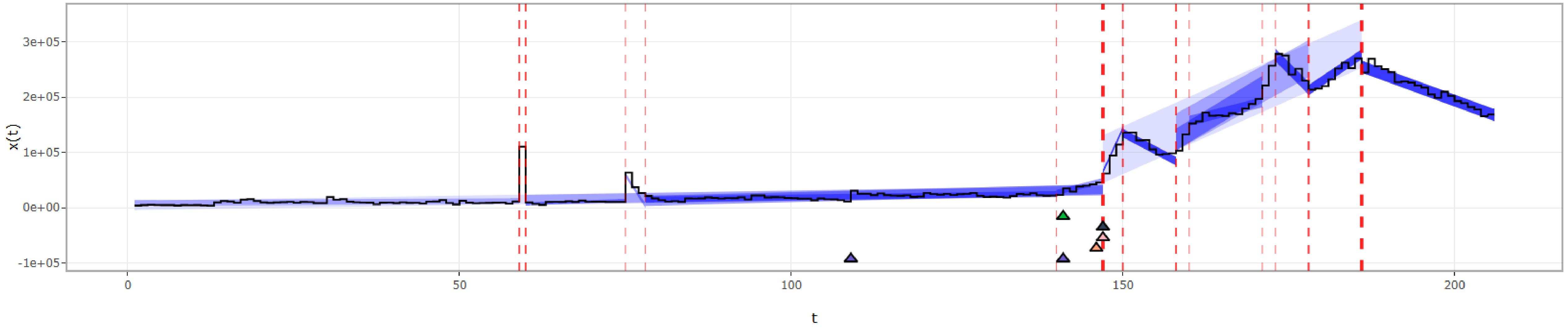
Demo available: https://huet.shinyapps.io/scoth-segmentation

Back-up slide: other qualitative results

Well-log data from Ó Ruanaidh and Fitzgerald (1996)



The number of applicants for a license plate in Shanghai. Data obtained from Kaggle



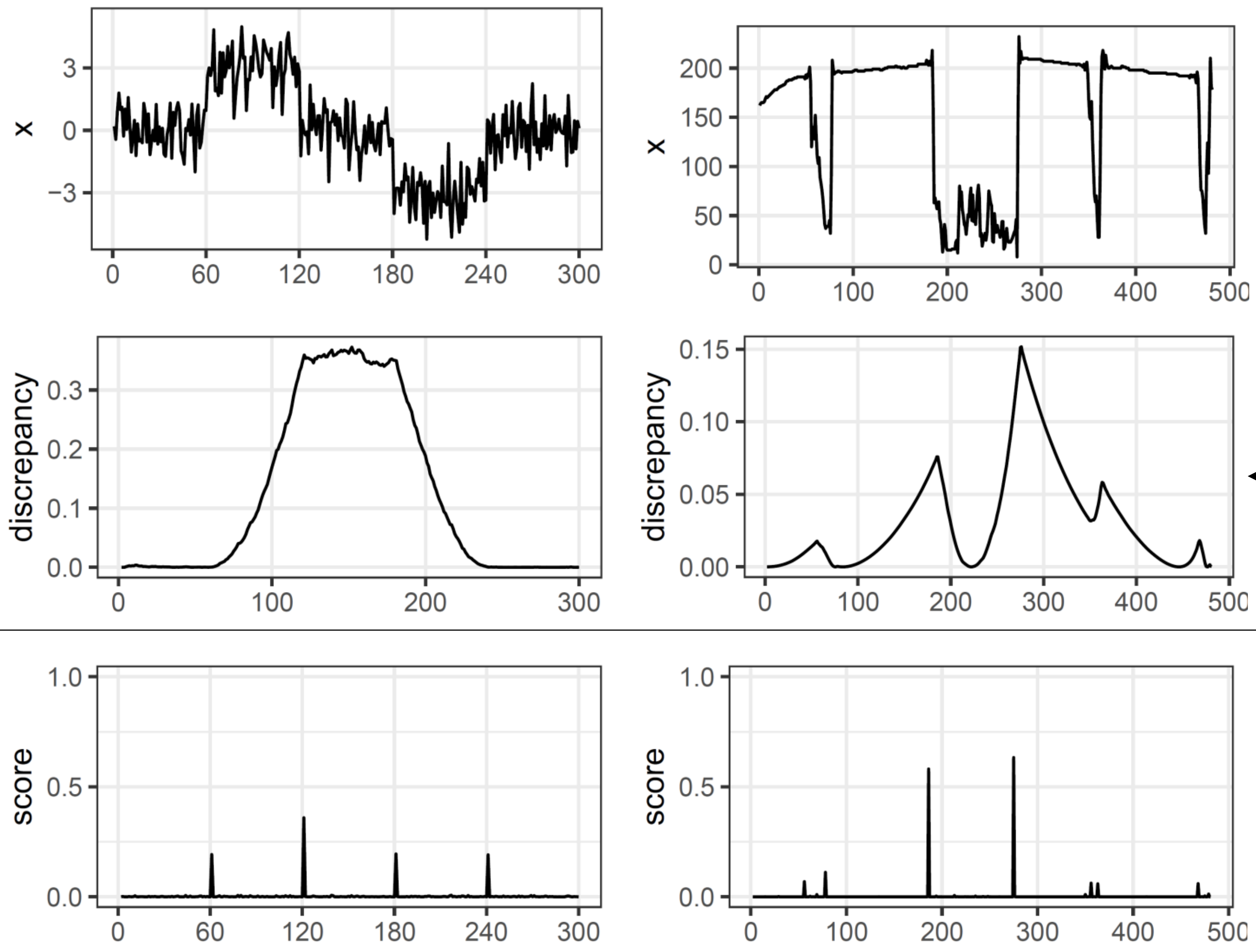
Annotator (ground truth labels) ▲ user 6 ▲ user 8 ▲ user 10 ▲ user 12 ▲ user 13

