KDD 2022 Tutorial

Alibaba DAMO Academy

Robust Time Series Analysis and Applications: An Industrial Perspective

Qingsong Wen
Staff Engineer
Alibaba DAMO

Linxiao Yang
Senior Engineer
Alibaba DAMO

Tian Zhou
Senior Engineer
Alibaba DAMO

Liang Sun
Engineering Director
Alibaba DAMO

Tutorial Website: https://github.com/DAMO-DI-ML/KDD2022-Tutorial-Time-Series

Who we are

- Alibaba DAMO Academy - Decision Intelligence Lab
  - Research Focus:
    - **AI for Time Series (AI4TS)**
    - **Explainable Artificial Intelligence (XAI)**
    - **Optimization**
    - **Others: ML, RL, ...**
  - Products and Applications:
    - **Green Energy AI:** Energy Forecasting and Scheduling
    - **Optimization Solver:** MindOpt
    - **AIOps:** Cloud Autoscaling AHPA
    - **...**
  - More Information:
    - https://damo.alibaba.com/labs/decision-intelligence

Outline

- **Introduction** (by Liang Sun)
- **Preliminaries** (by Liang Sun)
- **Robust Time Series Processing Blocks**
  - Time Series Periodicity Detection (by Qingsong Wen)
  - Time Series Trend Filtering (by Qingsong Wen)
  - Time Series Seasonal-Trend Decomposition (by Qingsong Wen)
  - Time Series Similarity (by Liang Sun)
- **Robust Time Series Applications and Practices**
  - Forecasting: Tree Models, Deep Ensemble, Transformers, etc. (by Qingsong Wen)
  - Autoscaling: Query Modeling, Scaling Decision, etc. (by Qingsong Wen)
  - Anomaly Detection: Decomposition Model, Deep State Space Model, Transformers, etc. (by Qingsong Wen)
  - Fault Cause Localization: Rule Set Learning, Root Cause Analysis, etc. (by Liang Sun)

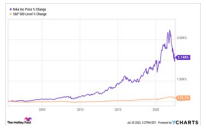
Outline

- ***Introduction***
- **Preliminaries**
- **Robust Time Series Processing Blocks**
- **Robust Time Series Applications and Practices**

Time Series Data is Ubiquitous

- A wide range of time series data
  - AIOps
  - IoT
  - Business data, e.g., sales volume, stock price
  - Many others

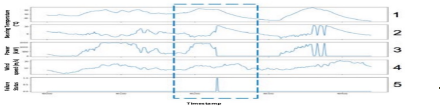
stocks



sales

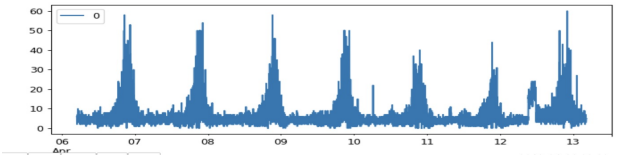
goods consumption

sensor

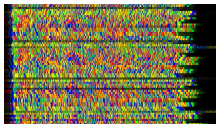


power demand

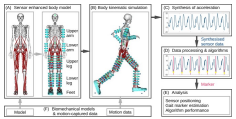
Cloud service monitoring



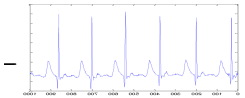
DNA sequence



motion detect

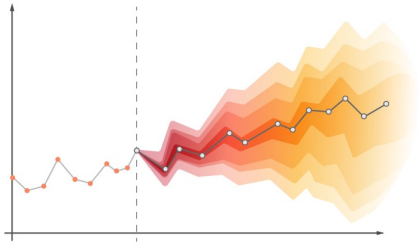


ECG

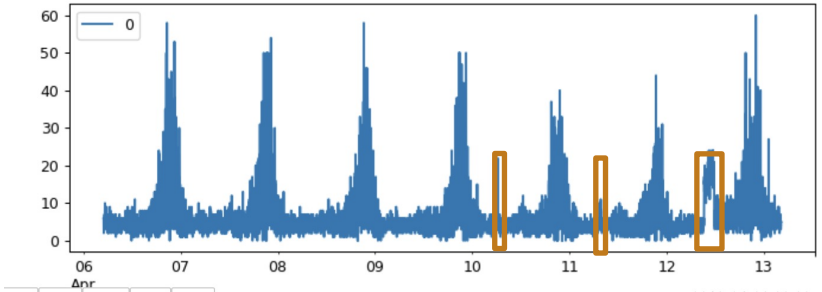


Typical Applications of Time Series

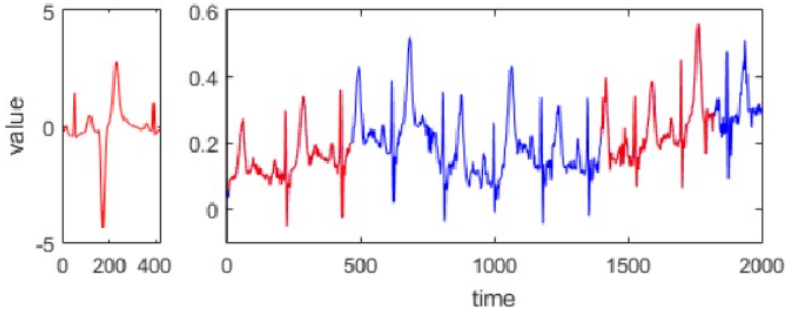
Time Series Forecasting



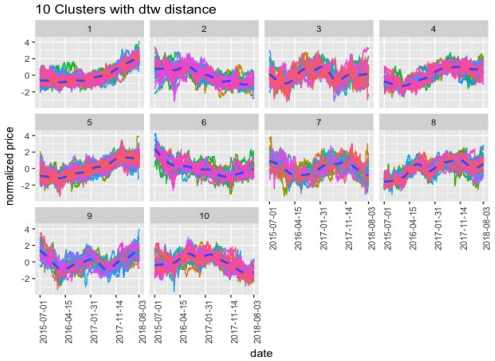
Time Series Anomaly Detection



Time Series Search/Query

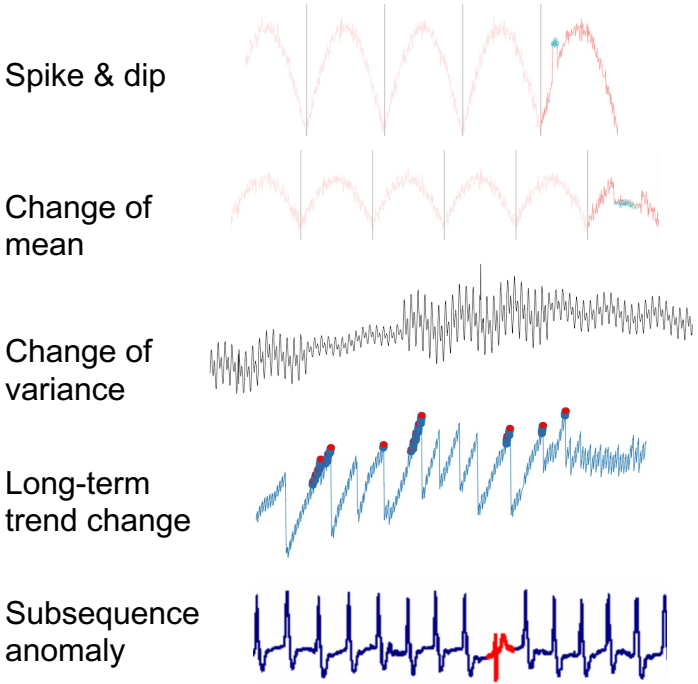


Time Series Classification/Clustering



AIOps: From Anomaly Detection to Root Cause Analysis

Different Types of Anomalies



- Business service
- Software service
- Container and VM
- Server and components
- Network
- Data center infrastructure

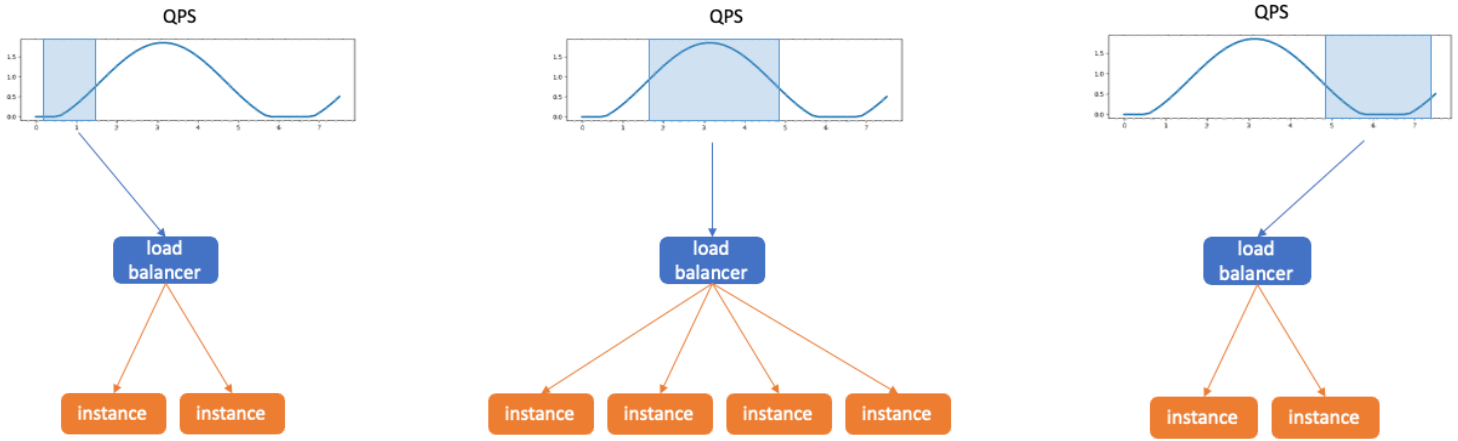
In a typical applications of Elastic Computing Service at Alibaba Cloud (2020):

- 5M+ servers monitored
- 50M+ metrics monitored
- In 2020 some typical root identification time is ~ 1 person-month
- Some cases incur 1M+ USD loss

- ***Huge amount of data calls for automatic and accurate anomaly detection algorithm***
- ***Finding the root is the ultimate goal***

From Forecasting to Decision: AutoScaling

- Autoscaling in cloud computing is an effective method to improve the usage of computing resources
  - It automatically allocates resources for cloud-based applications while maintaining SLA (service level agreement)
  - Horizontal scaling (add/delete instances or VMs) vs vertical scaling (up/downgrade CPU, RAM, network, etc.)
  - Time series forecasting and decision-making on resources



Applications of Time Series in Industry

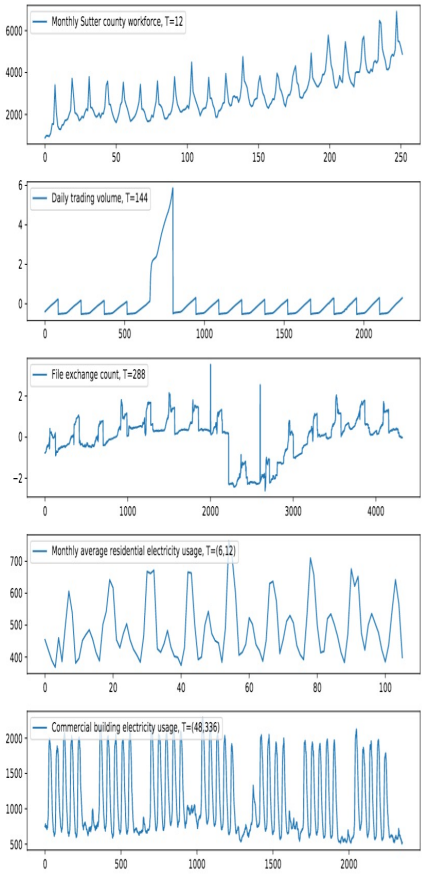
- Huge amount of data
  - Hundreds of millions metrics or even more
  - High frequency: unlike traditional time series, many time series data are sampled with higher frequency (e.g., every minute or several minutes)
  - Very limited labels in many applications
  - Many applications (e.g., anomaly detection) require fast or real-time processing
  - Low deployment cost
- Time series data with different complex types of characteristics
  - Noises and outliers
  - Periodicity/Seasonality: no/single/multiple periodic components, periodic component shift/change
  - Fully automatic solution is challenging but highly preferred
- Closely connected with other applications
  - Anomaly detection is closed related to root cause analysis
  - Forecasting is closely related to decision-making (e.g., autoscaling)

*Our solution: **robust** algorithm family for time series analysis*

Why Periodicity so Important?

- Periodic time series dominate in many real-world applications
  - In AIOps, 60%+ lower level monitoring data is periodic, and 90%+ higher level monitoring data is periodic
  - In electricity load data, 100% system load data is periodic, and 90%+ bus load data is periodic
  - Many more periodic time series in a wide range of real-world applications
- Why periodicity so important?
  - Periodic and non-periodic time series requires quite different processing
  - Different periodic patterns requires different processing
- Challenges
  - Complicated periodic patterns: multiple periodic components, periodic shift and change
  - Change of trend
  - Noises/outliers

Different types of periodic time series



Outline

- ❑ **Introduction**
- ➢ ***Preliminaries***
  - Robust Statistics: Robust Regression, M-estimators
  - Optimization Algorithms: ADMM, MM Algorithm
  - Signal Processing: Fourier, Wavelet
  - Deep Learning: RNN, CNN, GNN, Transformer, Data Augmentation
- ❑ **Robust Time Series Processing Blocks**
- ❑ **Robust Time Series Applications and Practices**

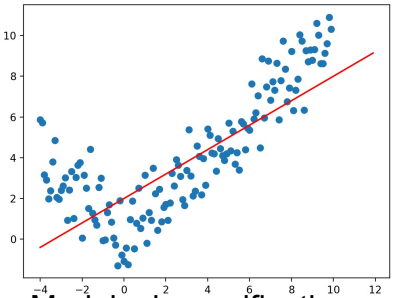
Robust Regression

- Regression methods rely on assumptions
  - Assumptions on model, e.g. linear
  - Assumptions on noise, e.g. Gaussian

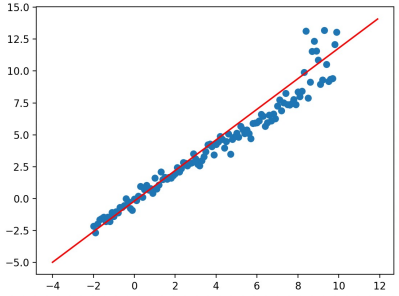
$$y = f(x;\theta) + \epsilon$$

linear Gaussian

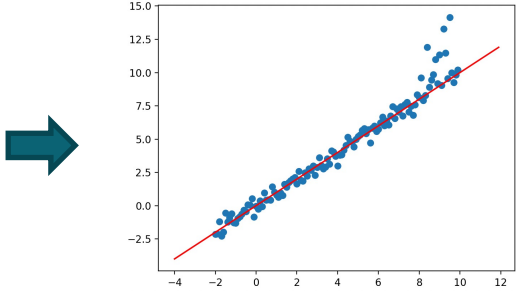
- When assumptions not hold



Model misspecification



Outliers



Robust regression

- Robust regression
  - Robust regression methods are designed to be not overly affected by violations of assumptions by the underlying data-generating process.

M-estimator

- Generalized maximum likelihood estimation

$$\theta^* \leftarrow \arg\min_{\theta} \sum_{i=1}^{N} r^2(\theta) \quad \Rightarrow \quad \theta^* \leftarrow \arg\min_{\theta} \sum_{i=1}^{N} \rho(r(\theta))$$

Assuming Gaussian Error

residual

M-Estimator

- Properties of ρ
  - Continuous
  - Symmetrical $\rho(\xi) = \rho(-\xi)$
  - ρ(0) = 0
  - Positive: $\rho(\xi) > 0 \; for \; \xi \neq 0$
  - Increasing monotonic $\rho(\xi) > \rho(\xi')$ for $|\xi| > |\xi'|$, but it does not increase too much as ξ increases

M-estimator

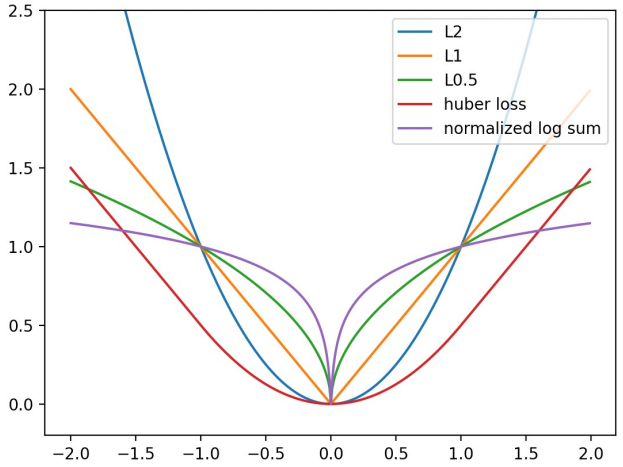
- Robust M-estimator
  - L1-norm

$$\rho(\xi) = |\xi|$$

  - Huber-loss

$$\rho_\epsilon(\xi) = \begin{cases} \frac{1}{2}\xi^2 & |\xi| \le \epsilon \\ \epsilon(|\xi| - \frac{1}{2}\epsilon) & \text{otherwise} \end{cases}$$

  - Lp (p<1) pseudo-norm
  - Log-sum loss



Alternating direction method of multipliers

- Consider the problem

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) + g(\boldsymbol{z}) \quad \text{s.t. } \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z}$$

- The augmented Lagrangian of the problem is given as

$$L_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + g(\boldsymbol{z}) + \boldsymbol{\lambda}^T(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x} - \boldsymbol{B}\boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x} - \boldsymbol{B}\boldsymbol{z}\|^2$$

- ADMM process

$\rho > 0$ is called the penalty parameter

$$\begin{aligned}
\boldsymbol{x}^{(k+1)} &= \arg\min_{\boldsymbol{x}} L_\rho(\boldsymbol{x}, \boldsymbol{z}^{(k)}, \boldsymbol{\lambda}^{(k)}) \\
\boldsymbol{z}^{(k+1)} &= \arg\min_{\boldsymbol{z}} L_\rho(\boldsymbol{x}^{(k+1)}, \boldsymbol{z}, \boldsymbol{\lambda}^{(k)}) \\
\boldsymbol{\lambda}^{(k+1)} &= \boldsymbol{\lambda}^{(k)} + \rho(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^{(k+1)} - \boldsymbol{B}\boldsymbol{z}^{(k+1)})
\end{aligned}$$

Alternating direction method of multipliers

- Advantages of ADMM:
  - Not require gradient
  - Converges to modest accuracy within a few tens of iterations, sufficient for many applications in ML

- Apply to robust regression:

$\min_{\boldsymbol{x}} f(\boldsymbol{x}) + g(\boldsymbol{x})$

data fitting term

regularization term, such as L1 norm, may be difficult to optimize

Regression Problem

$\min_{\boldsymbol{x}} f(\boldsymbol{x}) + g(\boldsymbol{z}) \quad \text{s.t. } \boldsymbol{x} = \boldsymbol{z}$

Variable split

$$\boldsymbol{x}^{(k+1)} = \text{prox}_{f,1/\rho}(\boldsymbol{z}^{(k)} - \frac{1}{\rho}\boldsymbol{\lambda}^{(k)})$$

$$\boldsymbol{z}^{(k+1)} = \text{prox}_{g,1/\rho}(\boldsymbol{x}^{(k+1)} + \frac{1}{\rho}\boldsymbol{\lambda}^{(k)})$$

$$\boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} + \rho(\boldsymbol{x}^{(k+1)} - \boldsymbol{z}^{(k+1)})$$

$$\text{prox}_f(v) = \arg\min_{x \in \mathcal{X}} \left(f(x) + \frac{1}{2}\|x - v\|_{\mathcal{X}}^2 \right)$$

Update using ADMM

Wang, Junxiang, Fuxun Yu, Xiang Chen, and Liang Zhao, "ADMM for efficient deep learning with global convergence." KDD 2019.

Majorization-Minimization (MM) Algorithms

- Consider the problem

$$\min_{\boldsymbol{x}} f(\boldsymbol{x})$$ Maybe difficult to optimize

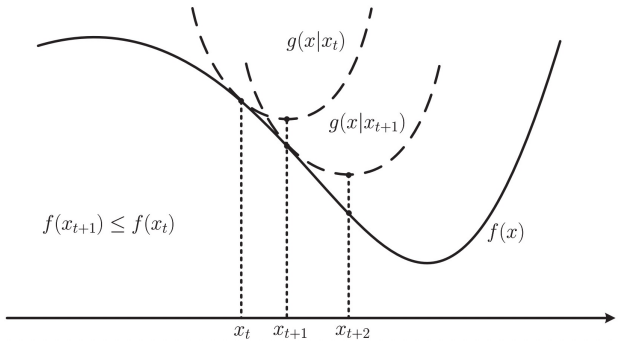
- Surrogate function

$$g(\boldsymbol{x}|\boldsymbol{x}_t) \geq f(\boldsymbol{x}) + c_t \;\; \text{where} \;\; c_t = g(\boldsymbol{x}_t|\boldsymbol{x}_t) - f(\boldsymbol{x}_t)$$

Easy to optimize

- Convergence guarantee

$$f\left(\mathbf{x}_{t+1}\right) \leq g\left(\mathbf{x}_{t+1}|\mathbf{x}_t\right) - c_t \leq g\left(\mathbf{x}_t|\mathbf{x}_t\right) - c_t = f\left(\mathbf{x}_t\right)$$



Sun, Ying, Prabhu Babu, and Daniel P. Palomar. "Majorization-minimization algorithms in signal processing, communications, and machine learning." IEEE Transactions on Signal Processing 65.3 (2016): 794-816.

Majorization-Minimization Algorithms

- Surrogate function construction
  - Taylor Expansion

Lp norm minimization: $|\mu|^{\frac{p}{2}} \leq \frac{p}{2}|\mu^{(t)}|^{(\frac{p}{2}-1)}|\mu| \xRightarrow{|\mu| = x^2} |x|^p \leq p|x^{(t)}|^{(p-2)}x^2$

  - Convexity Inequality

$$f_{\text{cvx}}\left(\sum_{i=1}^{n} w_i \mathbf{x}_i\right) \leq \sum_{i=1}^{n} w_i f_{\text{cvx}}\left(\mathbf{x}_i\right)$$

Examples: *Jensen's Inequality*

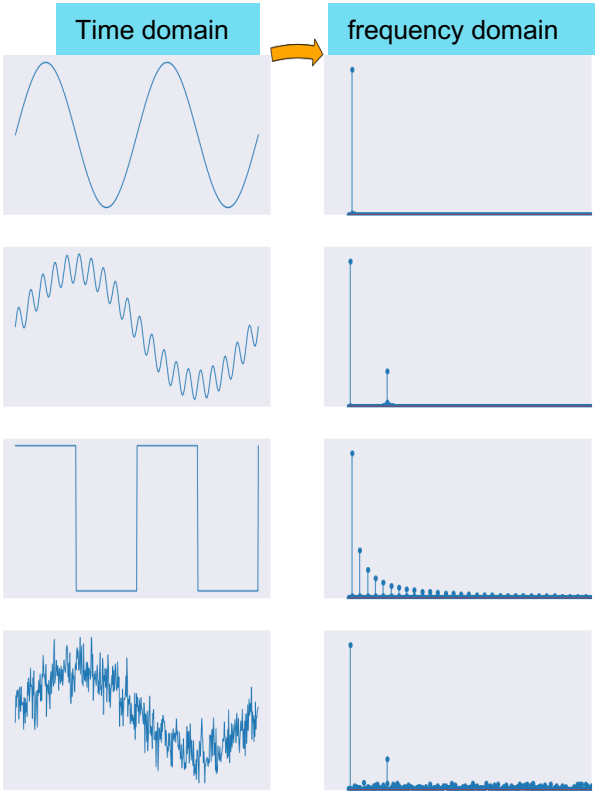
  - Arithmetic-Geometric Mean Inequality

L1 norm minimization: $x^2 + (x^{(t)})^2 \geq 2|x||x^{(t)}| \Rightarrow |x| \leq \frac{x^2 + (x^{(t)})^2}{2x^{(t)}}$

Signal Processing

- Time domain and Frequency domain
  - Time domain representation: how a signal changes over time
  - Frequency domain representation: how much of a signal lies within each frequency over a band of frequencies
- Fourier Transform
  - Mathematical transform that decompose signal of time domain into signal of frequency domain using sine and cosine

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi\xi x} dx, \quad \forall \xi \in \mathbb{R}$$

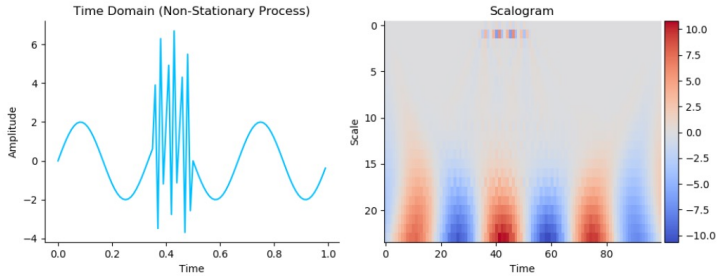


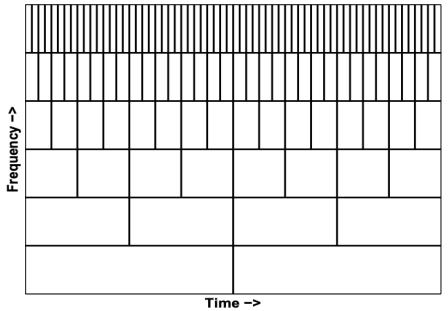
Signal Processing

- Short-time Fourier transform(STFT)
  - Divide a long time signal into short segments of equal length and compute the Fourier transform separately in each segment
- Wavelet Transform
  - Decompose the time signal by expanding by a family of orthonormal basis functions

$$X(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} \overline{\Psi\left(\frac{t-b}{a}\right)} x(t) dt$$

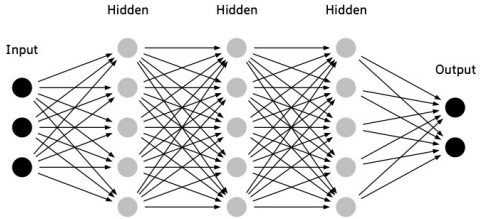
  - Transform the time series from time domain to the time-scale (or time–frequency) domain

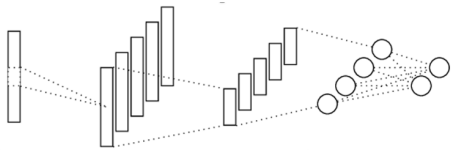


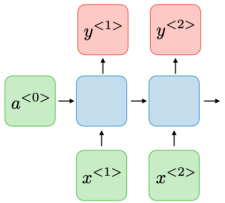


Deep Learning: Model

- MLP(multiple layer perceptron)
  - Fully connected feedforward artificial neural network
- CNN(Convolutional neural network)
  - Shared-weight architecture of filters that slide along input features
- RNN(Recurrent neural network)
  - Connection between nodes form a directed or undirected graph along a temporal sequence
- Transformer
  - Deep learning model adopts the mechanism of self-attention, differentially weighting the significance of each part of input sequence







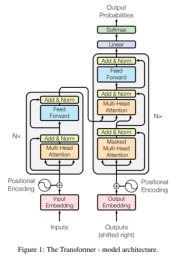
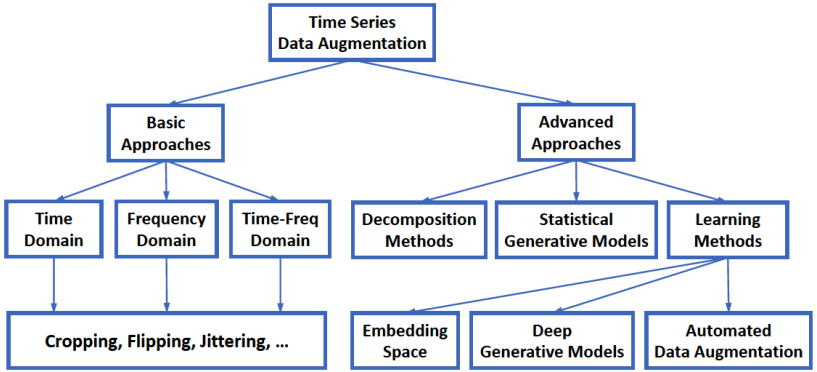
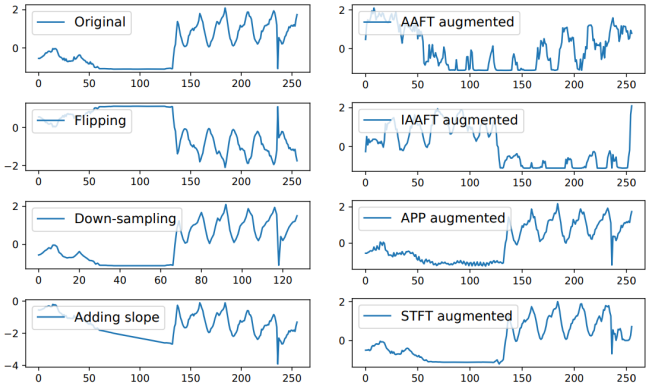


Figure 1: The Transformer - model architecture.

Data Augmentation

- Taxonomy
  - Basic Approaches: time, freq, time-freq
  - Advanced Approaches: decomp, generative, learning
- Time Domain
  - Flip & Down-sampling & Adding slope
  - Window warping & noise injection
- Frequency Domain
  - Amplitude and phase perturbations(APP)
  - Amplitude adjusted Fourier transform(AAFT)
- Time-Frequency Domain
  - Short Fourier transform based augmentation(STFT)





(a) time domain (b) (time-)frequency domain

Data augmentations in time, frequency, time-frequency domains

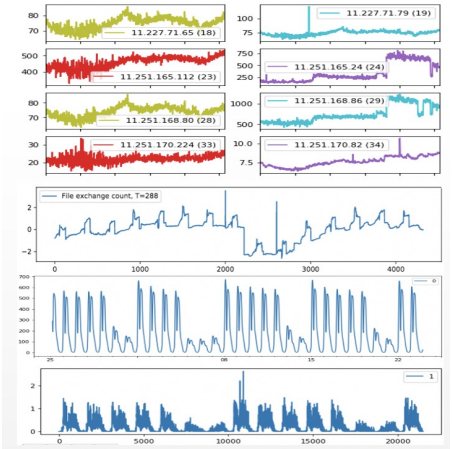
Qingsong Wen, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang, Huan Xu, "Time Series Data Augmentation for Deep Learning: A Survey," in the 30th International Joint Conference on Artificial Intelligence (IJCAI 2021), Montreal, Canada, Aug. 2021.

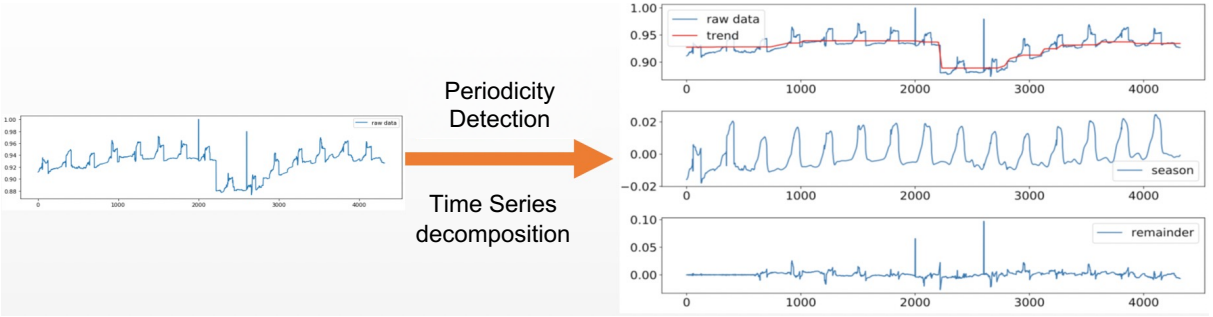
Outline

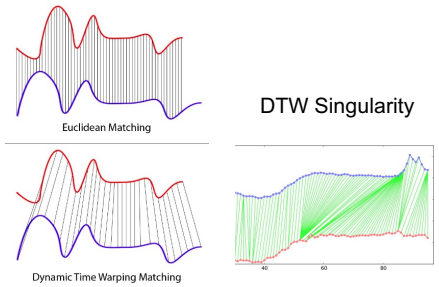
- ❑ **Introduction**
- ❑ **Preliminaries**
- ➢ ***Robust Time Series Processing Blocks***
  - Time Series Periodicity Detection
  - Time Series Trend Filtering
  - Time Series Seasonal-Trend Decomposition
  - Time Series Similarity
- ❑ **Robust Time Series Applications and Practices**

Complex Periodic Time Series

- Time series
  - Periodicity/seasonality, trend, and noises are common
  - Periodicity detection, decomposition, similarity are crucial for downstream tasks (forecasting, anomaly detection, etc.)
- Challenges of robust time series processing blocks
  - Noise, outliers, missing data, abrupt trend changes
  - None/single/multiple periodicity
  - Need automatic and accurate processing for large-scale industrial time series







Common Periodicity Detection Algorithms (ACF/FFT)

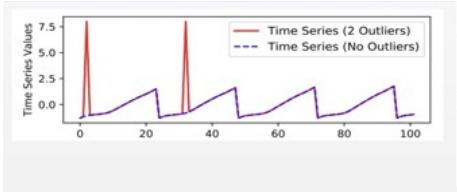
- Time domain: autocorrelation function (ACF)
  - Calculate normalized ACF for time series x_t

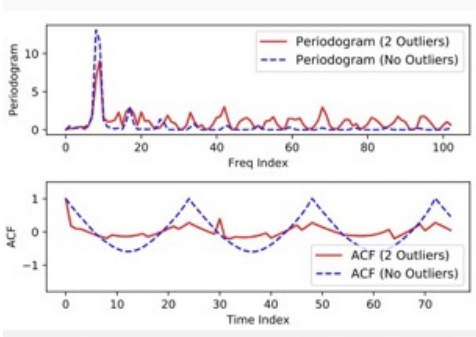
$$ACF(t) = \frac{1}{(N-t)\delta_x^2} \sum_{n=0}^{N-t-1} x_n x_{n+t}$$

  - Calculate the (mean/median) distance of ACF peaks exceeding predefined threshold as period length
- Freq domain: Fisher's test by Periodogram (FFT)
  - Fisher's test for dominant periodicity detection

$$g = max_k P_k / (\sum_{j=1}^{N} P_j) \qquad P_k = ||\mathrm{DFT}\{x_t\}||^2 = \frac{1}{N'} \sum_{t=0}^{N'-1} \left\| x_t e^{-i2\pi kt/N'} \right\|^2$$

  - Distribution of the periodogram: $P_k \sim (1/2)\sigma^2 \chi^2(2)$
  - If passed Fisher's test, period candidate: N'/k where $k = \arg\max_k P_k$



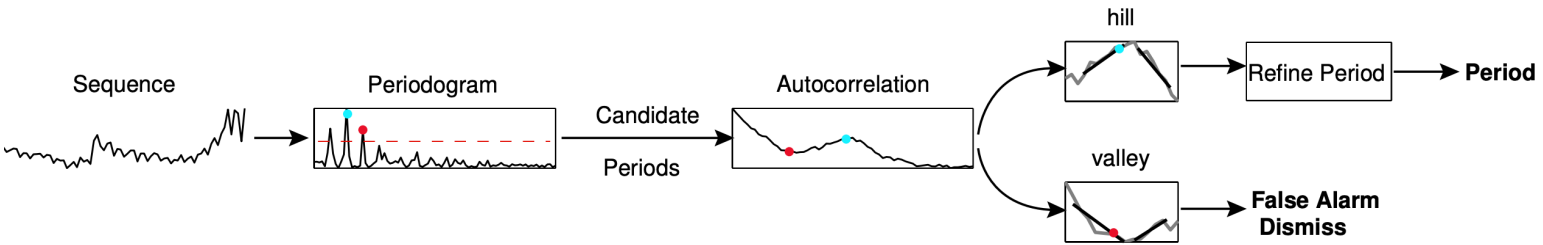


- Not robust to outliers
- Hard to handle multiple periodicities

AUTOPERIOD

- Key idea: Combining time and frequency processing for periodicity detection
- **Pros**: more accurate than time/freq domain only methods, like ACF, FFT
- **Cons**: hard to isolate and detect in complicated multi-period time series

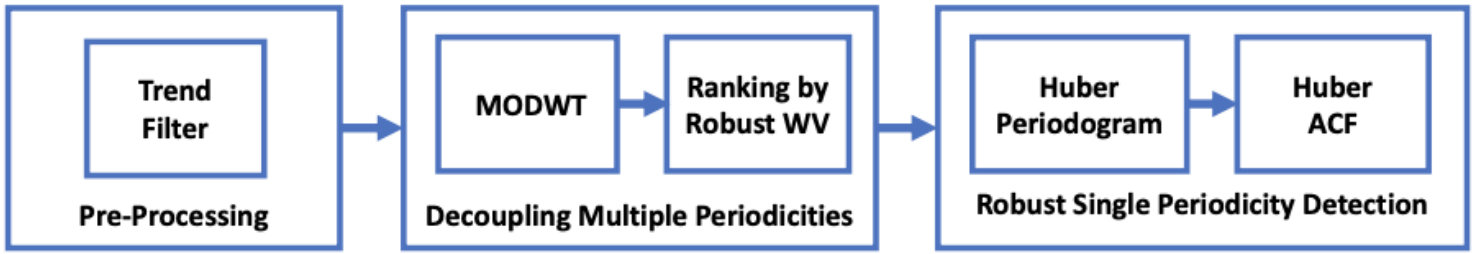
Method	Easy to threshold	Accurate short periods	Accurate large periods	Complexity
Periodogram	yes	yes	no	$O(NlogN)$
Autocorrelation	no	yes	yes	$O(NlogN)$
Combination	yes	yes	yes	$O(NlogN)$



Michail Vlachos, Philip Yu, and Vittorio Castelli. 2005. On periodicity detection and structural periodic similarity. In Proceedings of the 2005 SIAM International Conference on Data Mining (SDM), pp. 449–460, 2005.

RobustPeriod Algorithm

- Robust and general periodicity detection: RobustPeriod
  - Key idea: isolate periodic components, then detect each one robustly (“divide and conquer”)
  - Three blocks: Pre-processing, Decoupling Multiple Periodicities, Robust Single Periodicity Detection



$$y_t = \tau_t + \sum_{i=1}^{m} s_{i,t} + r_t$$

For periodic time series, period lengths are denoted as T_i ,i = 1, $\cdots m$, m is the number of periodic components

Qingsong Wen, Kai He, Liang Sun, Yingying Zhang, Min Ke, Huan Xu, "RobustPeriod: Time-Frequency Mining for Robust Multiple Periodicity Detection," in Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD 2021), Xi'an, China, Jun. 2021.

Decouple Multiple Periodicity

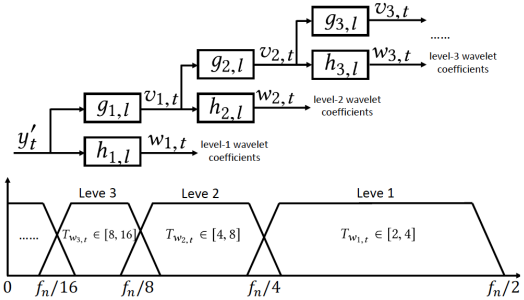
- MODWT for decoupling multiple periodicity
  - MODWT: maximal overlap discrete wavelet transform

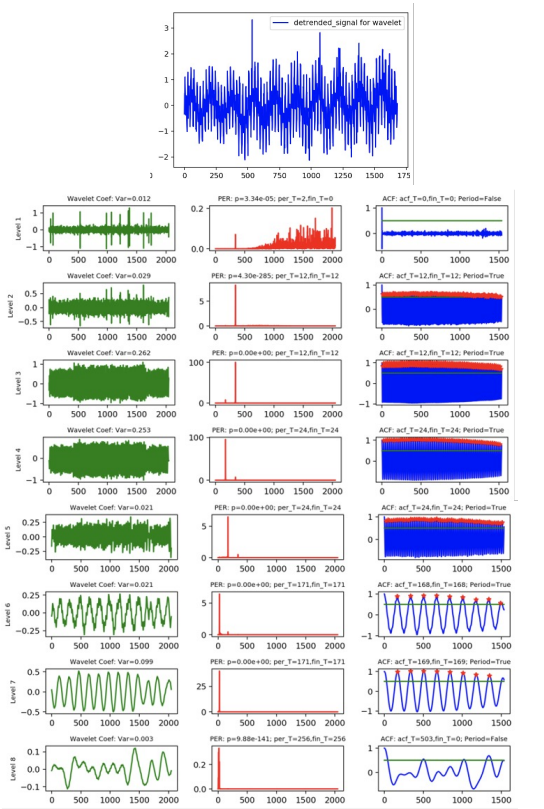
*j*th level wavelet coefficients

$$w_{j,t} = \sum_{l=0}^{L_j - 1} h_{j,l} y'_{t-l \bmod N}$$

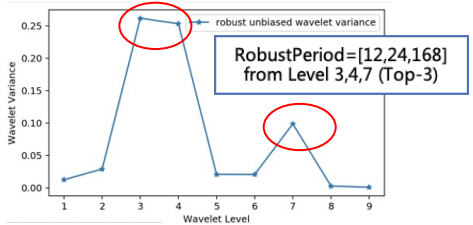
*j*th level scaling coefficients

$$v_{j,t} = \sum_{l=0}^{L_j - 1} g_{j,l} y'_{t-l \bmod N}$$





- Robust wavelet variance ranking for speedup
  - Wavelet variance decomposition: $\hat{\sigma}^2_{\mathbf{y}'} = \sum_{j=1}^{J_0} \hat{\sigma}^2_{\mathbf{w}_j} + \hat{\sigma}^2_{\mathbf{v}_{J_0}}$
  - Relationship to PSD: $\hat{\sigma}^2_{\mathbf{w}_j} \approx \int_{1/2^{j+1} \le |f| \le 1/2^j} S_{\mathbf{y}'}(f) df$
  - If there is a periodic component in the *j*th level wavelet coefficient, a large wavelet variance would be expected



RobustPeriod: Robust Single Periodicity Detection

- Original ACF for validating periodicity candidates
  - Calculate normalized ACF for each wavelet coefficients

$$ACF(t) = \frac{1}{(N-t)\delta_w^2} \sum_{n=0}^{N-t-1} w_n w_{n+t}$$

O(N^2), not robust to outliers

  - Calculate the median distance of ACF peaks exceeding predefined threshold as period length

- Robust Huber-ACF
  - Utilizing Wiener-Khinchin theorem based on Huber-periodogram

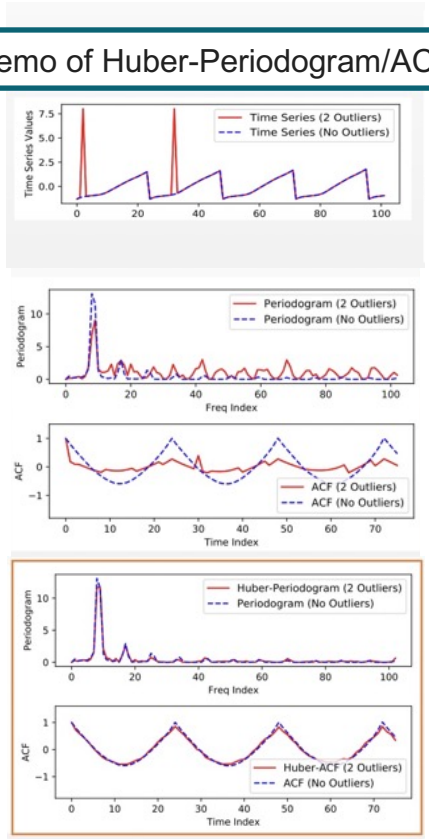
$$HuberACF(t) = \frac{p_t}{(N-t)p_0}$$

O($N\log N$), robust to outliers

$$p_t = \text{IDFT}\{\bar{P}_k\} = \frac{1}{\sqrt{N'}} \sum_{k=0}^{N'-1} \bar{P}_k e^{i2\pi kt/N'}$$

$$\bar{P}_k = \begin{cases} P_k^M & k = 0, 1, \cdots, N-1 \\ \left(\sum_{k=0}^{N-1} x_{2k} - x_{2k+1}\right)^2 \Big/ N' & k = N \\ P_{N'-k}^M & k = N+1, \cdots, N'-1 \end{cases}$$

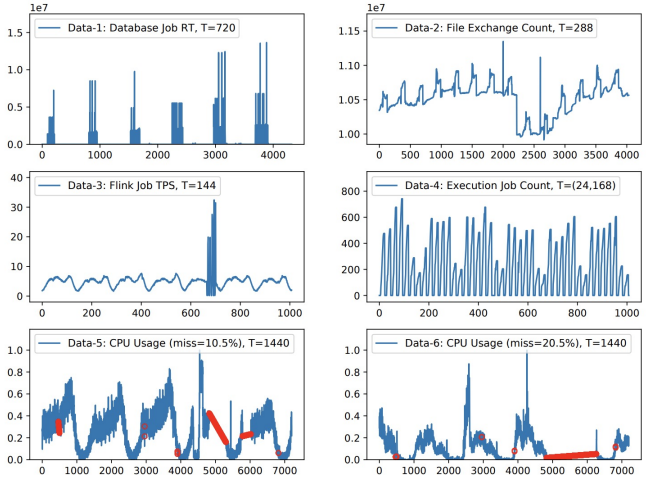
Demo of Huber-Periodogram/ACF



Periodicity Detection Comparisons

- Real-world datasets:
  - Alibaba Cloud monitoring data with outliers, noise, trend, missing data (**single/multiple** periodicity)

Monitoring datasets from Alibaba Cloud



Algorithms	Data-1, T=720 Database RT	Data-2, T=288 File Exchange	Data-3, T=144 Flink TPS
Siegel	(655,769,...)	(288,576,...)	(141,144)
AUTOPERIOD	(353,241,9)	(288,439,...)	(68,141)
Wavelet-Fisher	(372,745,...)	(282,585,...)	(73,146)
RobustPeriod	**721**	**288**	**144**

Algorithms	Data-4, T=(24,168) Job Count	Data-5, T=1440 CPU Usage	Data-6, T=1440 CPU Usage
Siegel	(24,168)	(1459,2597,...)	(1575,1063,...)
AUTOPERIOD	(24,26)	(1488,739,...)	(366,2880,...)
Wavelet-Fisher	(12,24,...)	(1489,712,...)	(1489,364,...)
RobustPeriod	**(24,168)**	**1431**	**1426**

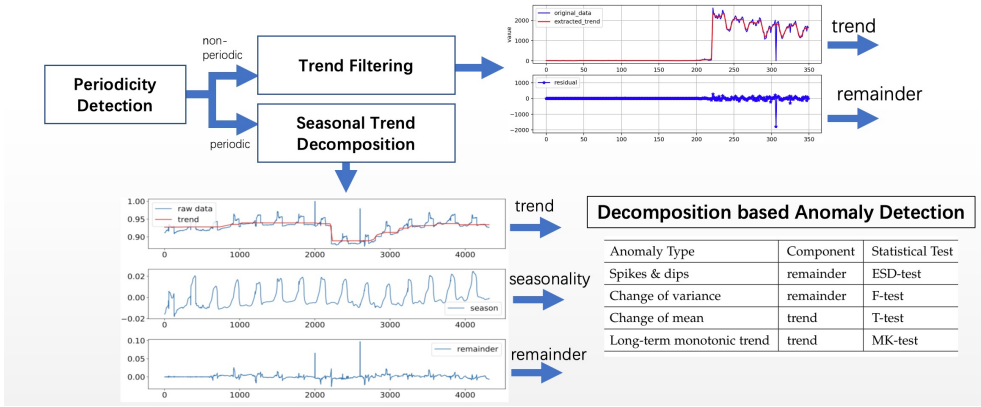
Algorithms	Robust to outliers	Robust to amplitude changes	Robust to trend changes	Support multiple periods	Not need priors for number of periods
ACF	×	×	×	×	×
FFT-Fisher	×	×	×	×	×
FFT-Siegel	×	×	×	✓	✓
Bandpass+ACF	×	✓	×	✓	×
Wavelet-Fisher	×	×	×	✓	✓
AUTOPERIOD	×	×	×	✓	✓
RobustPeriod	✓	✓	✓	✓	✓