Detected changepoints at each level with our methodology ¦ level 1 ¦ level 2 ¦ level 3 ¦ level 4

5%-95% percentile region between consecutive changepoints ▬ level 1 ▬ level 2 ▬ level 3 ▬ level 4

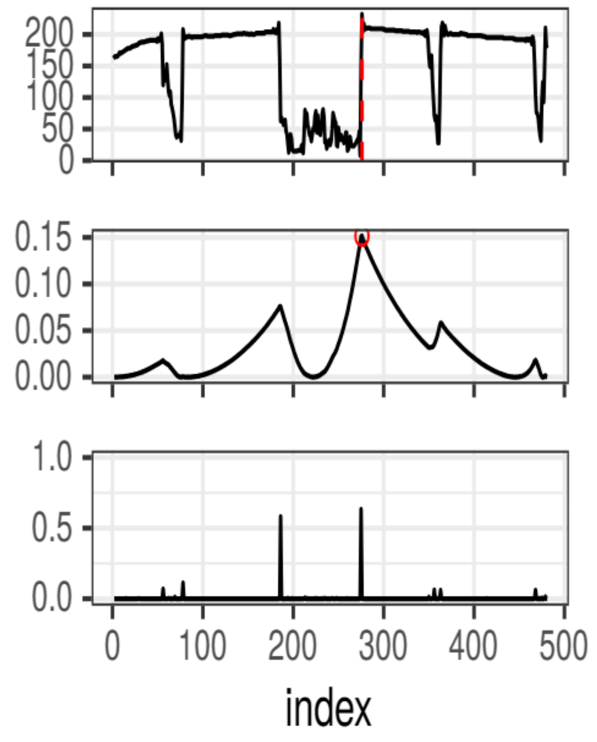
Back-up slide: series vs discrepancy vs scores

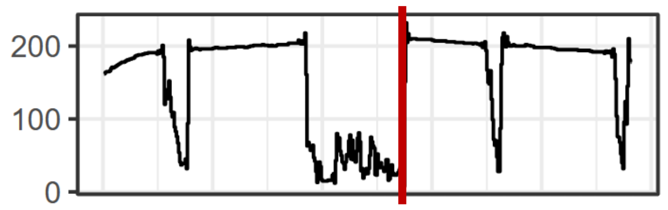
$$D(b; x_{[s,e)}, C) := C(x_{[s,e)}) - \big(C(x_{[s,b)}) + C(x_{[b,e)})\big)$$