Time Series Decomposition

- Trend filtering, seasonal-trend decomposition

$$y_t = \tau_t + r_t \qquad y_t = \tau_t + \sum_{i=1}^{m} s_{i,t} + r_t$$

- Why need decomposition
  - Insights and Interpretability from different components
  - Different utilization by components
    - e.g.: anomaly detection



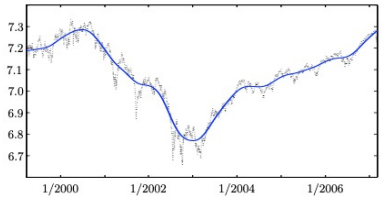
Non-periodic time series: Common Trend Filtering

- Moving average: not accurate with delay
- Hodrick–Prescott Filtering (H-P filter)
  - Regularization with ℓ2 norm of the second difference operator
  - linear estimation, convex with closed-form solution
- ℓ1 trend filtering
  - Regularization with ℓ1 norm for piecewise linear trend estimation
  - Nonlinear estimation, convex and linear computational complexity
- Challenges
  - Not robust to outliers and abrupt trend changes (HP)
  - Staircasing effect exists (ℓ1)

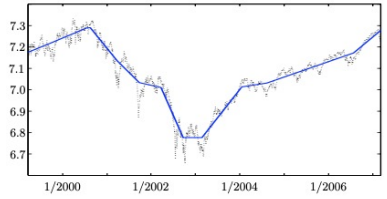
$$\frac{1}{2}\sum_{t=0}^{N-1}(y_t-\tau_t)^2+\lambda\sum_{t=1}^{N-2}(\tau_{t-1}-2\tau_t+\tau_{t+1})^2$$

$$\mathbf{D}^{(2)}=\begin{bmatrix}1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \ddots & & \\ & & 1 & -2 & 1\end{bmatrix}$$

$$\frac{1}{2}||\mathbf{y}-\boldsymbol{\tau}||_2^2+\lambda||\mathbf{D}^{(2)}\boldsymbol{\tau}||_2$$



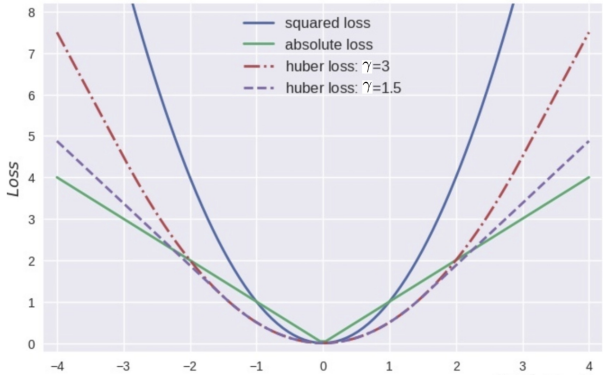
$$\frac{1}{2}||\mathbf{y}-\boldsymbol{\tau}||_2^2+\lambda||\mathbf{D}^{(2)}\boldsymbol{\tau}||_1$$



Seung-Jean Kim, Kwangmoo Koh, Stephen Boyd, Dimitry Gorinevsky. “ℓ1 trend filtering,” SIAM review 51, no. 2, pp 339-360, 2009.

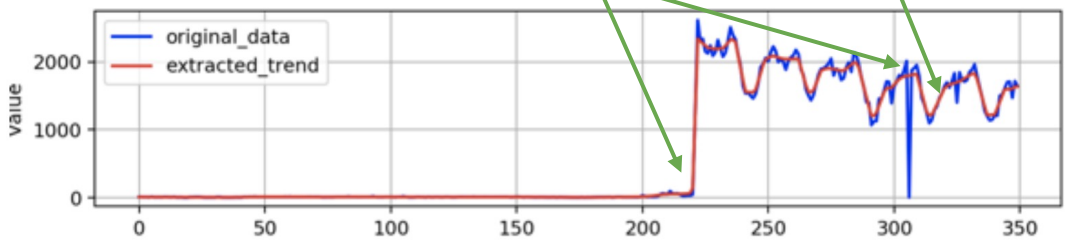
RobustTrend Filter

- Proposed RobustTrend filter for time series
  - Huber loss: robust to outliers
  - 1st order L1 regularization: abrupt trend changes
  - 2nd order L1 regularization: slow trend changes and can effectively reduce staircasing effect



$$\min_{\boldsymbol{\tau}} \; g_\gamma(\mathbf{y}-\boldsymbol{\tau}) + \lambda_1 ||\mathbf{D}^{(1)}\boldsymbol{\tau}||_1 + \lambda_2 ||\mathbf{D}^{(2)}\boldsymbol{\tau}||_1 \quad \text{where}$$

$$g_\gamma(x_i) = \begin{cases} \frac{1}{2}x_i^2, & |x_i| \le \gamma \\ \gamma|x_i| - \frac{1}{2}\gamma^2, & |x_i| > \gamma \end{cases}$$



$$\mathbf{D}^{(1)} = \begin{bmatrix} 1 & -1 & & \\ & 1 & -1 & \\ & & \ddots & \\ & & 1 & -1 \end{bmatrix}, \; \mathbf{D}^{(2)} = \begin{bmatrix} 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & & \ddots & \\ & & & 1 & -2 & 1 \end{bmatrix}$$

Qingsong Wen, Jingkun Gao, Xiaomin Song, Liang Sun, Jian Tan, "RobustTrend: A Huber Loss with a Combined First and Second Order Difference Regularization for Time Series Trend Filtering," in Proc. 28th International Joint Conference on Artificial Intelligence (IJCAI 2019), Macao, China, Aug. 2019.

ADMM for RobustTrend Filter

- Alternating direction method of multipliers (ADMM) for RobustTrend filter
  - Optimization formulation

$$\begin{aligned} \min_{\boldsymbol{\tau}} \quad & g_\gamma(\mathbf{y}-\boldsymbol{\tau}) + ||\mathbf{z}||_1 \\ \text{s.t.} \quad & \mathbf{D}\boldsymbol{\tau} = \mathbf{z} \end{aligned} \qquad \text{where} \quad \mathbf{D} = \begin{bmatrix} \lambda_1 \mathbf{D}^{(1)} \\ \lambda_2 \mathbf{D}^{(2)} \end{bmatrix}$$

  - Updating steps: (Tau−minimization step is not efficient)

$$\boldsymbol{\tau}^{k+1} = \arg\min_{\boldsymbol{\tau}} \left(g_\gamma(\mathbf{y}-\boldsymbol{\tau}) + \frac{\rho}{2}||\mathbf{D}\boldsymbol{\tau} - \mathbf{z}^k + \mathbf{u}^k||_2^2 \right)$$

➡ Not efficient, no closed form

$$\mathbf{z}^{k+1} = \arg\min_{\mathbf{z}} \left(||\mathbf{z}||_1 + \frac{\rho}{2}||\mathbf{D}\boldsymbol{\tau}^{k+1} - \mathbf{z} + \mathbf{u}^k||_2^2 \right)$$

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \mathbf{D}\boldsymbol{\tau}^{k+1} - \mathbf{z}^{k+1}$$

➡ Efficient by soft thresholding

$$\mathbf{z}^{k+1} = S_\rho(\mathbf{D}\boldsymbol{\tau}^{k+1} + \mathbf{u}^k)$$

$$S_\rho(a) = \begin{cases} 0, & |a| \le \rho \\ a - \rho \text{sgn}(a), & |a| > \rho \end{cases}$$

Efficient MM Algorithm for Tau−Minimization

- One-iteration majorization minimization (MM) for Tau−minimization step

$$g_\gamma(\mathbf{x}) \leq \eta_\gamma(\mathbf{x}|\mathbf{x}^k) = \sum_i \eta_\gamma(x_i|x_i^k) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} + C$$

where $\mathbf{x} = \mathbf{y} - \boldsymbol{\tau}$ and $\mathbf{A} = \text{diag}\left(g'_\gamma(x_i^k)\right)\text{diag}^{-1}\left(g_\gamma(x_i^k)\right)$

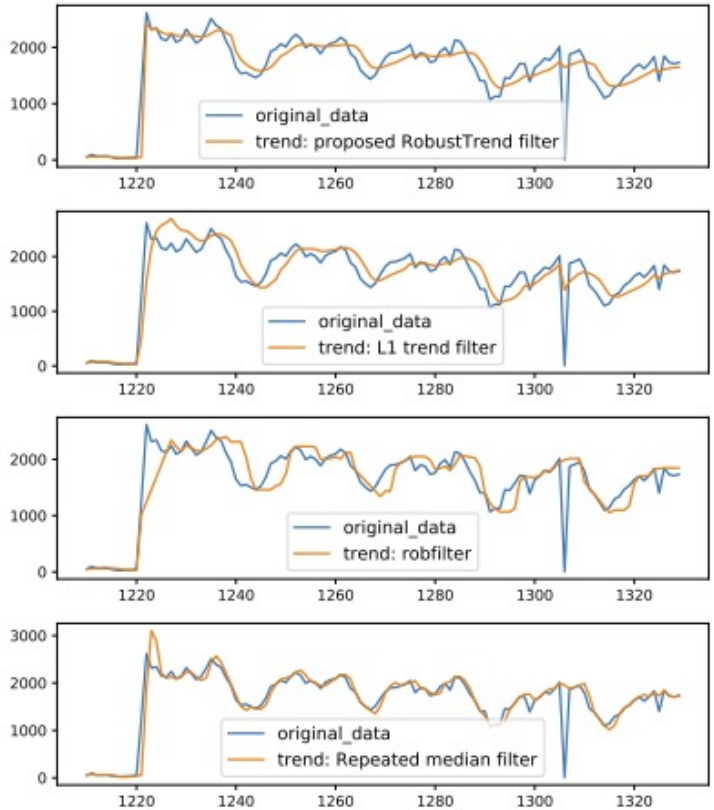
- Now the approximated Tau−minimization step

$$\boldsymbol{\tau}^{k+1} = \mathbf{y} - \rho(\mathbf{A} + \rho\mathbf{D}^T\mathbf{D})^{-1}\mathbf{D}^T(\mathbf{u}^k - \mathbf{z}^k + \mathbf{D}\mathbf{y})$$ ⇒ efficient with closed form

- Theoretical motivation [Eckstein and Bertsekas, 1992]
  - ADMM can still converge when the updating steps are carried out approximately

Experiments: Online Mode on Real-World Data

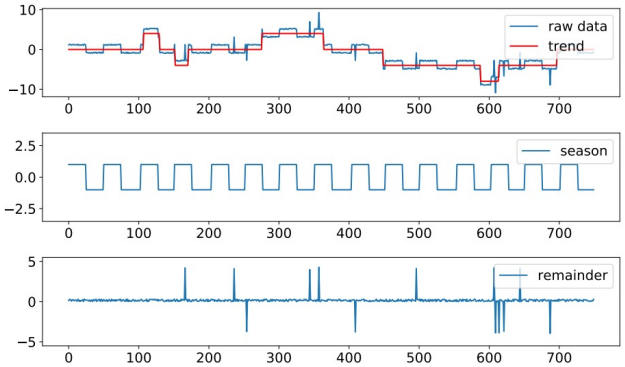
- Compare trend filters of top performance
- Performance highlights
  - L1 trend filter: sensitive to the outliers
  - Robfilter: some delay when trend changes
  - Repeated median filter: overshoots trend estimation
  - **RobustTrend filter**: best tradeoff under outliers and abrupt trend changes



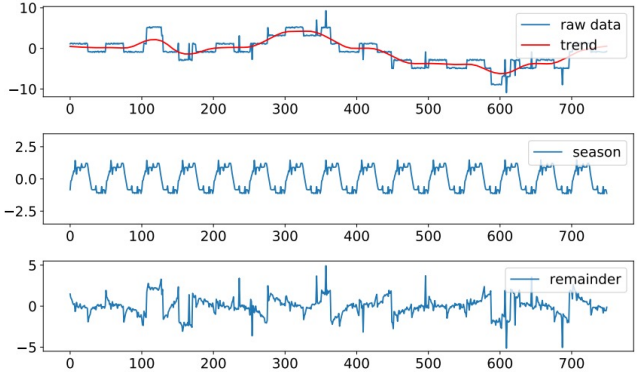
Periodic time series: STL Algorithm

- Seasonal-Trend decomposition using LOESS regression (STL)
  - A sequence of applications of the LOESS smoother
  - LOESS (locally estimated scatterplot smoothing): local regression
  - **Pros**: simple design, fast
  - **Cons**: not robust to trend changes and seasonality shift

$$y_t = \tau_t + s_t + r_t$$



synthetic data with trend changes and seasonality shift

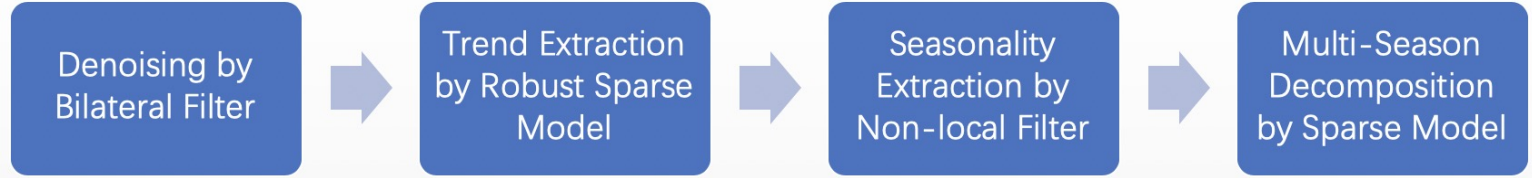


Results of STL decomposition

Cleveland, R. B., Cleveland, W. S., McRae, J. E., & Terpenning, I. J. (1990). STL: A seasonal-trend decomposition procedure based on loess. Journal of Official Statistics, 6(1), 3–33.

Fast RobustSTL Algorithm

- Fast and robust seasonal-trend decomposition: $y_t = \tau_t + \sum_{i=1}^{m} s_{i,t} + r_t$
  - Key idea: sequentially extract components in time series
  - Four blocks: noise removal, trend extraction, seasonality extraction, multi-season decomposition
  - Efficient GADMM for trend extraction and multi-season decomposition: O(*NlogN*)



Qingsong Wen, Jingkun Gao, Xiaomin Song, Liang Sun, Huan Xu, Shenghuo Zhu, "RobustSTL: A Robust Seasonal-Trend Decomposition Algorithm for Long Time Series," in Proc. 33th AAAI Conference on Artificial Intelligence (AAAI 2019), Honolulu, Hawaii, Jan. 2019. (single periodicity version)

Qingsong Wen, Zhe Zhang, Yan Li, Liang Sun, "Fast RobustSTL: Efficient and Robust Seasonal-Trend Decomposition for Time Series with Complex Patterns," in Proc. 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD 2020), San Diego, CA, Aug. 2020. (multiple periodicities and high-speed version)

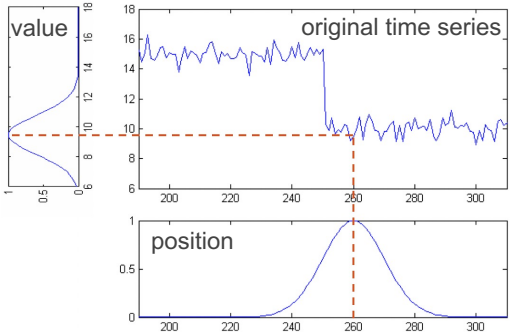
Linxiao Yang, Qingsong Wen, Bo Yang, Liang Sun, "A Robust and Efficient Multi-Scale Seasonal-Trend Decomposition," in Proc. IEEE 46th International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2021), Toronto, Canada, Jun. 2021. (multiple-scale version)

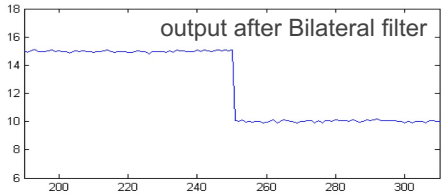
Noise Removal

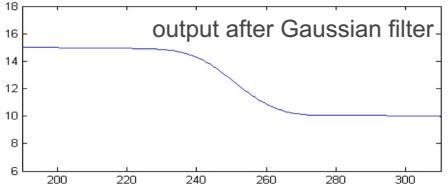
- Bilateral filtering
  - Consider both value and position difference in filtering
  - Smooth time series while preserving abrupt changes

$$y'_t = \sum_{j \in J} w_j^t y_j, \quad J = t, t \pm 1, \cdots, t \pm H$$

$$w_j^t = \frac{1}{z} e^{-\frac{|j-t|^2}{2\delta_d^2}} e^{-\frac{|y_j - y_t|^2}{2\delta_i^2}}$$







Trend Extraction

- Robust sparse model: LAD loss with two L1 regularizations
  - Mitigate season effect by seasonal difference

$$g_t = \nabla_T y'_t = y'_t - y'_{t-T} = \nabla_T \tau_t + \nabla_T s_t + \nabla_T r'_t$$

$$T = \max T_i, i = 1, 2, \cdots, m$$

  - LAD loss: robust to outliers; L1 regs: capture trend changes

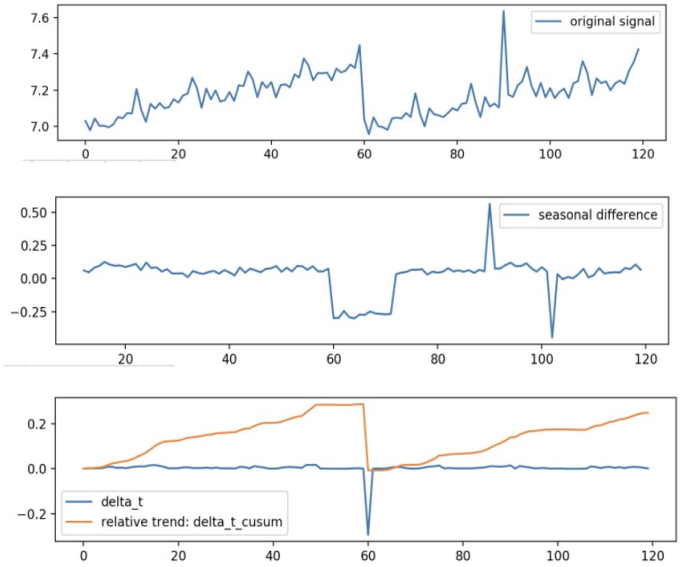
$$\sum_{t=T+1}^{N} |g_t - \sum_{i=0}^{T-1} \nabla\tau_{t-i}| + \lambda_1 \sum_{t=2}^{N} |\nabla\tau_t| + \lambda_2 \sum_{t=3}^{N} |\nabla^2\tau_t|$$

To capture abrupt change: $\nabla\tau_t = \tau_t - \tau_{t-1}$

To capture slow change: $\nabla^2\tau_t = \tau_t - 2\tau_{t-1} + \tau_{t-2}$

  - Finally, recover relative trend by cumulative sum

$$\tilde{\tau}_t^r = \tilde{\tau}_t - \tau_1 = \tilde{\tau}_t - \tilde{\tau}_1 = \begin{cases} 0, & t = 1 \\ \sum_{i=2}^{t} \nabla\tilde{\tau}_i, & t \geq 2 \end{cases}$$



Seasonality Extraction

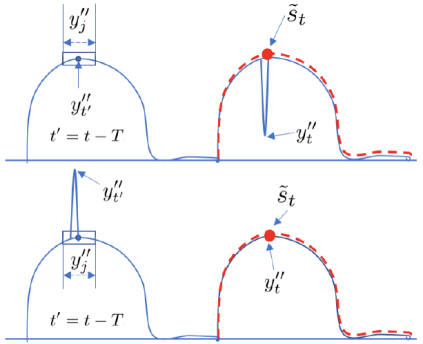
- Weighted non-local seasonal filtering
  - Consider *K* neighborhoods for each season
  - Filter weights based on both value and position difference
  - Robust to outlier and adaptive to season shift

$$\tilde{s}_t = \frac{1}{z}\sum_{i=1}^{m}\alpha_i \sum_{(t_i',j)\in\Omega} w^t_{(t_i',j)} y''_j$$

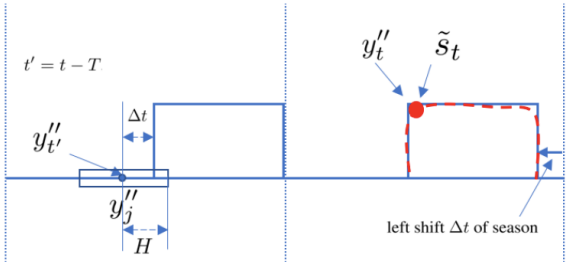
$$w^t_{(t_i',j)} = e^{-\frac{|j-t_i'|^2}{2\delta_d^2} - \frac{|y''_j - y''_{t_i}|^2}{2\delta_i^2}}$$

$$\Omega = \{(t_i', j) | (t_i' = t \pm k \times T_i, j = t_i' \pm h)\}$$

$$k = 1, 2, \cdots, K;\ h = 0, 1, \cdots, H$$



(a) Outlier robustness



(b) Season shift adaptation

Further Multi-Season Decomposition

- Sparse model: MSE loss with three L1 regularizations
  - MSE loss: Outliers have been removed by non-local seasonal filter
  - Regularizations:
    - First two regs: capture abrupt and slow season changes (similar to trend extraction)
    - Last seasonal-wise second-order difference reg: for seasonal-wise stability

$$\underset{\{\mathbf{s}_i|i=1,2,\cdots,m\}}{\arg\min} \frac{1}{2}||\tilde{\tilde{\mathbf{s}}} - \sum_{i=1}^{m}\mathbf{s}_i||_2^2 + \sum_{i=1}^{m}\lambda_{1,i}||\mathbf{D}\mathbf{s}_i||_1 + \sum_{i=1}^{m}\lambda_{2,i}||\mathbf{D}^2\mathbf{s}_i||_1 + \sum_{i=1}^{m}\lambda_{3,i}||\mathbf{D}_{\mathbf{T}_i}^2\mathbf{s}_i||_1,$$

GADMM for Efficient Computation: O(*NlogN*)

- ADMM for trend extraction

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}} L_\rho(\mathbf{x}, \mathbf{y}, \mathbf{u}) = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T(\mathbf{b} + \mathbf{y}_t - \frac{1}{\rho}\mathbf{u}_t)$$ O(N^2)

$$\mathbf{y}_{t+1} = \arg\min_{\mathbf{y}} L_\rho(\mathbf{x}, \mathbf{y}, \mathbf{u}) = S_{1/\rho}(\mathbf{A}\mathbf{x}_{t+1} - \mathbf{b} + \frac{1}{\rho}\mathbf{u}_t)$$ O(N)

$$\mathbf{u}_{t+1} = \arg\min_{\mathbf{u}} L_\rho(\mathbf{x}, \mathbf{y}, \mathbf{u}) = \mathbf{u}_t + \rho(\mathbf{A}\mathbf{x}_{t+1} - \mathbf{y}_{t+1} - \mathbf{b})$$ O(N)

$$\mathbf{A} = \begin{bmatrix} \mathbf{M}_{(N-T)\times(N-1)} \\ \lambda_1 \mathbf{I}_{(N-1)\times(N-1)} \\ \lambda_2 \mathbf{D}_{(N-2)\times(N-1)} \end{bmatrix}, \mathbf{b} = \begin{bmatrix} \mathbf{g}_{(N-T)\times 1} \\ \mathbf{0}_{(2N-3)\times 1} \end{bmatrix}$$