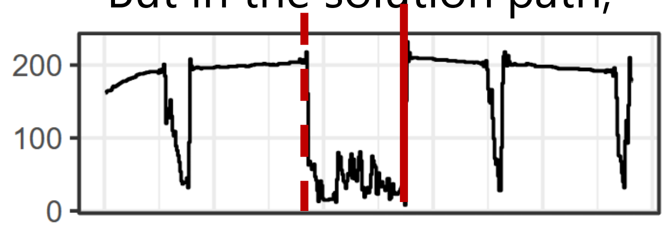
Here normalized discrepancy

Back-up slide: why scoring 0.63 while direct discrepancy is 0.15?





Direct discrepancy is 0.15 for this point.
But in the solution path,



scanline_42049

Bottom-up

```
274   185   77   55  355  467  362  478
0.15
0.63 0.56
0.63 0.58 0.04
0.63 0.58 0.11 0.07
0.52 0.58 0.11 0.07 0.01
0.52 0.58 0.11 0.07 0.00 0.04
0.52 0.58 0.11 0.07 0.06 0.05 0.06
0.52 0.58 0.11 0.07 0.06 0.06 0.06 0.01
----------------------------------------
scores  0.63 0.58 0.11 0.07 0.06 0.06 0.06 0.01
```

Given the dashed change, the red one has 0.5 gain.
And symmetrically, at that step (since using the max over the path, not the last element), the other one also has 0.5 gain

Back-up slide: algorithms

Algorithm 1 Scoring through maximum normalized gains vector

```
1: **Inputs** $x = (x_1, \dots x_T)$ series; $C$ cost function.
2: $S \leftarrow [\![2, T]\!]$    ▷ initial segmentation, as a fully segmented set
3: score $\leftarrow (0)_{[\![2,T]\!]}$    ▷ initial scores, vector of zeros of length $T - 1$
4: while $S \neq \varnothing$ :
5:     for $b \in S$ :
6:         score$(b) \leftarrow \max(\text{score}(b), G(b, S))$    ▷ update scores
7:     $S \leftarrow S \setminus \text{argmin}_{b \in S}$ score$(b)$    ▷ remove the worst cut
8: **Outputs** score$/V(\varnothing)$    ▷ returns normalized scores $\in [0, 1]$.
```

Algorithm 2 Recursive thresholdings

```
1: **Inputs** threshold$\in [0, 1]$; score vector; $x$ series; $C$ cost
2: $L \leftarrow []$    ▷ placeholder for the changes at each level
3: $L[0] \leftarrow \varnothing$    ▷ level 0
4: level $\leftarrow 1$
5: $L[\text{level}] \leftarrow \{b \ ; \ \text{score}(b) \geq \text{threshold}\}$    ▷ level 1 changes
6: while $L[\text{level}] \neq L[\text{level} - 1]$ :
7:     $S \leftarrow L[\text{level}]$    ▷ all the changes marked up to now
8:     $L[\text{level} + 1] \leftarrow L[\text{level}] \cup \{b \notin S; \text{score}(b)\frac{V(\varnothing)}{V(S)} \geq \text{threshold}\}$
9:     level $\leftarrow$ level $+ 1$
10: **Outputs** $L$    ▷ list of changes at each level.
```

Back-up slide: ablation study

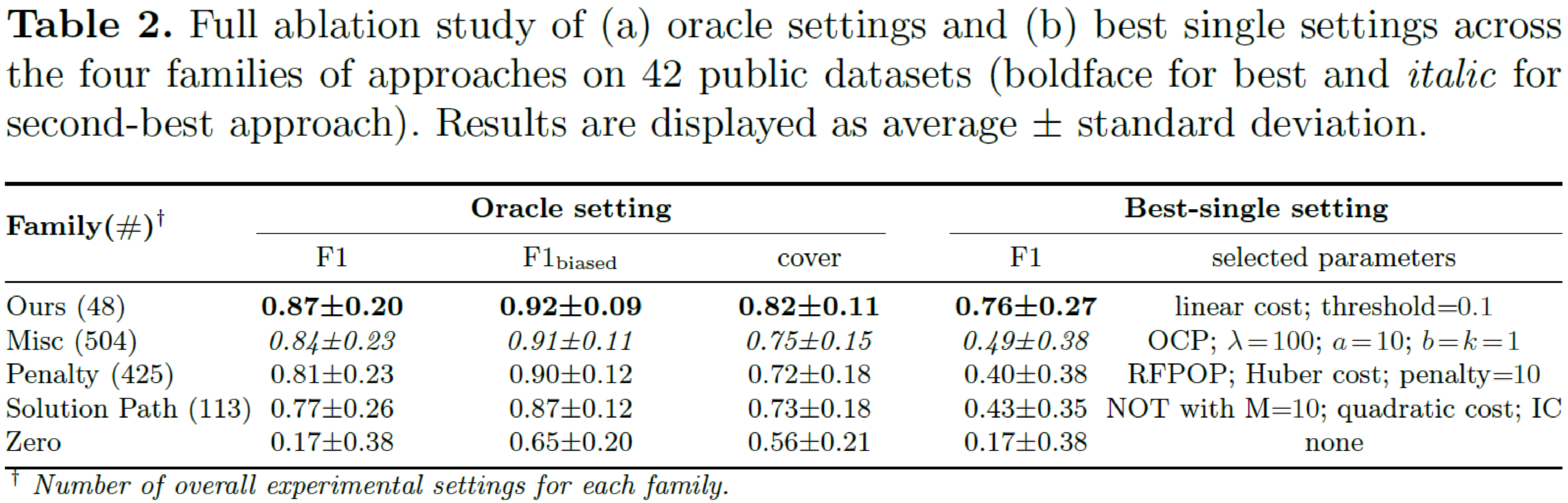


Table 2. Full ablation study of (a) oracle settings and (b) best single settings across the four families of approaches on 42 public datasets (boldface for best and *italic* for second-best approach). Results are displayed as average ± standard deviation.

Family(#)[†]	**Oracle setting**			**Best-single setting**	
	F1	$F1_{\text{biased}}$	cover	F1	selected parameters
Ours (48)	**0.87±0.20**	**0.92±0.09**	**0.82±0.11**	**0.76±0.27**	linear cost; threshold=0.1
Misc (504)	*0.84±0.23*	*0.91±0.11*	*0.75±0.15*	*0.49±0.38*	OCP; $\lambda=100$; $a=10$; $b=k=1$
Penalty (425)	0.81±0.23	0.90±0.12	0.72±0.18	0.40±0.38	RFPOP; Huber cost; penalty=10
Solution Path (113)	0.77±0.26	0.87±0.12	0.73±0.18	0.43±0.35	NOT with M=10; quadratic cost; IC
Zero	0.17±0.38	0.65±0.20	0.56±0.21	0.17±0.38	none