- Similar GADMM can be formulated for multi-season decomposition

GADMM
X-update relaxation for trend extraction

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}} L_\rho(\mathbf{x}, \mathbf{y}, \mathbf{u}) = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T(\mathbf{b} + \mathbf{y}_t - \frac{1}{\rho}\mathbf{u}_t)$$

$$\mathbf{x}_{t+1} \approx \arg\min_{\mathbf{x}} \left(L_\rho(\mathbf{x}, \mathbf{y}, \mathbf{u}) + \underbrace{\|\mathbf{x} - \mathbf{x}_t\|^2_{\hat{\mathbf{H}}}}_{\text{extra proximal term}} \right)$$ O(N^2)

$$\hat{\mathbf{H}} = \mathbf{H} - \mathbf{A}^T\mathbf{A} = \tilde{\mathbf{A}}^T\tilde{\mathbf{A}} - \mathbf{A}^T\mathbf{A} = \hat{\mathbf{M}}^T\hat{\mathbf{M}} + \lambda_2^2\hat{\mathbf{D}}^T\hat{\mathbf{D}}$$

$$\mathbf{x}_{t+1} \approx \mathbf{H}^{-1}\mathbf{v} = \mathbf{F}^{-1}\text{diag}^{-1}(\mathbf{F}\mathbf{h})\mathbf{F}\mathbf{v}$$
$$= \text{IDFT}\left(\text{DFT}(\mathbf{v})./\text{DFT}(\mathbf{h})\right),$$ O(*NlogN*)

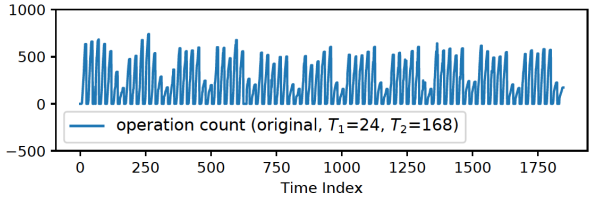
construct circulant matrix

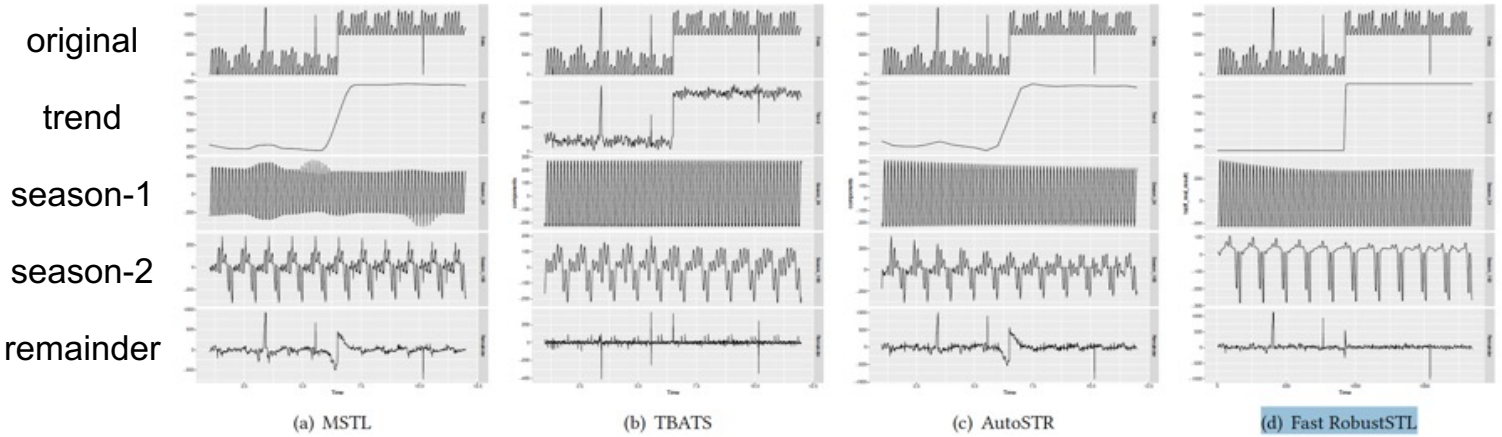
$$\tilde{\mathbf{A}} = \begin{bmatrix} \tilde{\mathbf{M}}_{(N-T)\times(N-1)} \\ \lambda_1 \mathbf{I}_{(N-1)\times(N-1)} \\ \lambda_2 \tilde{\mathbf{D}}_{(N-2)\times(N-1)} \end{bmatrix} \tilde{\mathbf{D}} = \begin{bmatrix} \mathbf{D} \\ \hat{\mathbf{D}} \end{bmatrix} = \begin{bmatrix} 1 & -1 & & & \\ & 1 & -1 & & \\ & & \ddots & & \\ & & & 1 & -1 \\ -1 & & & & 1 \end{bmatrix} \} \mathbf{D}$$

$$\tilde{\mathbf{M}} = \begin{bmatrix} \mathbf{M} \\ \hat{\mathbf{M}} \end{bmatrix} = \begin{bmatrix} 1 & \cdots & 1 & & & & \\ & 1 & \cdots & 1 & & & \\ & & \ddots & & & & \\ & & & 1 & \cdots & 1 & \\ 1 & & & & 1 & \cdots & \\ \ddots & & & & & \ddots & \\ \cdots & 1 & & & & & 1 \end{bmatrix} \} \mathbf{M}$$

Decomposition on Real-World Data

- Real-world data with daily and weekly seasons
  - adding abrupt trend change
  - adding 2 single-point outliers, 1 pattern outlier spanning 1 day





Fast RobustSTL is robust to abrupt trend changes and outliers

Comparisons

- Speed Experiments

N	Metrics	RobustSTL-ADMM	RobustSTL-PDHG	**Fast RobustSTL**
N=1080	Iter	41	310	37
	Time(Sec)	0.142	0.0319	0.0109
	SpeedUp		4.45x	**13.0x**
N=2160	Iter	48	319	40
	Time(Sec)	1.11	0.0571	0.0295
	SpeedUp		19.4x	**37.6x**
N=4320	Iter	68	602	37
	Time(Sec)	5.98	0.191	0.0988
	SpeedUp		31.3x	**60.5x**
N=8640	Iter	92	1377	53
	Time(Sec)	36.7	0.62	0.254
	SpeedUp		59.1x	**144x**

Algorithm	Time (N=4320)
AutoSTR	4441 seconds
TBATS	60 seconds
(M)STL	0.2 second
RobustSTL	6 seconds
Fast RobustSTL	**0.1 second**

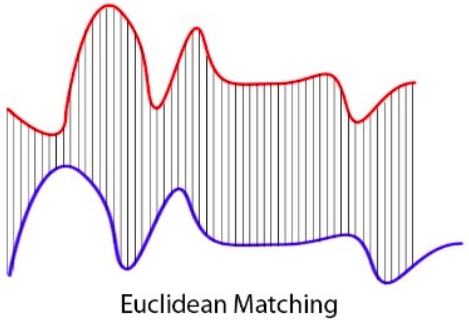
Fast RobustSTL is more efficient than others, and has about 10x~100x speedup compared with RobustSTL.

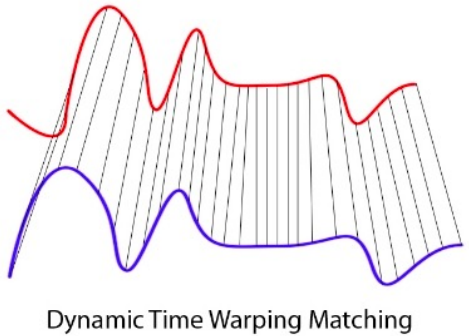
- Algorithm Comparisons

Algorithm	Robust to outlier	Robust to seasonal shift	Robust to trend changes	Support multiple periods	Efficient computation
ARIMA/SEATS	×	×	×	×	×
SSA	×	×	×	×	×
TBATS	×	×	✓	✓	×
STL	×	×	×	×	✓
MSTL	×	×	×	✓	×
STR	✓	✓	×	✓	×
Fast RobustSTL	✓	✓	✓	✓	✓

How to Measure Similarity between Time Series

- Similarity measure is a core component in many tasks involving time series, e.g., time series classification, clustering and retrieval
- Different types of time series similarity
  - Euclidean distance
  - Feature-based measures (e.g., Fourier coefficients)
  - Model-based measures (e.g., auto-regressive)
  - Elastic measures (e.g., dynamic time warping and edit distance): introduce warping/aligning in the temporal domain
- Superior of DTW
  - A classical approach still among the top choices for time series similarity measure
  - Empirical studies on 38 datasets for 9 similarity measures in time series classification confirm the consistent superior of DTW
  - 1-NN with DTW beats most advanced time series classifiers





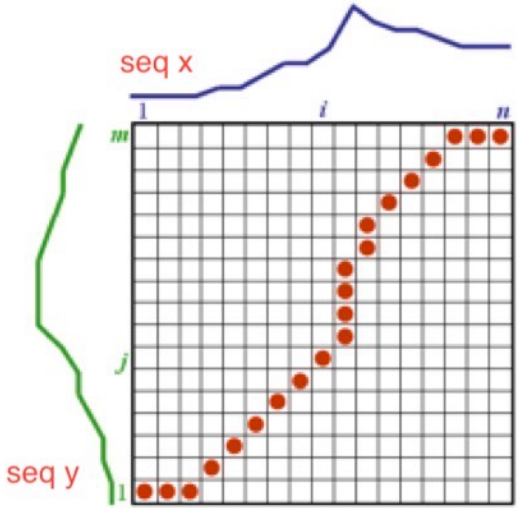
Joan Serra and Josep L. Arcos, An empirical evaluation of similarity measures for time series classification, Knowlege-Based Systems, 67:305–314, 2014.

Dynamic Time Warping (DTW)

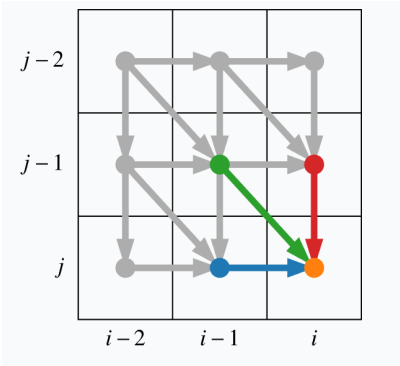
- Dynamic Time Warping in the optimization form:

$$D(x, y) = \min \sum_{t} dist(x_t, y_{\phi(t)})$$

- Time warp function $\phi(t)$ satisfies
  - Monotonicity
  - Continuity
  - Boundary
- How to compute DTW using dynamic programming?



How time warp function changes the similarity computation



$$D(i,j) = Dist(i,j) + \min[D(i-1,j), D(i,j-1), D(i-1,j-1)]$$

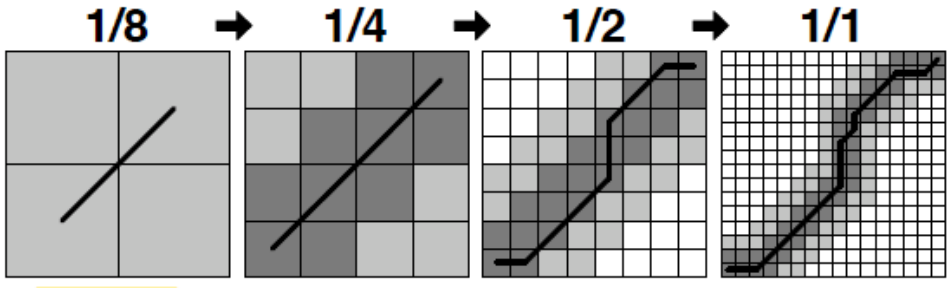
where $D(i,j)$ is the DTW distance between $x_{1:i}$ and $y_{1:j}$

$dist(i,j)$ is the distance between x_i and y_j

Resource for DTW: https://rtavenar.github.io/blog/dtw.html

FastDTW

- The time complexity of standard DTW is $O(n^2)$
- How to reduce the computational cost significantly?
- FastDTW: linear time and space complexity $O(nr)$
- Key ideas of FastDTW
  - It estimates the warp function in a multi-resolution framework
  - The warp function estimated at the current resolution is searched in a constrained space specified by the warping estimate from the low-resolution



It repeatedly uses low-resolution warping estimate as the constraint to generate the warping estimate in the current resolution

Stan Salvador, and Philip Chan. "FastDTW: Toward accurate dynamic time warping in linear time and space." Intelligent Data Analysis 11.5 (2007): 561-580.

RobustDTW: Key Ideas

- Challenges of DTW:
  - Noises and outliers bias the similarity
  - Block of missing values occurs in many real-world applications
  - How to perform DTW efficiently?
- Key ideas of RobustDTW:
  - It not only estimates the time warp function, also estimates the trend to handle noises and outliers
  - Trend estimation is achieved via graph detrending
  - An alternating framework is applied
- Advantages of RobustDTW:
  - Robust to noises and outliers
  - Efficient implementation thanks to the multi-resolution framework
  - Can handle (block) missing values effectively

Xiaomin Song, Qingsong Wen, Yan Li, and Liang Sun, "Robust Time Series Dissimilarity Measure for Outlier Detection and Periodicity Detection," in Proc. 31st ACM International Conference on Information and Knowledge Management (CIKM 2022), Atlanta, GA, Oct. 2022.

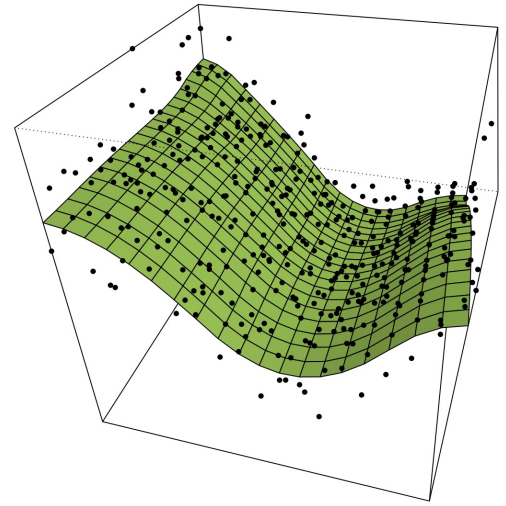
Graph Detrending for Trend Estimation

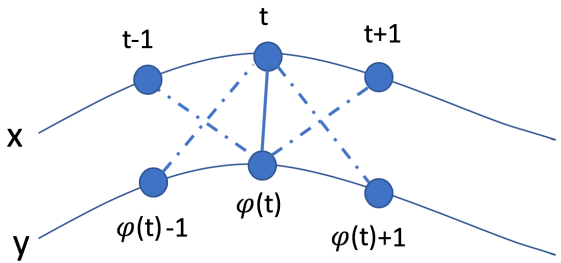
- Graph smoothing: given a graph $G = (V, E)$ with vertices denoted $V = \{1, \cdots, n\}$, we assume $y_i = \mu_i + \epsilon_i, i = 1, \cdots, n$, where ϵ_i assumed to have zero mean. Our goal is to estimate μ_i, assumed to be smooth w.r.t. edges E
- Graph trend filtering solves

$$\min_{\beta} \left|\left|y - \beta\right|\right|_2^2 + \lambda \left|\left|\Delta^{(k+1)}\beta\right|\right|_1$$

$\Delta^{(k+1)}$ is a graph difference operator of order $k+1$ over G

- Advantages of trend filtering
  - Computational fast
  - Locally adaptive
- How to construct graph in DTW:
  - Each time point in time series corresponds to a vertice
  - In each time series, each time is connected to its previous and next time points
  - Cross time series connections are determined by the time warp function and extensions



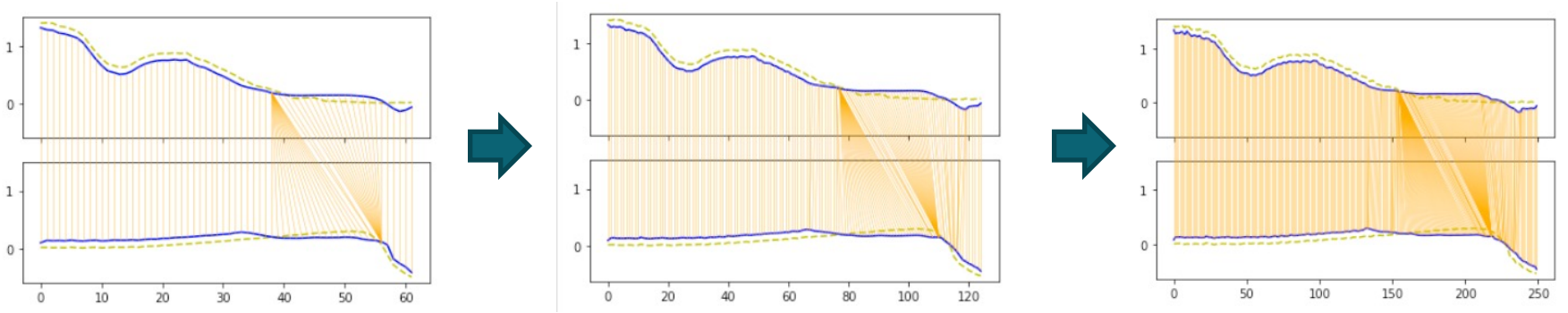


Yu-Xiang Wang, James Sharpnack, Alexander J. Smola, Ryan J. Tibshirani, Trend Filtering on Graphs, Journal of Machine Learning Research, 17(105):1−41, 2016.

Detailed Procedure of RobustDTW

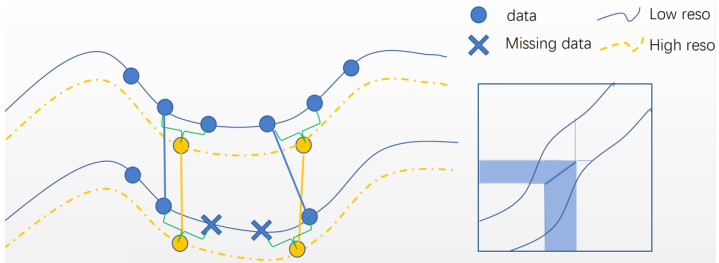
- Step 1. Preprocess: detrend and separately
- Step 2. Learn the time warp function estimate
- Step 3. Construct a graph based on the learned time warp function, and perform graph detrending
- Step 4. Repeat Steps 2~3 until convergence

- Put the alternating procedure in a multi-resolution framework
- The time warp function is estimated based on that in the lower-resolution

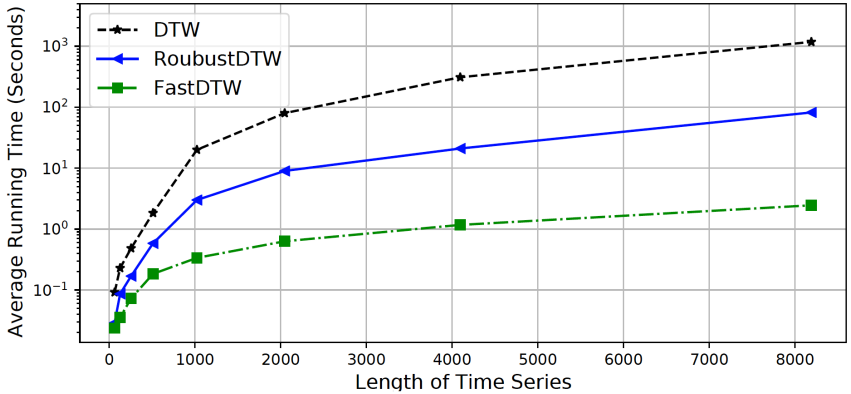


RobustDTW: Efficiency and Missing Value Processing

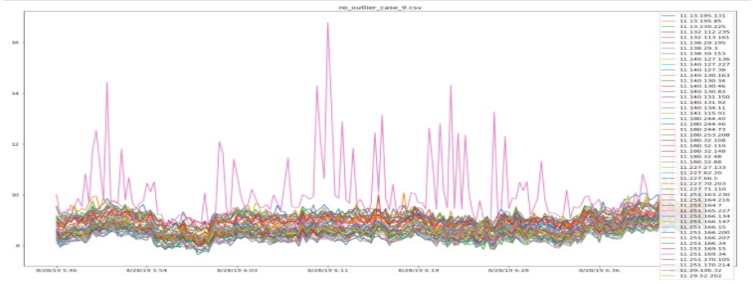
- Block missing data can processed adequately in the low-resolution representation
- When mapping into the high-resolution, the corresponding segments are excluded in further computation

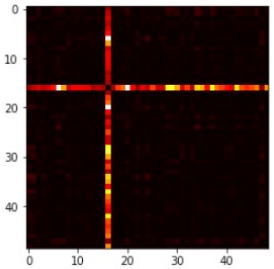


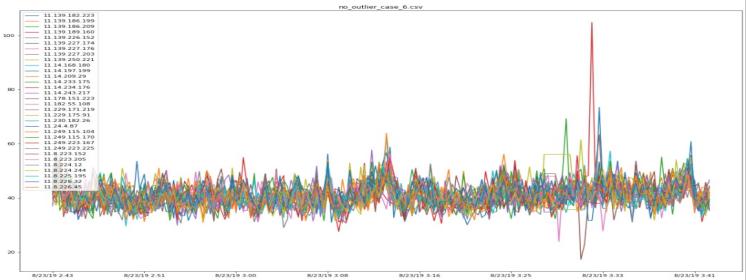
Running time comparison with different time series lengths

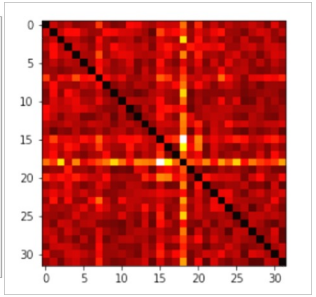


Applications of RobustDTW: Time Series Outlier Detection









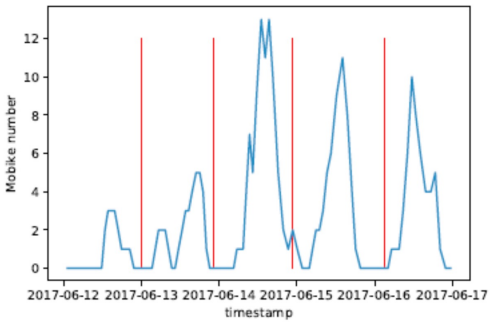
Comparison of AUC scores of outlier detection via local outlier factor (LOF) algorithm with different similarity measures and noise conditions

Dataset	Noise level	ED	DTW	FastDTW	**RobustDTW**
RT	raw data	0.887	0.986	0.949	**0.997**
	+ dips	0.811	0.965	0.918	**0.996**
	+ spikes	0.362	0.939	0.775	**0.972**
	+ spikes & dips	0.317	0.580	0.655	**0.969**
NetSpd	raw data	0.674	0.982	0.917	**0.988**
	+ dips	0.635	0.849	0.649	**0.982**
	+ spikes	0.642	0.915	0.874	**0.975**
	+ spikes & dips	0.508	0.627	0.616	**0.938**

- RobustDTW can effectively identify the outlier machine (represented by a time series) with noises and outliers in real-world

RobustDTW: Robust Periodicity Detection

- RobustDTW can be applied in periodicity detection when the periodic length L is known
- Slice the input time series into segments with length L
  - If the pairwise similarity with adjacent segments computed by RobustDTW is above a threshold, the input time series is predicted as periodic



Periodicity detection on 200 service monitor time series from AlibabaCloud, where 50% of ts are daily periodic

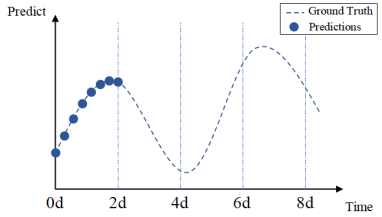
Methods		Precision	Recall	F1
ACF		0.960	0.701	0.810
AUTOPERIOD		**0.980**	0.715	0.827
RobustPeriod |		0.920	0.902	0.911
Slicing	ED	0.919	0.800	0.855
	DTW	0.911	0.930	0.920
	FastDTW	0.873	0.970	0.919
	RobustDTW	0.951	**0.980**	**0.966**

Outline

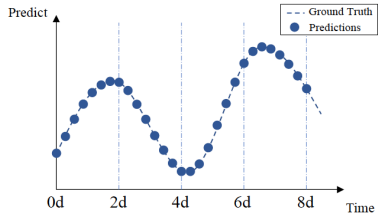
- ❑ **Introduction**
- ❑ **Preliminaries**
- ❑ **Robust Time Series Processing Blocks**
- ➢ ***Robust Time Series Applications and Practices***
  - Forecasting: Tree Models, Deep Ensemble, Transformers, etc.
  - Autoscaling: Query Modeling, Scaling Decision, etc.
  - Anomaly Detection: Decomposition Model, Deep State Space Model, Transformers, etc.
  - Fault Cause Localization: Rule Set Learning, Root Cause Analysis, etc.