[†] *Number of overall experimental settings for each family.*

Back-up slide: tested algorithms

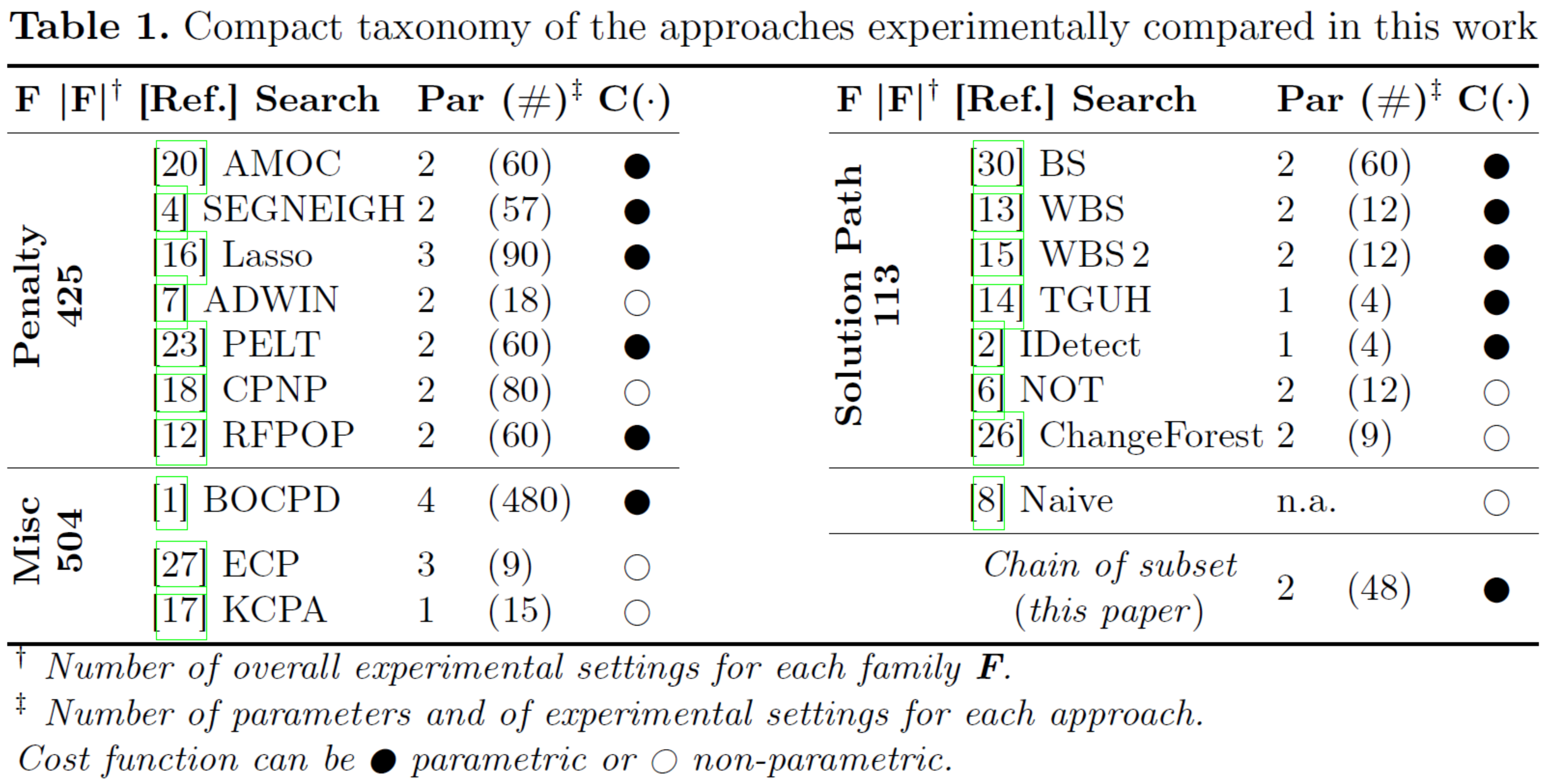


Table 1. Compact taxonomy of the approaches experimentally compared in this work

F	**|F|**[†]	**[Ref.]**	**Search**	**Par**	**(#)**[‡]	**C(·)**
Penalty	**425**	[20]	AMOC	2	(60)	●
		[4]	SEGNEIGH	2	(57)	●
		[16]	Lasso	3	(90)	●
		[7]	ADWIN	2	(18)	○
		[23]	PELT	2	(60)	●
		[18]	CPNP	2	(80)	○
		[12]	RFPOP	2	(60)	●
Misc	**504**	[1]	BOCPD	4	(480)	●
		[27]	ECP	3	(9)	○
		[17]	KCPA	1	(15)	○
Solution Path	**113**	[30]	BS	2	(60)	●
		[13]	WBS	2	(12)	●
		[15]	WBS 2	2	(12)	●
		[14]	TGUH	1	(4)	●
		[2]	IDetect	1	(4)	●
		[6]	NOT	2	(12)	○
		[26]	ChangeForest	2	(9)	○
		[8]	Naive	n.a.		○
			Chain of subset (this paper)	2	(48)	●

[†] *Number of overall experimental settings for each family* ***F***.