Forecasting: Background

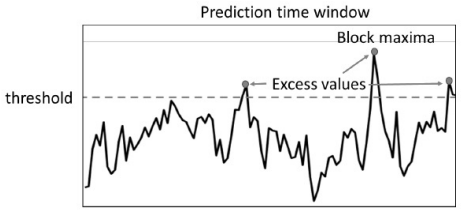
- Common Forecasting Types
  - **Short-term** forecasting: predict the near future
  - **Long-term** forecasting: predict the future with an extended period
  - **Extreme value** forecasting: predict the extreme values
  - **Point or Probabilistic** forecasting: predict point value or interval/probability distribution
- Challenges:
  - Accuracy, robustness, insufficient data
- Models:
  - Traditional: Statistical (ARIMA, ETS, Prophet)
  - *Ensemble : Tree, MLP*
  - *Deep Models: CNN, RNN, Transformers*



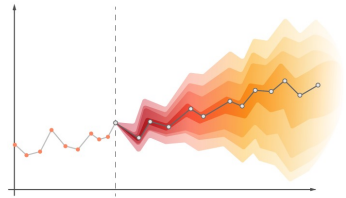
Short term forecasting



Long term forecasting



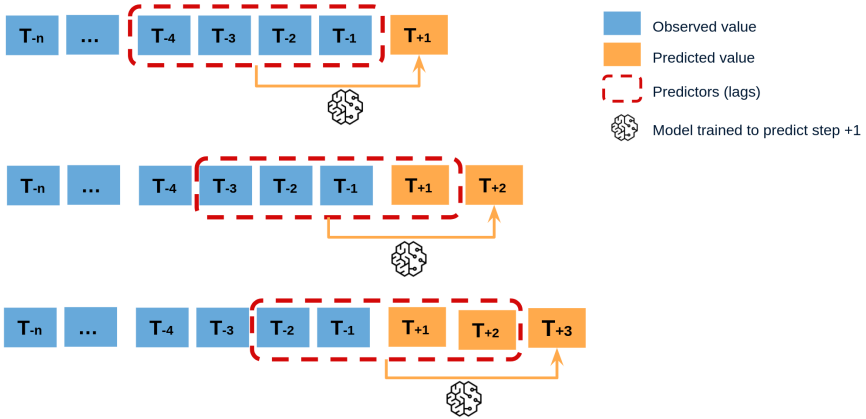
Extreme value forecasting


Probabilistic forecasting

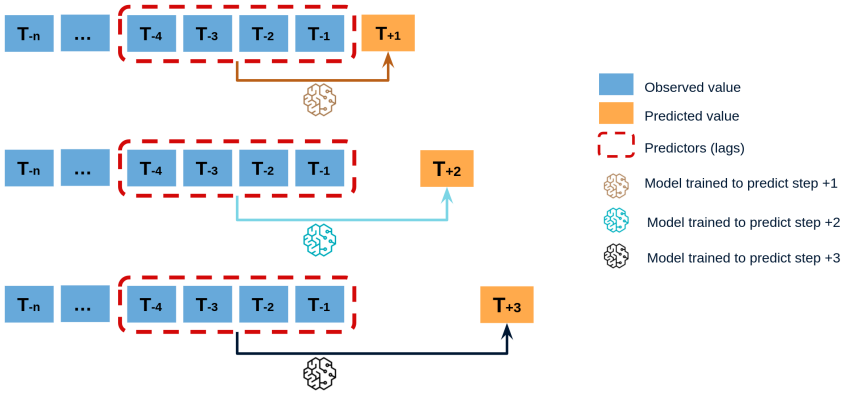
Forecasting: Ensemble (Tree Models)

- Tree Models: an ensemble of weak prediction models
  - XGBoost, LightGBM, CatBoost

Recursive forecasting with single tree based model



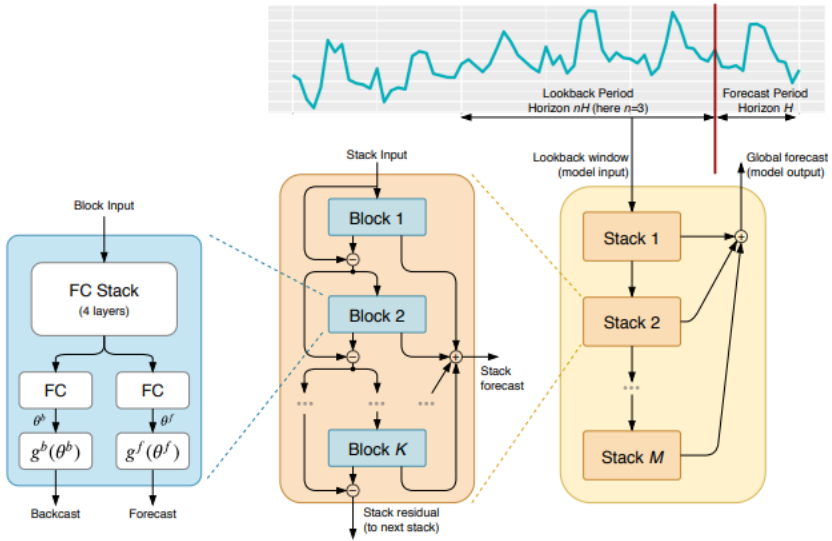
Direct forecasting with multiple tree based model for different steps



Amat Rodrigo, J., & Escobar Ortiz, J. skforecast (Version 0.5.0) [Computer software]

Forecasting: Deep Ensemble (MLP based Models)

- N-BEATS
  - Doubly residual stackings with forward and backward residual links
  - Forecasts are aggregated in hierarchical way
  - Trend and seasonal models for interpretability
  - Ensemble: e.g., 18 to 180 models
    - Fit on different metrics: sMAPE, MASE, MAPE
    - Train on input windows of different length
    - Train with different random initializations



Oreshkin, B. N., Carpov, D., Chapados, N., & Bengio, Y. “N-BEATS: Neural basis expansion analysis for interpretable time series forecasting.” ICLR 2020.

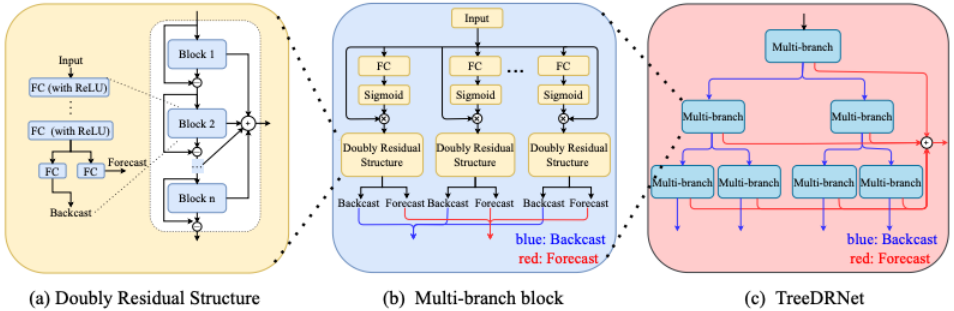
Forecasting: Deep Ensemble (MLP based Models)

- TreeDRNet
  - Doubly residual link motivated by robust iterative reweighted algorithm
  - Ensemble of models and tree-based aggregation motivated by Kolmogorlv-Arnold representation theorem
  - Efficient, accurate, and robust against input anomalies/attacks

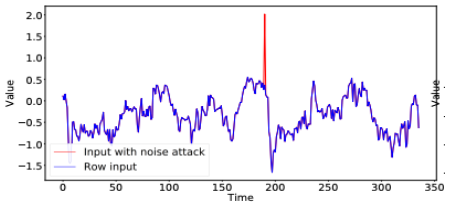
$$\delta y_k = y - \sum_{j=1}^{k-1} z_j, \quad \underbrace{x_k = f(x_{k-1}; \delta y_k) + x_{k-1}}_{\text{Backcast + Skip Link}},$$

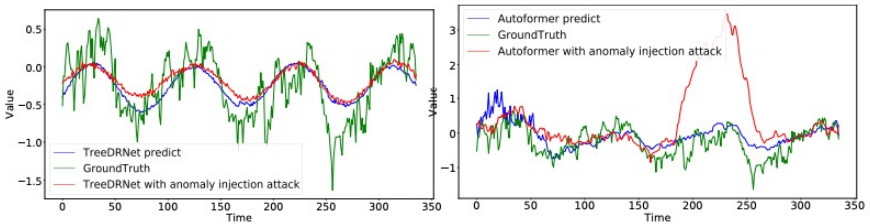
$$\Delta\beta_k = g(x_k, \delta y_k), \quad \underbrace{z_k = \langle \Delta\beta_k, x_k \rangle}_{\text{Forecast Link}},$$

$$\underbrace{h(\boldsymbol{x}) = \sum_{k=0}^{2d} \Phi_k \left(\underbrace{\sum_{p=1}^{m} \phi_{k,p} \left(\underbrace{\boldsymbol{x} \circ \boldsymbol{m}_{k,p}}_{\text{Feature Selection}} \right)}_{\text{Model Ensemble}} \right)}_{\text{Tree Structure}},$$



(a) Doubly Residual Structure (b) Multi-branch block (c) TreeDRNet

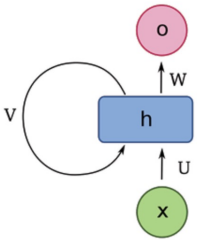




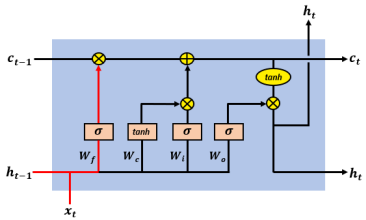
Tian Zhou, Jianqing Zhu, Xue Wang, Ziqing Ma, Qingsong Wen, Liang Sun, Rong Jin, "TreeDRNet: A Robust Deep Model for Long Term Time Series Forecasting", arXiv:2206.12106.

Forecasting: RNN based Models

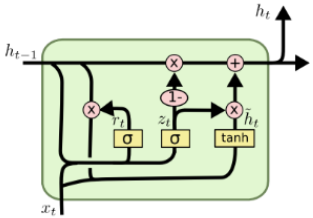
- Recurrent Neural Networks
  - From RNN to LSTM/GRU: control information flow by gates, mitigating vanishing gradient problem
  - DeepAR: time series probabilistic forecasting through autoregressive recurrent network



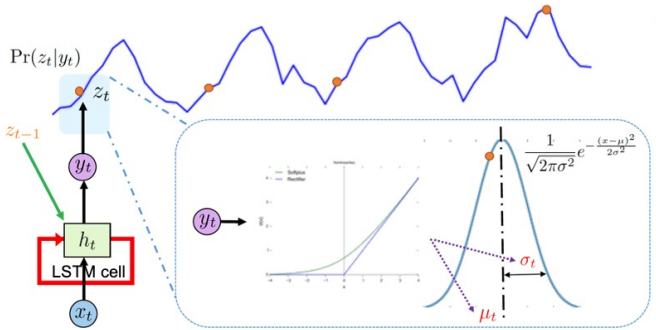
Vanilla RNN



LSTM



GRU

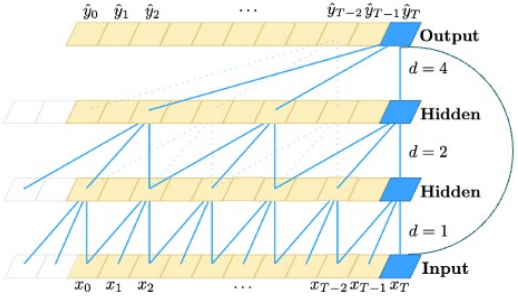


DeepAR

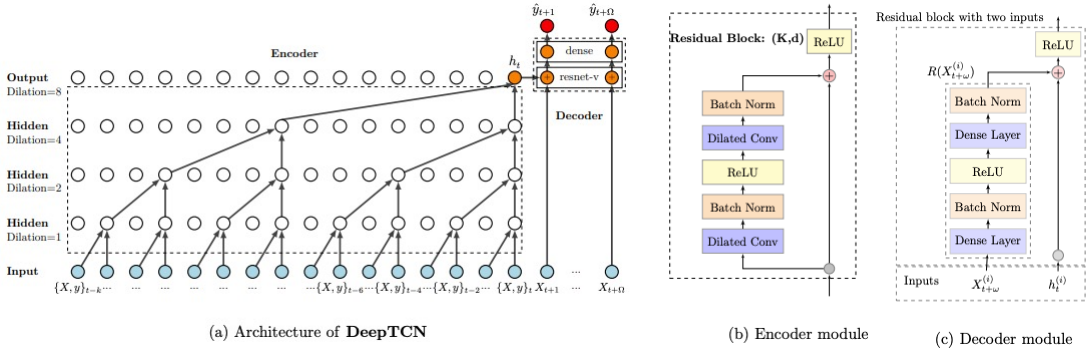
Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation, 1997.
Chung, Junyoung, et al. "Empirical evaluation of gated recurrent neural networks on sequence modeling." arXiv preprint arXiv:1412.3555, 2014.
Salinas, D., Flunkert, V., Gasthaus, J., & Januschowski, T. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. International Journal of Forecasting, 2020.

Forecasting: CNN based Models

- TCN (Temporal Convolutional Networks)
  - 1D dilated causal CNN
  - Dilated convolutional layers for large receptive field
  - Skip connection for raw info flow like ResNet
  - Empirically good performance and fast training



- Deep TCN
  - Encoder and decoder with TCN
  - Probabilistic forecasting under both parametric (Gaussian) and non-parametric (quantile) settings

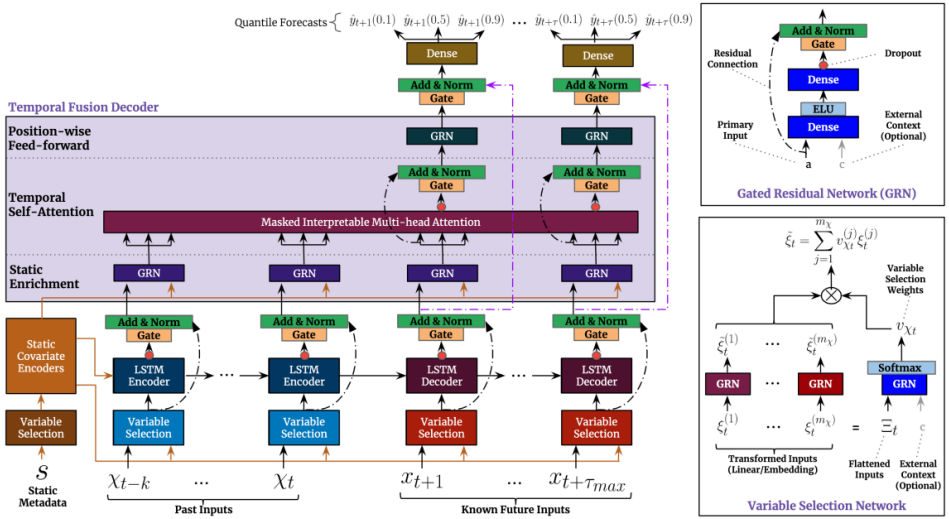


(a) Architecture of **DeepTCN** (b) Encoder module (c) Decoder module

Bai, Shaojie, J. Zico Kolter, and Vladlen Koltun. "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling." *arXiv preprint arXiv:1803.01271, 2018.*
Chen, Yitian, Yanfei Kang, Yixiong Chen, and Zizhuo Wang. “Probabilistic forecasting with temporal convolutional neural network.” Neurocomputing, 2020.

Forecasting with Transformer: TFT

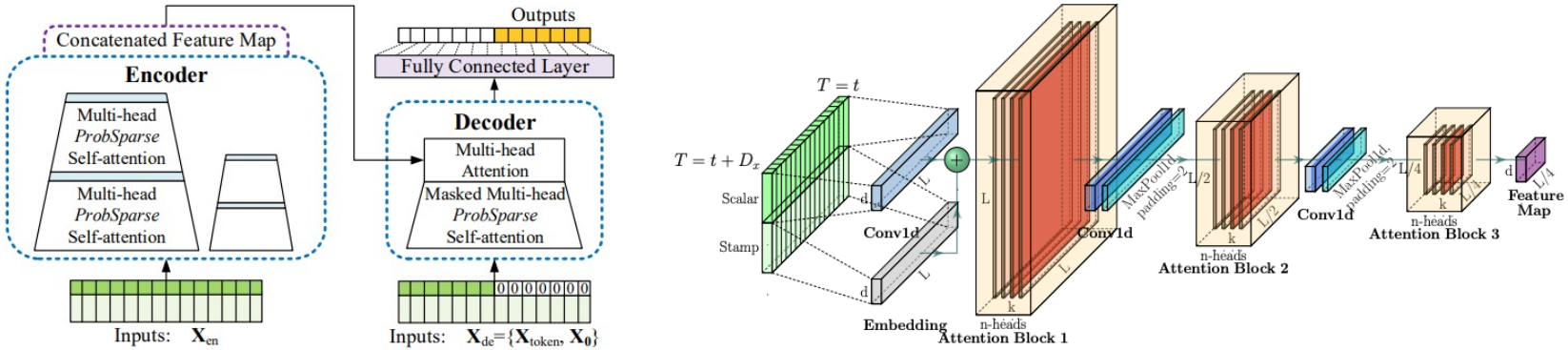
- TFT (Temporal Fusion Transformer)
  - Variable selection network: select most salient features
  - Gated residual network: efficient information flow with skip connections and gating layers
  - Multi-scales network: recurrent layers for local processing, interpretable self-attention layers for long-term dependencies



Lim, B., Arık, S. Ö., Loeff, N., & Pfister, T. "Temporal fusion transformers for interpretable multi-horizon time series forecasting." International Journal of Forecasting, 2021.

Forecasting with Transformer: Informer

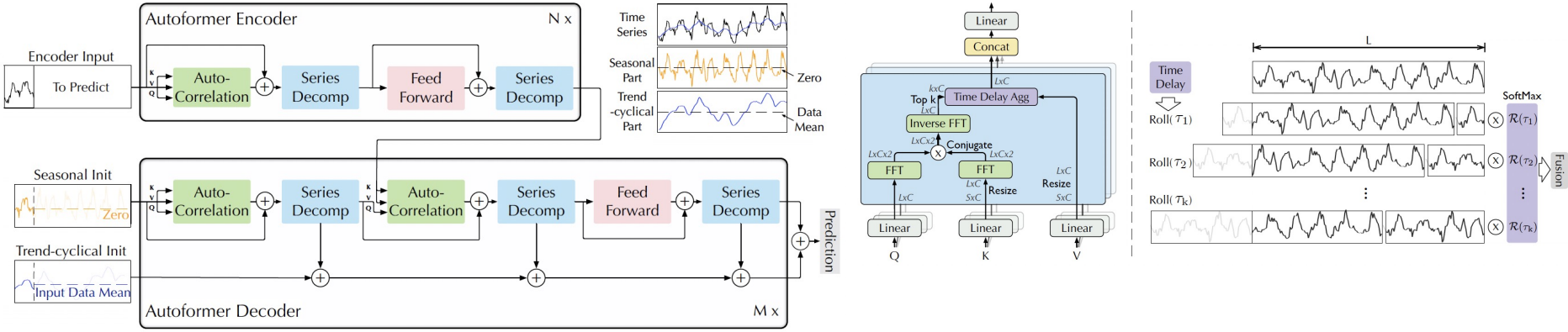
- Informer
  - ProbSparse self-attention for efficient and robust attention mechanism
  - Self-attention distilling: extract dominating attention and reducing the network size
  - Generative style decoder: produce long sequence forecasts with only one forward step, avoiding cumulative error spreading during inference



Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H. and Zhang, W., "Informer: Beyond efficient transformer for long sequence time-series forecasting." AAAI, 2021.

Forecasting with Transformer: Autoformer

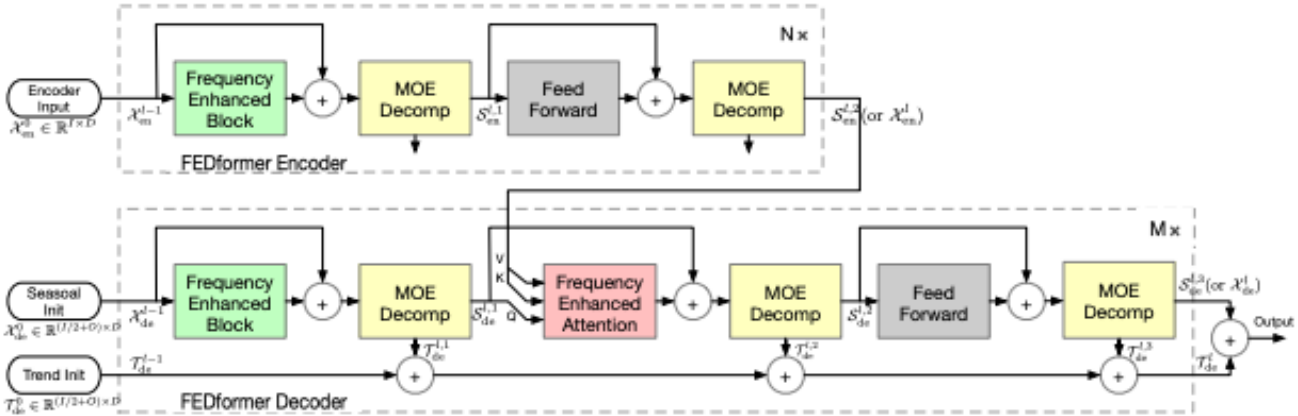
- Autoformer: Transformer with auto-correlation mechanism
  - Decomposition architecture to disentangle complex temporal patterns (seasonality, trend)
  - Auto-correlation instead point-wise self-Attention to discover period-based dependencies, aggregate similar sub-processes from different periods, and reduce complexity.



Wu, H., Xu, J., Wang, J. and Long, M., “Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting”, NeurIPS 2021.

Forecasting with Transformer: FEDformer

- FEDformer: Frequency enhanced decomposed Transformer
  - Mixture of Experts seasonal-trend decomposition: to better capture global properties in time series
  - Efficient and robust frequency domain processing: to capture important structures in time series
    - Frequency Enhanced Block: substitute self-attention
    - Frequency Enhanced Attention: substitute cross-attention



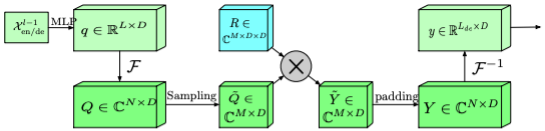


Figure 3. Frequency Enhanced Block with Fourier transform (FEB-f) structure.

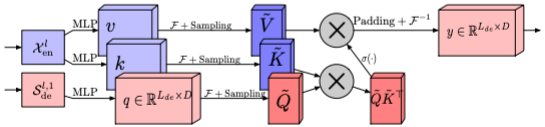


Figure 4. Frequency Enhanced Attention with Fourier transform (FEA-f) structure, $\sigma(\cdot)$ is the activation function.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, Rong Jin, "FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting," ICML 2022.

Forecasting with Transformer: FEDformer

SOTA result of FEDformer in six benchmark datasets

Table 2. Multivariate long-term series forecasting results on six datasets with input length $I = 96$ and prediction length $O \in \{96, 192, 336, 720\}$ (For ILI dataset, we use input length $I = 36$ and prediction length $O \in \{24, 36, 48, 60\}$). A lower MSE indicates better performance, and the best results are highlighted in bold.