[‡] *Number of parameters and of experimental settings for each approach.*

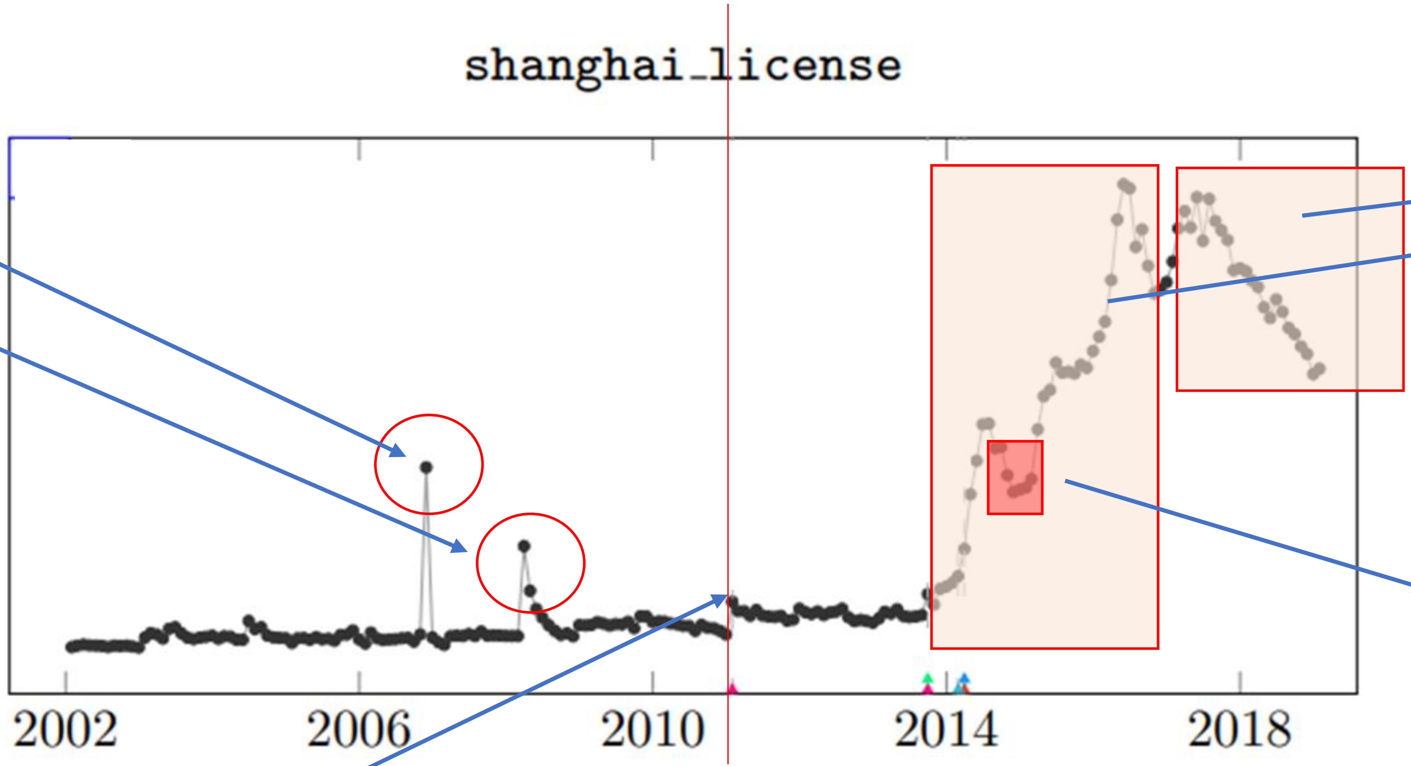
Cost function can be ● *parametric or* ○ *non-parametric.*

Back-up slide: disambiguation of the terms

The terms “anomaly” and “change point” have different meaning depending on the context/researcher

anomaly

➔ spikes
(global with respect to the past)



segmentation

➔ long disruption of the pattern

contextual anomaly

➔ short disruption

change point

➔ change in distribution

Figure 36: The number of applicants for a license plate in Shanghai. Data obtained from Kaggle.