Methods	Metric	ETTm2				Electricity				Exchange				Traffic				Weather				ILI			
		96	192	336	720	96	192	336	720	96	192	336	720	96	192	336	720	96	192	336	720	24	36	48	60
FEDformer-f	MSE	**0.203**	**0.269**	**0.325**	**0.421**	0.193	0.201	0.214	0.246	0.148	0.271	0.460	1.195	0.587	0.604	0.621	0.626	**0.217**	**0.276**	**0.339**	**0.403**	3.228	2.679	2.622	2.857
	MAE	**0.287**	**0.328**	**0.366**	**0.415**	0.308	0.315	0.329	0.355	0.278	0.380	0.500	0.841	0.366	0.373	0.383	0.382	**0.296**	**0.336**	**0.380**	**0.428**	1.260	1.080	1.078	1.157
FEDformer-w	MSE	0.204	0.316	0.359	0.433	**0.183**	**0.195**	**0.212**	**0.231**	**0.139**	**0.256**	**0.426**	**1.090**	**0.562**	**0.562**	**0.570**	**0.596**	0.227	0.295	0.381	0.424	**2.203**	**2.272**	**2.209**	**2.545**
	MAE	0.288	0.363	0.387	0.432	**0.297**	**0.308**	**0.313**	**0.343**	**0.276**	**0.369**	**0.464**	**0.800**	**0.349**	**0.346**	**0.323**	**0.368**	0.304	0.363	0.416	0.434	**0.963**	**0.976**	**0.981**	**1.061**
Autoformer	MSE	0.255	0.281	0.339	0.422	0.201	0.222	0.231	0.254	0.197	0.300	0.509	1.447	0.613	0.616	0.622	0.660	0.266	0.307	0.359	0.419	3.483	3.103	2.669	2.770
	MAE	0.339	0.340	0.372	0.419	0.317	0.334	0.338	0.361	0.323	0.369	0.524	0.941	0.388	0.382	0.337	0.408	0.336	0.367	0.395	0.428	1.287	1.148	1.085	1.125
Informer	MSE	0.365	0.533	1.363	3.379	0.274	0.296	0.300	0.373	0.847	1.204	1.672	2.478	0.719	0.696	0.777	0.864	0.300	0.598	0.578	1.059	5.764	4.755	4.763	5.264
	MAE	0.453	0.563	0.887	1.338	0.368	0.386	0.394	0.439	0.752	0.895	1.036	1.310	0.391	0.379	0.420	0.472	0.384	0.544	0.523	0.741	1.677	1.467	1.469	1.564
LogTrans	MSE	0.768	0.989	1.334	3.048	0.258	0.266	0.280	0.283	0.968	1.040	1.659	1.941	0.684	0.685	0.7337	0.717	0.458	0.658	0.797	0.869	4.480	4.799	4.800	5.278
	MAE	0.642	0.757	0.872	1.328	0.357	0.368	0.380	0.376	0.812	0.851	1.081	1.127	0.384	0.390	0.408	0.396	0.490	0.589	0.652	0.675	1.444	1.467	1.468	1.560
Reformer	MSE	0.658	1.078	1.549	2.631	0.312	0.348	0.350	0.340	1.065	1.188	1.357	1.510	0.732	0.733	0.742	0.755	0.689	0.752	0.639	1.130	4.400	4.783	4.832	4.882
	MAE	0.619	0.827	0.972	1.242	0.402	0.433	0.433	0.420	0.829	0.906	0.976	1.016	0.423	0.420	0.420	423	0.596	0.638	0.596	0.792	1.382	1.448	1.465	1.483

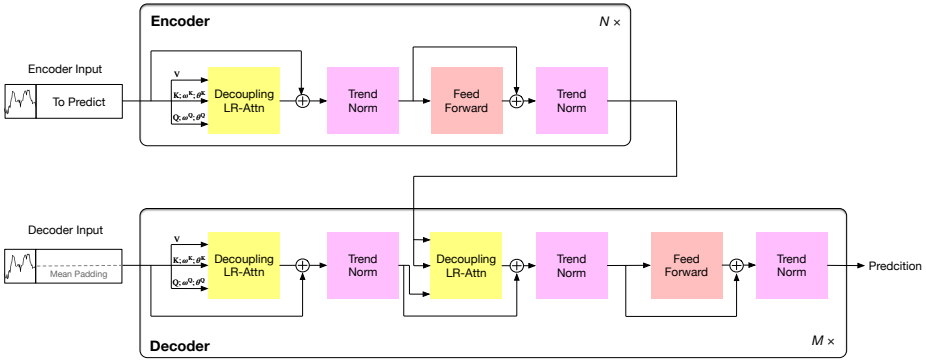
Linear complexity of FEDformer

Table 1. Complexity analysis of different forecasting models.

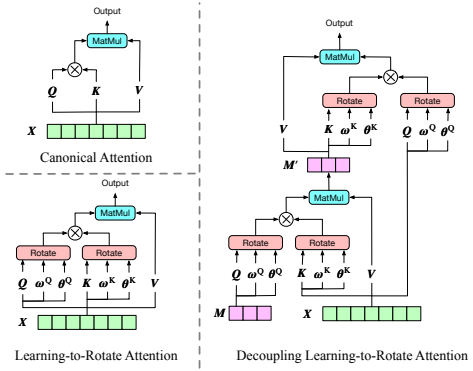
Methods	Training		Testing
	Time	Memory	Steps
FEDformer	$\mathcal{O}(L)$	$\mathcal{O}(L)$	1
Autoformer	$\mathcal{O}(L \log L)$	$\mathcal{O}(L \log L)$	1
Informer	$\mathcal{O}(L \log L)$	$\mathcal{O}(L \log L)$	1
Transformer	$\mathcal{O}\left(L^2\right)$	$\mathcal{O}\left(L^2\right)$	L
LogTrans	$\mathcal{O}(L \log L)$	$\mathcal{O}\left(L^2\right)$	1
Reformer	$\mathcal{O}(L \log L)$	$\mathcal{O}(L \log L)$	L
LSTM	$\mathcal{O}(L)$	$\mathcal{O}(L)$	L

Forecasting with Transformer: Quatformer

- Quatformer
  - Learning-to-Rotate Attention: modeling complicated periodical patterns by quaternions
  - Trend Normalization: modeling slowly varying trend
  - Decoupling LR-Attn: reducing complexity from O(N^2) to O(2cN)



Quatformer Architecture



Learning-to-Rotate Attention

$$\frac{\gamma}{\sigma} \odot (\mathcal{X} - \text{MovingAvg}(\mathcal{X})) + \mathcal{T},$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\mathcal{X}_i - \mu)^2}, \quad \mu = \frac{1}{N}\sum_{i=1}^{N}\mathcal{X}_i,$$

$$\mathcal{T} = \sum_{i=0}^{p} \boldsymbol{\beta}_i \text{pos}^i, \quad \text{pos} = [0, 1, 2, \cdots, N-1]^\top / N.$$

Trend Normalization

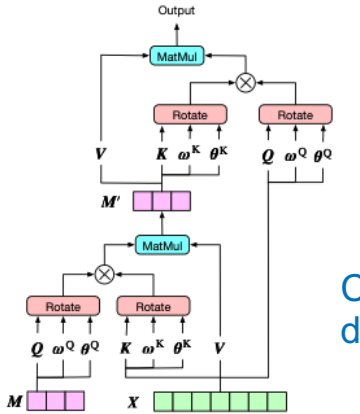
Weiqi Chen, Wenwei Wang, Bingqing Peng, Qingsong Wen, Tian Zhou, Liang Sun, "Learning to Rotate: Quaternion Transformer for Complicated Periodical Time Series Forecasting", KDD 2022.

Forecasting with Transformer: Quatformer

- Quatformer: Decoupling LR-Attn, Performance
  - LRA's complexity is $O(N^2)$
  - Decoupling LR-Attn introduces a momentum-updated c-length latent series $\mathcal{M} \in \mathbb{R}^{c \times d}$, and decouple $\mathcal{H} = \text{LR-Attn}(\mathcal{X}, \mathcal{Y})$ into

$$\mathcal{H} = \text{LR-Attn}(\mathcal{X}, \mathcal{M}') \in \mathbb{R}^{N \times d},$$

$$\mathcal{M}' = \text{LR-Attn}(\mathcal{M}, \mathcal{Y}) \in \mathbb{R}^{c \times d}.$$



Decoupling Learning-to-Rotate Attention

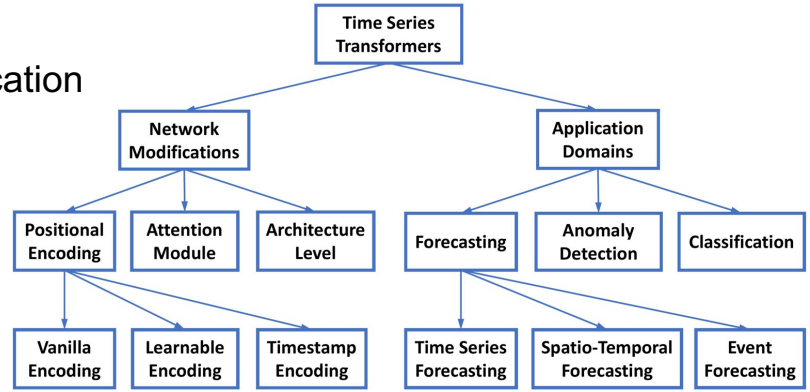
Complexity is decreased to $O(2cN)$.

Performance of Quatformer

Models		Quatformer		Quatformer†		Autoformer	
		MSE	MAE	MSE	MAE	MSE	MAE
ETTh	96	**0.403**	**0.434**	0.426	0.450	0.442	0.451
	192	**0.444**	**0.463**	0.453	0.466	0.500	0.482
	336	**0.452**	**0.455**	0.464	0.474	0.512	0.492
	720	0.477	0.490	**0.474**	**0.487**	0.514	0.512
ETTm	96	0.220	0.301	**0.217**	**0.297**	0.247	0.325
	192	0.279	0.333	**0.269**	**0.329**	0.278	0.335
	336	0.331	**0.354**	**0.330**	0.366	0.336	0.370
	720	**0.422**	**0.413**	0.433	0.428	0.439	0.435
Weather	96	**0.211**	**0.279**	0.213	0.287	0.259	0.332
	192	**0.263**	**0.325**	0.265	0.326	0.300	0.359
	336	**0.310**	**0.344**	0.315	0.354	0.364	0.401
	720	**0.381**	**0.374**	0.382	0.378	0.439	0.440
Exchange	96	**0.147**	**0.274**	0.148	0.276	0.154	0.284
	192	**0.254**	**0.364**	0.255	0.365	0.272	0.381
	336	0.427	0.481	**0.425**	**0.480**	0.461	0.509
	720	**0.974**	**0.751**	1.095	0.800	1.100	0.813
Traffic	96	0.618	**0.384**	**0.617**	0.387	0.636	0.397
	192	0.619	0.384	**0.600**	**0.367**	0.618	0.381
	336	0.622	**0.384**	**0.618**	0.385	0.626	0.388
	720	0.629	0.383	**0.616**	**0.379**	0.653	0.400
Electricity	96	**0.197**	**0.308**	0.200	0.311	0.203	0.318
	192	**0.205**	**0.302**	0.216	0.330	0.233	0.338
	336	**0.220**	**0.329**	0.228	0.343	0.259	0.359
	720	0.245	0.350	**0.242**	**0.349**	0.255	0.361

Forecasting with Transformer: Summary

- Taxonomy of Transformers in Time Series
  - Network modifications
  - Applications: forecasting, anomaly detection, classification
- Evaluation and Comparison
  - Robustness Analysis
  - Model Size Analysis
  - Seasonal-Trend Decomposition Analysis
- Future Research Opportunities
  - Inductive Biases for Time Series Transformers
  - Transformers and GNN for Time Series
  - Pre-trained Transformers for Time Series
  - Transformers with NAS/AutoML for Time Series



Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, Liang Sun, "Transformers in Time Series: A Survey", arXiv:2202.07125, 2022.

From Forecasting to *Decision*: Autoscaling

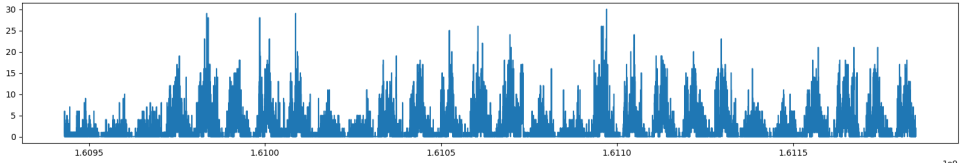
- Autoscaling in Cloud Computing
  - Automatically add/delete resources to match the demand
- Strategies
  - Fixed, Reactive (k8s HPA), Proactive/Predictive
- Proactive/Predictive Autoscaling
  - Many cloud applications with periodic pattern (often over 50%)
  - More potential in *periodic* scenario

periodic scenario
more potential for proactive autoscaling

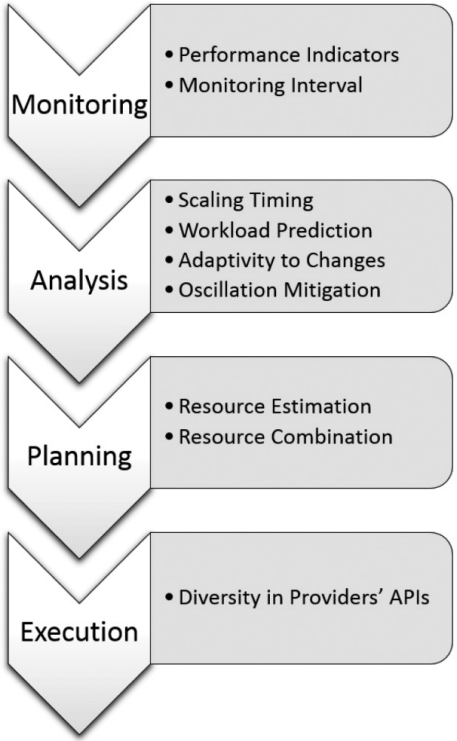
nonperiodic & random scenario
less potential for proactive autoscaling

Autoscaling

- Autoscaling procedure: continuously repeats MAPE
  - Monitoring
  - Analysis: time series forecasting/anomaly detection
  - Planning: decision with optimization
  - Execution
- Autoscaling challenges related to time series:
  - Complex periodic patterns, data contamination
  - Uncertainty: query arrival time, workload amount



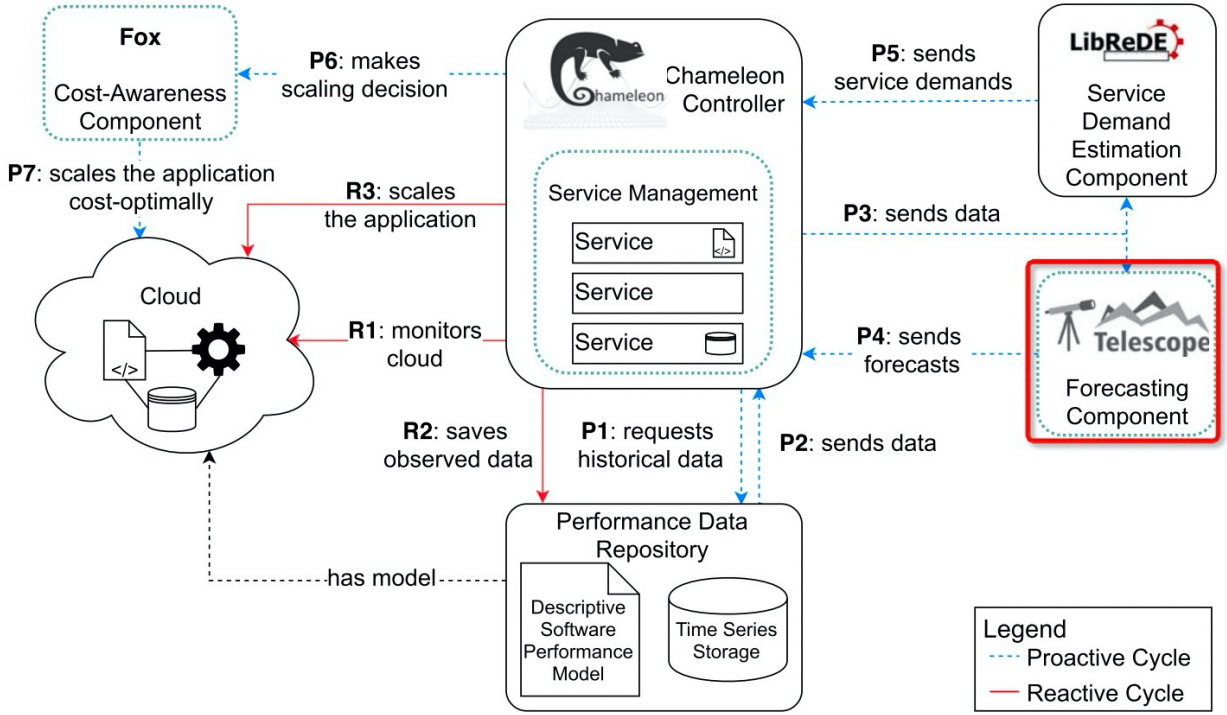
real-world QPS series from Alibaba container registry service



Qu, C., Calheiros, R. N., & Buyya, R. (2018). Auto-scaling web applications in clouds: A taxonomy and survey. ACM Computing Surveys (CSUR), 51(4), 1–33.

Chamulteon

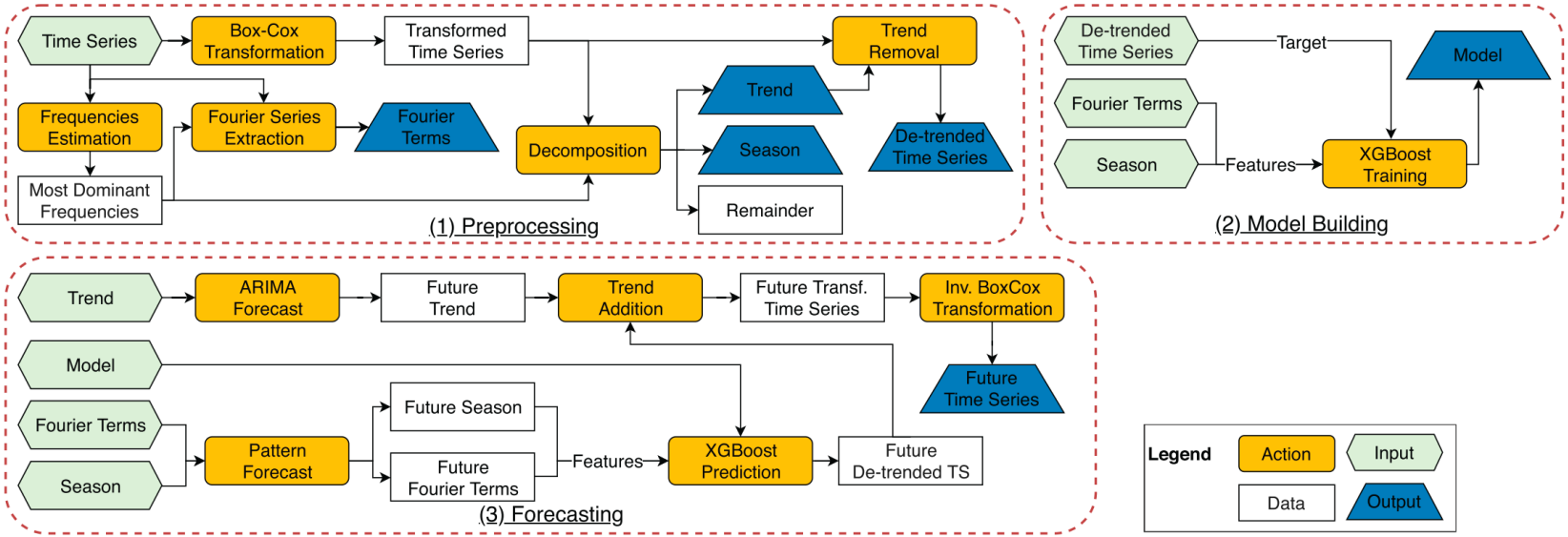
- System:
  - Reactive
  - Proactive
    - Monitor
    - Forecast
    - Demand Estimate
    - Decision
    - Optimization



Bauer, A., Lesch, V., Versluis, L., Ilyushkin, A., Herbst, N., & Kounev, S. (2019). Chamulteon: Coordinated auto-scaling of micro-services. IEEE International Conference on Distributed Computing Systems, 2019-July, 2015–2025.

Chamulteon: Forecasting

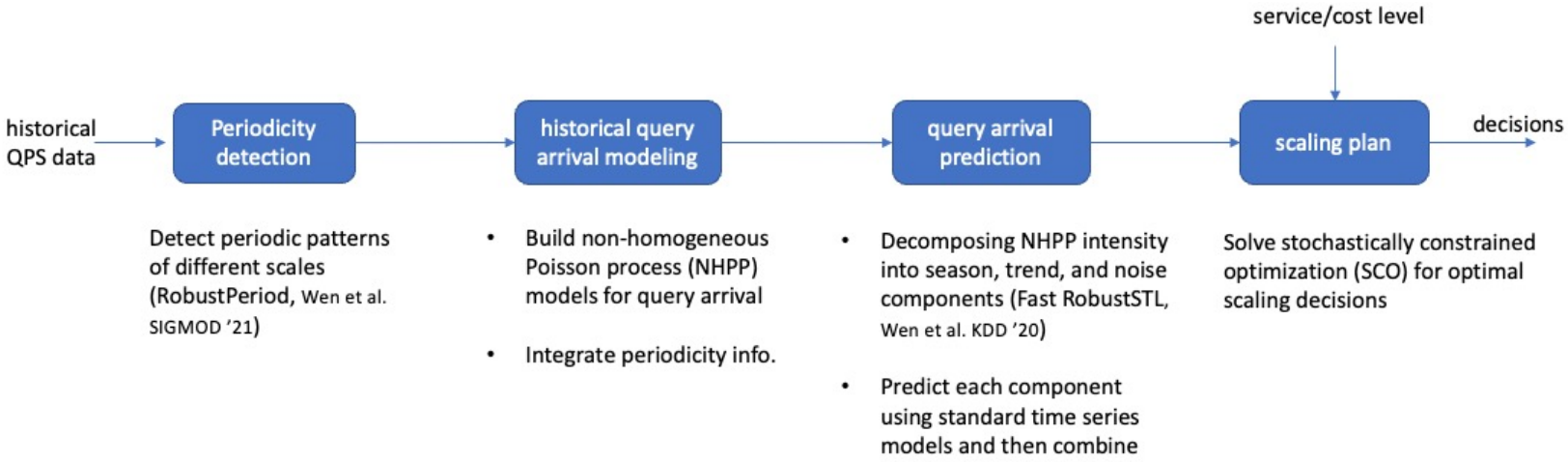
- Telescope: Optimized for *seasonal* time series forecasting



Bauer, André; Züfle, Marwin; Herbst, Nikolas; Kounev, Samuel; Curtef, V. (2020). Telescope : An Automatic Feature Extraction and Transformation Approach for Time Series Forecasting on a Level-Playing Field. IEEE 36th International Conference on Data Engineering (ICDE), 1902–1905.

Autoscaling: RobustScaler

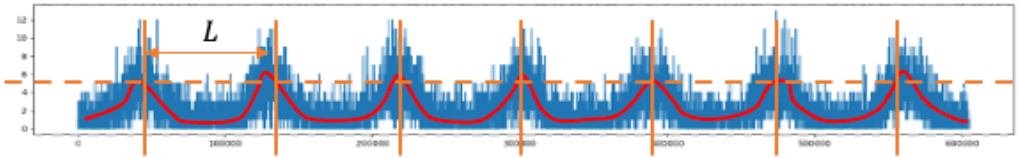
- A Robust and Efficient Proactive Autoscaling Scheme
  - Nonhomogeneous Poisson Process + periodicity regularization for complex periodic pattern and data contamination
  - Stochastically constrained optimization for uncertainty of query arrival times



Huajie Qian, Qingsong Wen, Liang Sun, Jing Gu, Qiulin Niu, Zhimin Tang, "RobustScaler: QoS-Aware Autoscaling for Complex Workloads," in Proc. IEEE 38th International Conference on Data Engineering (ICDE 2022), Kuala Lumpur, Malaysia, May 2022.

RobustScaler: Arrival Modeling as NHPP

Input:
$\boldsymbol{Q} = (Q_t, t = 1, \dots, T)$: query count within each time step Δt
Δt: length of each time step
L: known period length (from periodicity detection)



effect of periodicity penalty

Output:
$\boldsymbol{r} = (r_t, t = 1, \dots, T)$: $\log(intensity)$ for each time step, assumed piece-wise constant

Poisson likelihood for each interval t: $\frac{\exp(-\lambda_t \Delta t)(\lambda_t \Delta t)^{Q_t}}{Q_t!}$ with $\lambda_t = \exp(r_t)$

- Final formulation: $\min_{\boldsymbol{r}} \underbrace{-\boldsymbol{Q}^T \boldsymbol{r} + \Delta t \cdot \mathbf{1}^T \exp(\boldsymbol{r})}_{\text{log likelihood}} + \underbrace{\beta_1 \|D^2 \boldsymbol{r}\|_1}_{\text{smoothness penalty}} + \underbrace{\frac{\beta_2}{2} \|D_L \boldsymbol{r}\|_2}_{\text{periodicity penalty}}$

$$D^2 = \begin{bmatrix} 1 & -2 & 1 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & 1 & -2 & 1 & 0 & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & 1 & -2 & 1 \end{bmatrix} \in \mathrm{R}^{(T-2)\times T}$$

L entries apart

$$D_L = \begin{bmatrix} 1 & 0 & \cdots & 0 & -1 & \cdots & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 & -1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots & 0 \\ 0 & \cdots & \cdots & 1 & 0 & \cdots & 0 & -1 \end{bmatrix} \in \mathrm{R}^{(T-L)\times T}$$

RobustScaler: Constrained Optimization Model

- Optimize cost (i.e., idle time) while QoS (i.e., response time (RT) or hitting probability (HP)) is controlled
  - RobustScaler-HP: Minimize average idle time while controlling HP above a threshold $1-\alpha$:

$$\min_{x_1,\ldots,x_K\geq 0} \sum_{i=1}^{K} E[\text{idle time of instance}_i] \quad \text{s.t. probability of } i\text{-th query being hit} \geq 1-\alpha,\ i=1,\ldots,K$$

$$\longrightarrow \quad x_i^* = \alpha \text{ quantile of } (\xi_i - \tau_i)$$

  - RobustScaler-RT: Minimize average **idle time** while controlling average **RT** below a threshold d:

$$\min_{x_1,\ldots,x_K\geq 0} \sum_{i=1}^{K} E[(\xi_i - \tau_i - x_i)_+] \quad \text{s.t. } \mu_s + E[(\tau_i - (\xi_i - x_i)_+)_+] \leq d,\ i=1,\ldots,K$$

$$\longrightarrow \quad x_i^* \text{ such that } E[(\tau_i - (\xi_i - x_i^*)_+)_+] = d - \mu_s$$

  - RobustScaler-cost: Minimize average **RT** while controlling average **idle time** below a threshold B:

$$\min_{x_1,\ldots,x_K\geq 0} \sum_{i=1}^{K} E[(\tau_i - (\xi_i - x_i)_+)_+] \quad \text{s.t. } E[(\xi_i - \tau_i - x_i)_+] \leq B,\ i=1,\ldots,K$$

$$\longrightarrow \quad \begin{cases} x_i^* = 0, & \text{if } E[(\xi_i - \tau_i)_+] \leq B \\ x_i^* \text{ s.t. } E[(\xi_i - \tau_i - x_i^*)_+] = B & \text{otherwise} \end{cases}$$

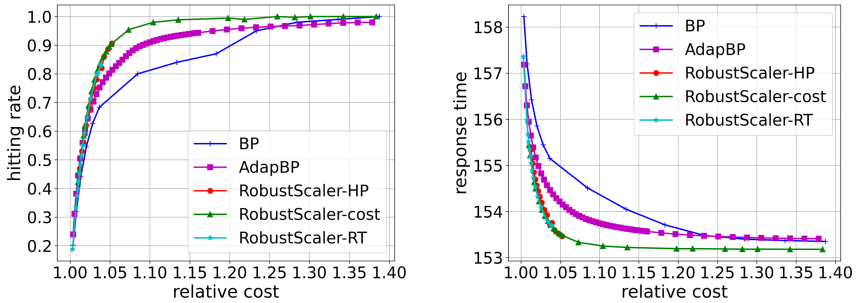
RobustScaler: QoS-cost Pareto Plots

- Baseline autoscaling methods:

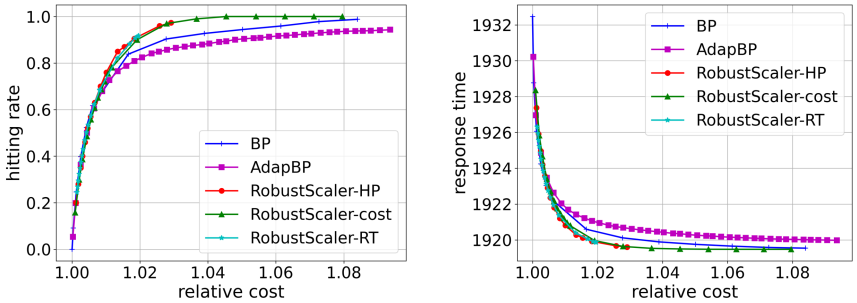
- **Backup Pool (BP):** A warm instance pool of fixed size B is maintained throughout, whenever one of the instances is consumed, fill with a new one immediately. (B=0 is purely reactive)
- **Adaptive Backup Pool (AdapBP):** Adaptive version of BP with B=*coeff**QPS for some fixed coefficient *coeff*.

	#queries	duration
Alibaba cluster 2018 trace (https://github.com/alibaba/clusterdata)	503,850	5 days
Google cluster 2019 trace (https://github.com/google/cluster-data)	20,254	24 hours

QoS-cost Pareto Plots on Alibaba trace



QoS-cost Pareto Plots on Google trace

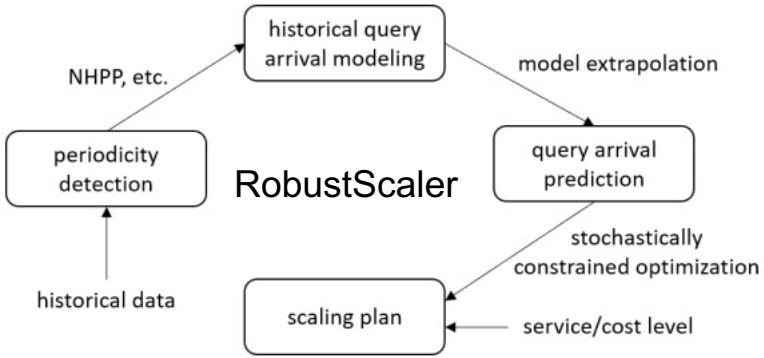


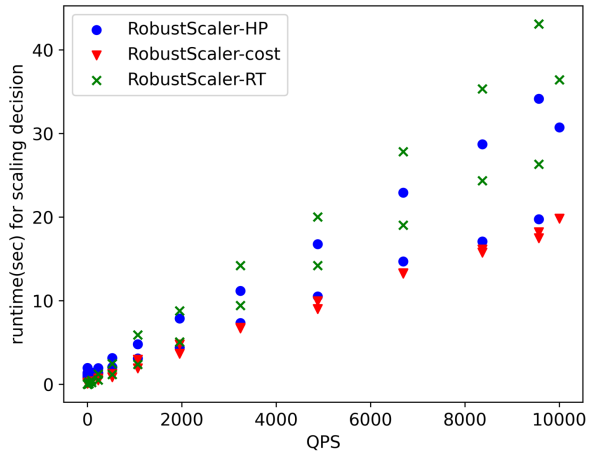
Under the same relative cost, higher hitting rate or lower response time is better.

RobustScaler is better than others (AdapBP, BP)

RobustScaler: Scalability

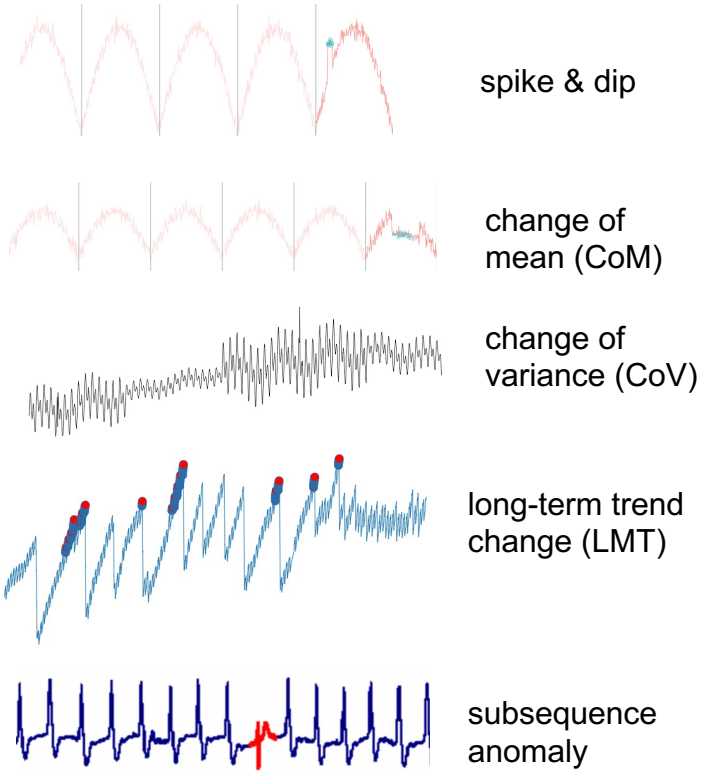
- Scalability by Modules
  - Periodicity detection & query arrival prediction: RobustPeriod & Fast RobustSTL known to be scalable
  - Historical query arrival modeling: 7 secs for QPS series of four days, 100 secs for QPS series of three weeks
  - Scaling plan: Linearly growing runtime, remain efficient under high QPS





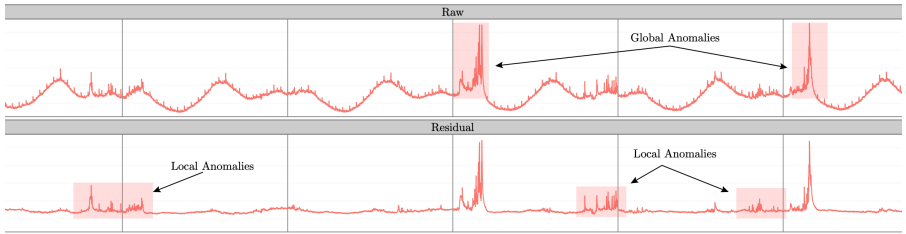
Time Series Anomaly Detection: Background

- Time-series Anomalies
  - *Points/sequences that* significantly deviate *from the* normal pattern
  - **Differences** from static data anomaly detection:
    - Anomalies have temporal context
    - Noise is non-stationary (i.e., time-varying noise)
    - Lack of anomaly labels
- Types of Anomalies
  - *Point:* Spikes/Dips, CoM, CoV, LMT
  - *Subsequence anomaly: i*dentifying anomalous subsequences (sequences of points)
- Models
  - *Traditional methods, deep models*



Anomaly Detection: Decomposition based Model

- Decomposition + Statistics
  - Seasonal ESD (S-ESD)
    - Seasonal-trend decomposition (STL) for residual
    - Extreme Studentized Deviate (ESD) test for anomalies
  - Seasonal Hybrid ESD (S-H-ESD)
    - Robust statistics in ESD: median, median absolute deviation (MAD)
    - More robust to a higher percentage of anomalies
  - Pros
    - Both global and local anomalies from decomposition
    - detect local anomalies that would otherwise be masked by seasonal data
  - Cons
    - STL→ not robustness to trend changes, seasonalies changes



Algorithm 1 S-ESD Algorithm

Input:
X = A time series
n = number of observations in X
k = max anomalies (iterations in ESD)

Output:
X_A = An anomaly vector wherein each element is a tuple $(timestamp, observed\ value)$

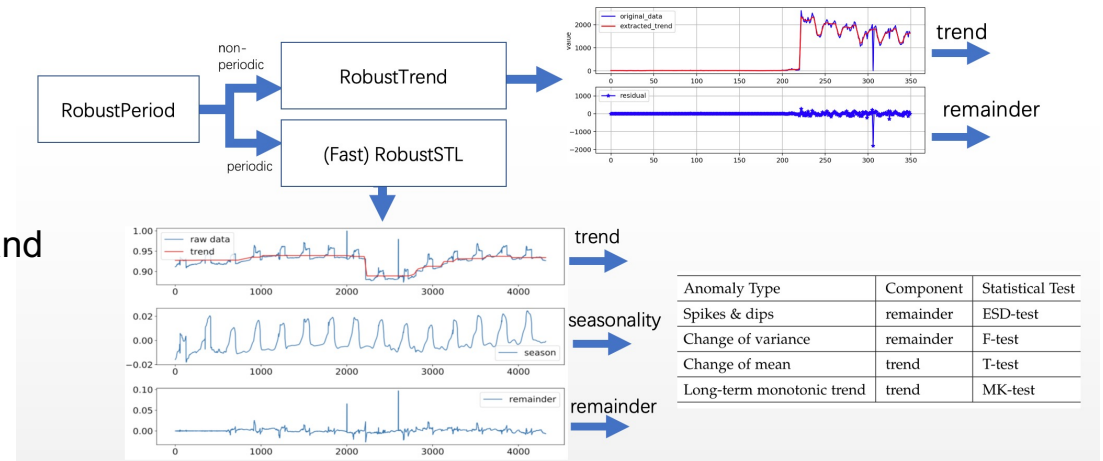
Require:
$k \leq (n \times .49)$
1. Extract seasonal component S_X using STL Variant
2. Compute median $\tilde{X}$
/* Compute residual */
3. $R_X = X - S_X - \tilde{X}$
/* Detect anomalies vector X_A using ESD */
4. $X_A = \text{ESD}(R, k)$

return X_A

Hochenbaum, Jordan, Owen S. Vallis, and Arun Kejariwal. "Automatic anomaly detection in the cloud via statistical learning." arXiv preprint arXiv:1704.07706 (2017).

Anomaly Detection: Decomposition based Model

- Robust Decomposition + Statistics
  - Robust time series decompositions reduce complexity and bring explainability
  - Robust statistical tests on each components lead to high accuracy and efficiency

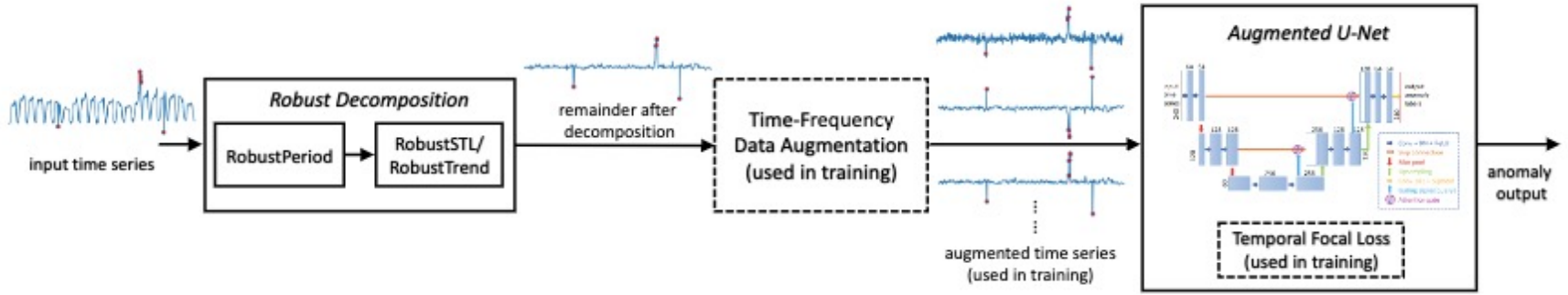


Anomaly Type	Component	Statistical Test
Spikes & dips	remainder	ESD-test
Change of variance	remainder	F-test
Change of mean	trend	T-test
Long-term monotonic trend	trend	MK-test

Yingying Zhang, Zhengxiong Guan, Huajie Qian, Leili Xu, Hengbo Liu, Qingsong Wen, Liang Sun, Junwei Jiang, Lunting Fan, Min Ke, "CloudRCA: A Root Cause Analysis Framework for Cloud Computing Platforms," in Proc. 30th ACM International Conference on Information and Knowledge Management (CIKM 2021), Queensland, Australia, Nov. 2021.

Anomaly Detection: RobustTAD

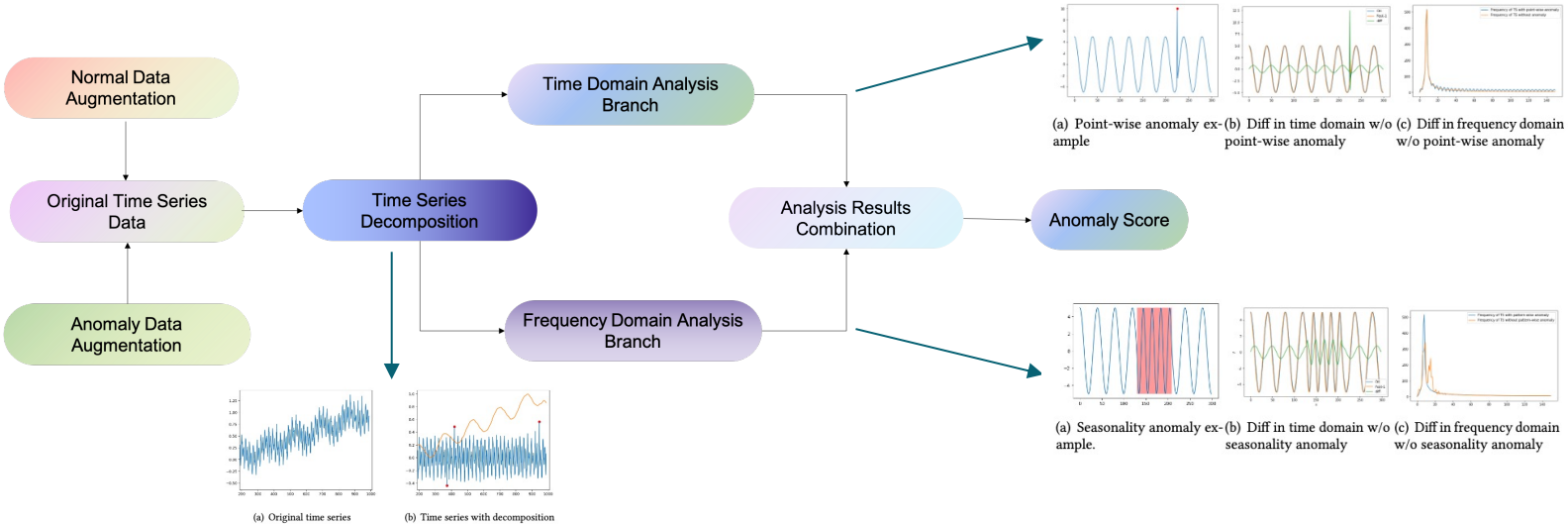
- Decomposition + Data Augmentation + *U-Net*
  - Robust decomposition: handle complicated patterns, and simplify neural network
  - Data Augmentation: mitigate the effects of limited labeled data
  - U-Net: capture multi-scale information
  - Loss: label-based weight and value-based weight for unbalanced label



Jingkun Gao, Xiaomin Song, Qingsong Wen, Pichao Wang, Liang Sun, Huan Xu, "RobustTAD: Robust Time Series Anomaly Detection via Decomposition and Convolutional Neural Networks," in ACM SIGKDD Workshop on Mining and Learning from Time Series (KDD-MiLeTS 2020), San Diego, CA, Aug. 2020.

Anomaly Detection: TFAD

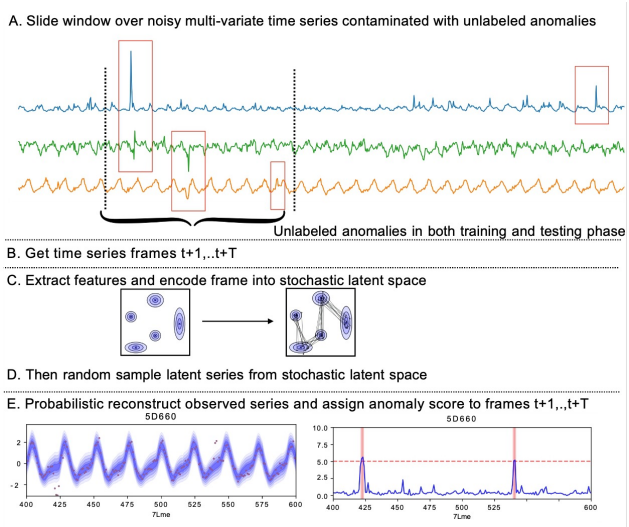
- Data Augmentation + Decomposition + *Time-Frequency Processing*
  - Anomalies types: seasonal anomaly, trend anomaly, global/context point anomaly
  - Seasonal anomaly easier to detect in Frequency; point anomaly easier to detect in Time

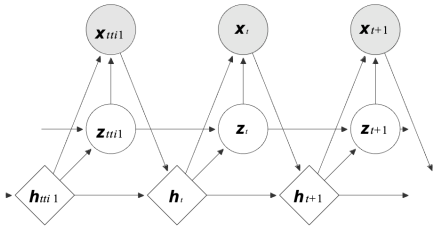


Chaoli Zhang, Tian Zhou, Qingsong Wen, Liang Sun, "TFAD: A Decomposition Time Series Anomaly Detection Architecture with Time-Freq Analysis," in Proc. 31st ACM International Conference on Information and Knowledge Management (CIKM 2022), Atlanta, GA, Oct. 2022.

Anomaly Detection: RDSSM

- Robust Deep State Space Model
  - Fully **unsupervised**: training on contaminated data w/o labels
  - Model temporal dependencies with deep state space model
  - With t-distribution as decoder for convergence on contaminated dataset



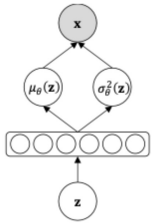


Generative model

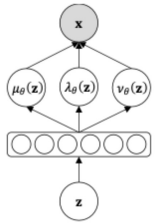
Observed sequence : with time-varying noise

Stochastic cells for modeling non-linear stochastic transition

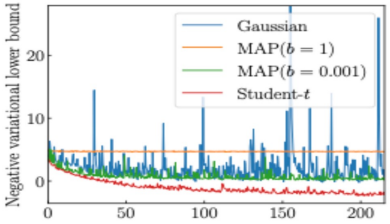
Deterministic cells for modeling long-term and non-linear transition



(a) Gaussian decoder.



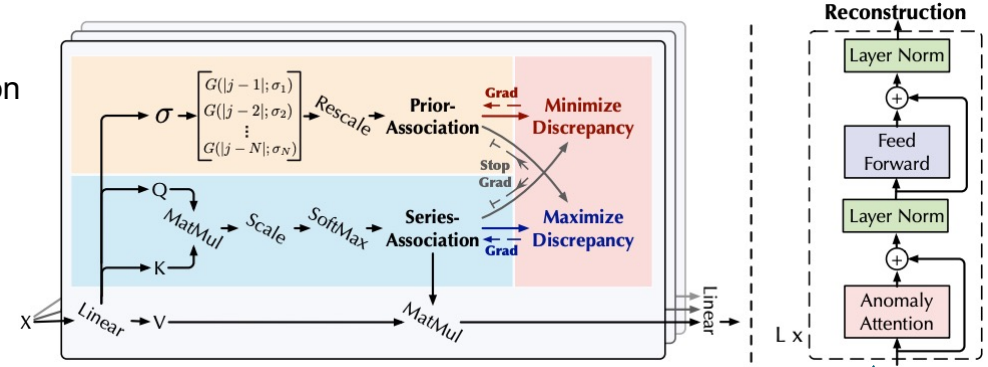
(b) Student-t decoder.



Longyuan Li, Junchi Yan, Qingsong Wen, Yaohui Jin, and Xiaokang Yang, "Learning Robust Deep State Space for Unsupervised Anomaly Detection in Contaminated Time-Series," IEEE Transactions on Knowledge and Data Engineering (TKDE 2022), 2022.

Anomaly Detection: Anomaly Transformer

- Architecture: Anomaly Transformer with Anomaly-Attention
- Training Strategy: Minimax Association Learning
- Criterion: Association-based Anomaly Criterion



$$\text{Prior-Association: } \mathcal{P}^l = \text{Rescale}\left(\left[\frac{1}{\sqrt{2\pi}\sigma_i}\exp\left(-\frac{|j-i|^2}{2\sigma_i^2}\right)\right]_{i,j\in\{1,\cdots,N\}}\right)$$

$$\text{Series-Association: } \mathcal{S}^l = \text{Softmax}\left(\frac{\mathcal{Q}\mathcal{K}^{\mathrm{T}}}{\sqrt{d_{\text{model}}}}\right)$$

Learnable Gaussian kernel: adapt to TS patterns
Self-attention: find the most effective associations

$$\text{AssDis}(\mathcal{P},\mathcal{S};\mathcal{X}) = \left[\frac{1}{L}\sum_{l=1}^{L}\left(\text{KL}(\mathcal{P}_{i,:}^l\|\mathcal{S}_{i,:}^l) + \text{KL}(\mathcal{S}_{i,:}^l\|\mathcal{P}_{i,:}^l)\right)\right]_{i=1,\cdots,N}$$

$$\mathcal{L}_{\text{Total}}(\widehat{\mathcal{X}},\mathcal{P},\mathcal{S},\lambda;\mathcal{X}) = \|\mathcal{X}-\widehat{\mathcal{X}}\|_{\text{F}}^2 - \lambda\times\|\text{AssDis}(\mathcal{P},\mathcal{S};\mathcal{X})\|_1$$

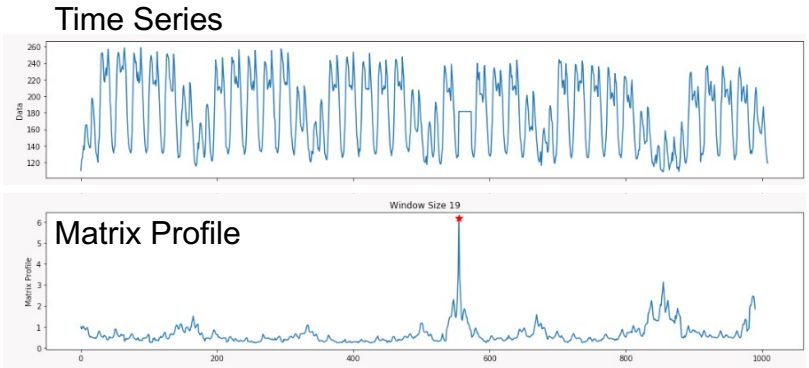
$$\mathcal{Z}^l = \text{Layer-Norm}\left(\text{Anomaly-Attention}(\mathcal{X}^{l-1}) + \mathcal{X}^{l-1}\right)$$
$$\mathcal{X}^l = \text{Layer-Norm}\left(\text{Feed-Forward}(\mathcal{Z}^l) + \mathcal{Z}^l\right),$$

Learning underlying associations from deep **multi-level** features.

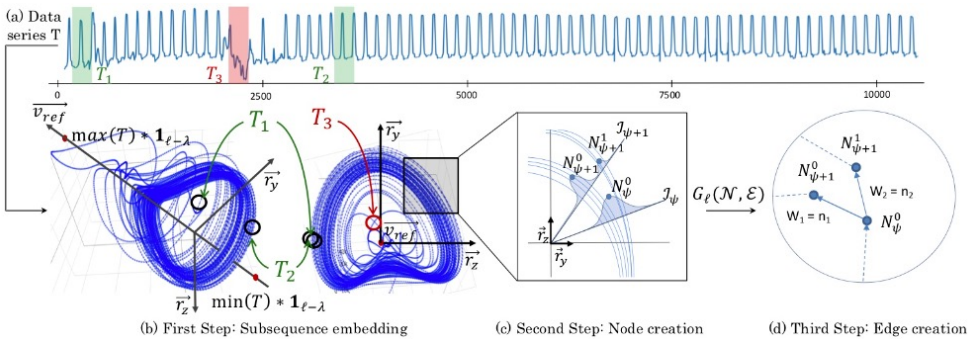
Jiehui Xu, Haixu Wu, Jianmin Wang, Mingsheng Long, "Anomaly Transformer: Time Series Anomaly Detection with Association Discrepancy," ICLR 2022.

Subsequence Anomaly Detection

- Matrix Profile (MP)
  - MP: records the distance of the subsequence to its its nearest neighbor
  - The higher the MP value, the greater the dissimilarity → such areas are anomalies/discords



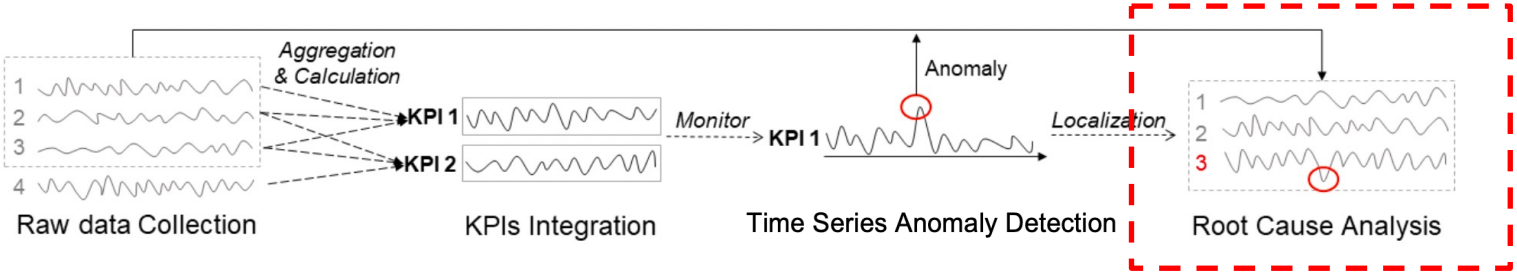
- Series2Graph
  - Graph representation of subsequences
  - Anomaly score based on the graph (edges/nodes and their weights/degrees)



Yeh, C. C. M., Zhu, Y., Ulanova, L., ... & Keogh, E."Matrix profile I: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets," ICDM, 2016.
Lu, Yue. "What are Time Series Discords?", Matrix Profile Foundation Tutorial, Aug. 2020.
Boniol, Paul, and Themis Palpanas. "Series2Graph: Graph-based Subsequence Anomaly Detection for Time Series," VLDB, 2020.

From Time Series *Anomaly Detection to Localization*

- Fault Cause Localization
  - Identify the cause for the fault by root cause analysis (RCA) that can best explain the observed system/network disorders



Challenge for RCA is how to accurately and quickly find a set of dimension-value combinations (leaf nodes) to resolve the anomaly detected in the root node.

[1] CMMD: Cross-Metric Multi-Dimensional Root Cause Analysis, https://arxiv.org/abs/2203.16280
[2] Integrated Fault and Security Management, Ehab Al-Shaer, Yan Chen, in Information Assurance, 2008

Multi-dimensional Fault Cause Localization

- Key performance indicators (KPIs) with multi-dimensional attributes are usually collected to provide good quality of service

category	city	Online	UV
shirt	Beijing	Y	3042
jacket	Shanghai	N	4133
skirt	Hongzhou	Y	2789

- It is desired to quickly localize the root cause when an anomaly happens so that operators can take actions to mitigate or fix the problem

Multi-dimensional Fault Cause Localization

- Fault Cause Localization

Category	City	Online	Anomaly
shirt	Beijing	Y	1
shirt	Beijing	N	1
shirt	Shanghai	N	0
skirt	Beijing	Y	0

Root cause is (shirt & Beijing)

- Fault Cause Localization aims to find the root cause (a combination of attribute values) that contribute most to the total value of anomalous

Fault Cause Localization

- Existing methods
  - Bottom up

    1, genetic algorithm: CMMD [1]

    2, Monte Carlo tree search: HotSpot [2]，GMCTS [3]
  - Top down

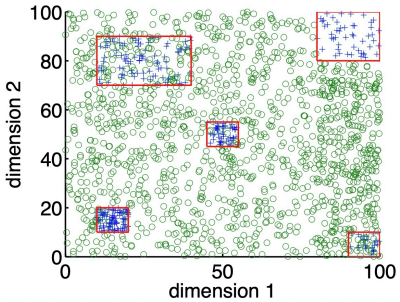
    1, Score based clustering:

    Squeeze [4], AutoRoot [5]

[1] Yan, Shifu, et al. "CMMD: Cross-Metric Multi-Dimensional Root Cause Analysis." arXiv preprint arXiv:2203.16280 (2022).
[2] Sun, Yongqian, et al. "Hotspot: Anomaly localization for additive kpis with multi-dimensional attributes." IEEE Access 6 (2018): 10909-10923.
[3] Wang, Chunlin, et al. "Network Abnormality Location Algorithm Based on Greedy Monte Carlo Tree." 2022 14th International Conference on Machine Learning and Computing (ICMLC). 2022.
[4] Li, Zeyan, et al. "Generic and robust localization of multi-dimensional root causes." 2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE). IEEE, 2019.
[5] Wang, Hanzhang, et al. "Groot: An event-graph-based approach for root cause analysis in industrial settings." 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, 2021.

Fault Cause Localization via Rule Set Learning

- Rule set learning



X	Y	Z	S	T	label
x1	y3	z1	s2	t1	1
x1	y2	z3	s1	t4	0
…	…	…	…	…	…

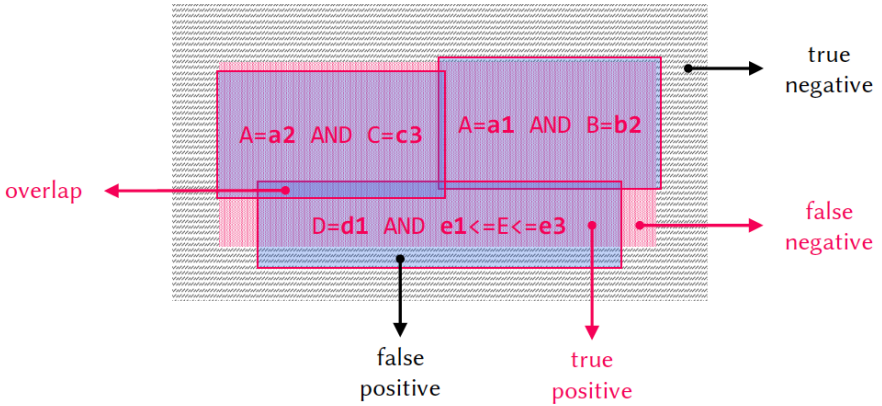
```
IF    ( X=x1 AND Y=y2 )
  OR  ( X=x2 AND Z=z3 )
  OR  ( S=s1 AND t1<=T<=t3 )
  OR  ( … )
THEN label=1
```

The rule set learning methods aim to find discriminate patterns that covers the samples of interest class

Fault Cause Localization

- Rule Set Learning
  - High classification accuracy
  - It can explicitly controls rule complexity and interpretability
  - It can handle large-scale data

$$L(\mathcal{S}) = \beta_0 \underbrace{\sum_{\mathcal{R}\in\mathcal{S}} \left|\mathcal{X}^-_{\{\mathcal{R}\}}\right|}_{\text{FP}} + \beta_1 \underbrace{\left|\mathcal{X}^+ \setminus \mathcal{X}^+_{\mathcal{S}}\right|}_{\text{FN}} + \beta_2 \underbrace{\left(\sum_{\mathcal{R}\in\mathcal{S}} \left|\mathcal{X}^+_{\{\mathcal{R}\}}\right| - \left|\mathcal{X}^+_{\mathcal{S}}\right|\right)}_{\text{Overlap}} + \lambda \underbrace{\sum_{\mathcal{R}\in\mathcal{S}} |\mathcal{R}|}_{\text{Complexity}}$$



Fan Yang, Kai He, Linxiao Yang, Hongxia Du, Jingbang Yang, Bo Yang, Liang Sun, Learning Interpretable Decision Rule Sets: A Submodular Optimization Approach, In Advances in Neural Information Processing Systems 2021 (NeurIPS 2021), Spotlight Paper

Rule Set Learning: A Submodular Optimization Approach

Minimize

$$L(\mathcal{S}) = \beta_0 \underbrace{\sum_{\mathcal{R}\in\mathcal{S}} \left|\mathcal{X}^-_{\{\mathcal{R}\}}\right|}_{\text{FP}} + \beta_1 \underbrace{\left|\mathcal{X}^+ \setminus \mathcal{X}^+_{\mathcal{S}}\right|}_{\text{FN}} + \beta_2 \underbrace{\left(\sum_{\mathcal{R}\in\mathcal{S}} \left|\mathcal{X}^+_{\{\mathcal{R}\}}\right| - \left|\mathcal{X}^+_{\mathcal{S}}\right|\right)}_{\text{Overlap}} + \lambda \underbrace{\sum_{\mathcal{R}\in\mathcal{S}} |\mathcal{R}|}_{\text{Complexity}}$$

Reorganize

$$L(\mathcal{S}) = \underbrace{\beta_1 \left|\mathcal{X}^+\right|}_{\text{Const}} - \underbrace{(\beta_1+\beta_2)\left|\mathcal{X}^+_{\mathcal{S}}\right|}_{\text{Submodular}} + \underbrace{\sum_{\mathcal{R}\in\mathcal{S}} \beta_0 \left|\mathcal{X}^-_{\{\mathcal{R}\}}\right| + \beta_2 \left|\mathcal{X}^+_{\{\mathcal{R}\}}\right| + \lambda|\mathcal{R}|}_{\text{Modular}}$$

Maximize

$$V(\mathcal{S}) = g(\mathcal{S}) - \sum_{\mathcal{R}\in\mathcal{S}} c(\mathcal{R})$$

A cardinality constrained submodular maximization problem

Rule Set Learning: A Submodular Optimization Approach

- Distorted greedy

Algorithm 1 Rule set learning

Input: Training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, hyperparameters $(\boldsymbol{\beta}, \lambda)$, cardinality K
Initialize $\mathcal{S} \leftarrow \emptyset$
for $k = 1$ **to** K **do**
  Define $v_k(\mathcal{R}) = (1 - 1/K)^{K-k} g(\mathcal{R}|\mathcal{S}) - c(\mathcal{R})$ /* $g(\mathcal{R}|\mathcal{S}) \coloneqq g(\mathcal{S} \cup \{\mathcal{R}\}) - g(\mathcal{S})$ */
  Solve $\mathcal{R}^\star \leftarrow \arg\max_{\mathcal{R} \subseteq [d]} v_k(\mathcal{R})$
  if $v_k(\mathcal{R}^\star) > 0$ **then** $\mathcal{S} \leftarrow \mathcal{S} \cup \{\mathcal{R}^\star\}$ **end if**
end for
Output: $\mathcal{S}$

- Approximation guarantee

$$V(\mathcal{S}) = g(\mathcal{S}) - \sum_{\mathcal{R} \in \mathcal{S}} c(\mathcal{R}) \geq (1 - 1/e) g(\boldsymbol{OPT}) - \sum_{\mathcal{R} \in \boldsymbol{OPT}} c(\mathcal{R})$$

Fault Cause Localization

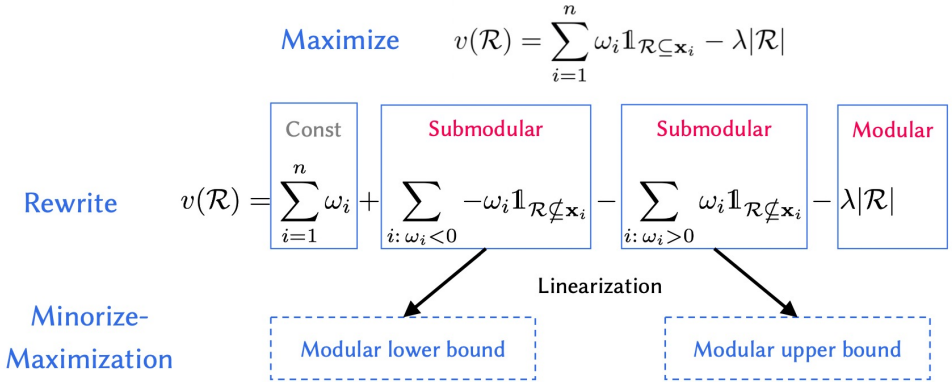
- Marginal Gain Maximization

4 Define $v_k(\mathcal{R}) = (1 - 1/K)^{K-k} g(\mathcal{R}|\mathcal{S}) - c(\mathcal{R})$
5 Solve $\mathcal{R}^\star \leftarrow \arg\max_{\mathcal{R} \subseteq [d]} v_k(\mathcal{R})$

Exhaustive enumeration

$O(2^d)$ ☹

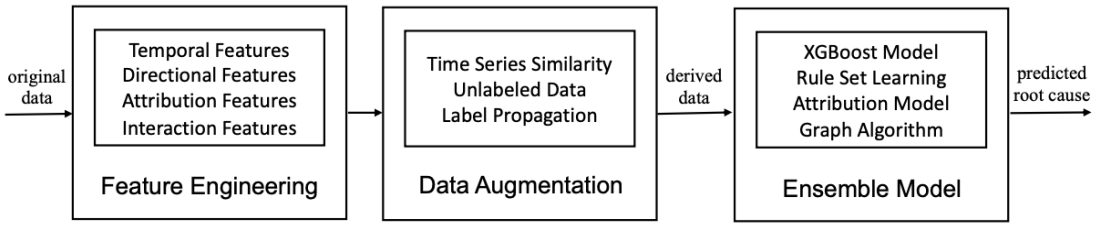
- We rewrite the subobjective as a difference of submodular (DS) functions

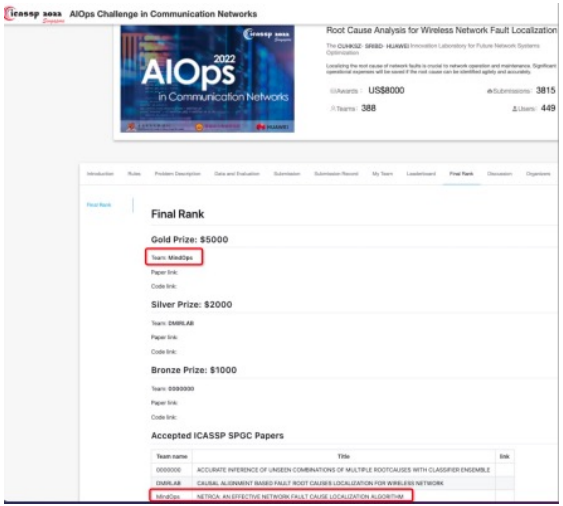


- If $v(\mathcal{R})$ is nonnegative monotonic submodular function, it can be optimized using greedy algorithm
- We express it as the difference of submodular functions and apply MM algorithm

NetRCA

- Effective and efficient root cause localization for network faults
  - Key idea: 1. The root cause localization problem can be treated as a rule learning problem
    2. Multiple models ensemble can further improve the precision
  - Three blocks: Feature engineering, Data augment, Model ensemble





Chaoli Zhang*, Zhiqiang Zhou*, Yingying Zhang*, Linxiao Yang*, Kai He*, Qingsong Wen*, Liang Sun* (*Equally Contributed), "NetRCA: An Effective Network Fault Cause Localization Algorithm," in Proc. IEEE 47th International Conference on Acoustics, Speech, and Signal Processing (ICASSP), May 2022. **(ICASSP'22 AIOps Challenge, First Place (1/382))**

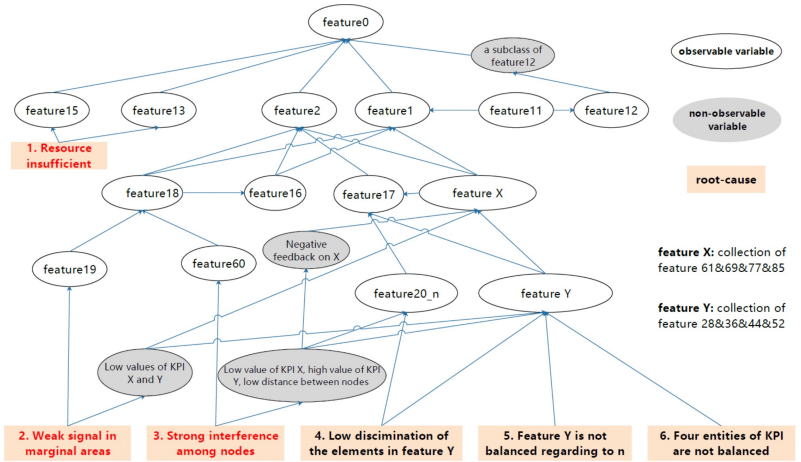
NetRCA

- ICASSP 2022 AIOps Challenge

Problem: Given the KPIs to find the root cause?

Challenges:
1. Interaction of features
2. Imbalanced data

Advantages of rule learning:
1. Able to capture the interaction of features
2. Able to handle data imbalance
3. Interpretable



NetRCA

- ICASSP 2022 AIOps Challenge

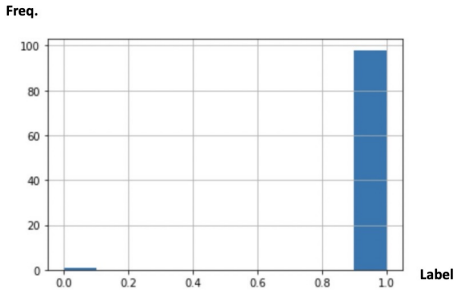
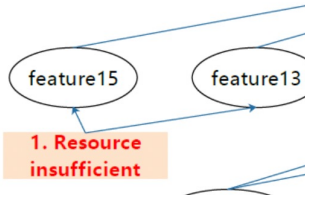


Figure 2: Histogram of samples covered by rule "$feature13_{min} \leq 1.75e^5$ and $feature13_{max} \leq 4.00e^5$ and $feature13_{quantile0.4} \leq 1.92e^5$" to predict root 1.



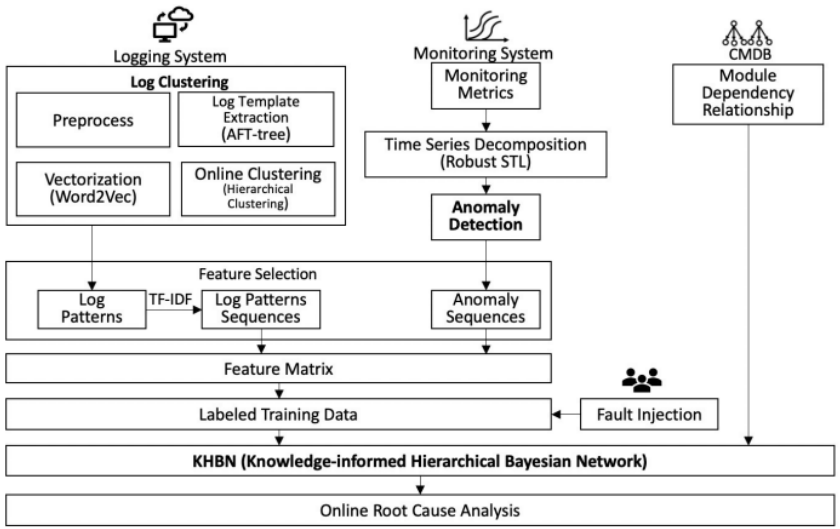
causal graph matches the intuition behind the rules.

a lower value of feature 13 is more likely linked to root cause 1.

the accuracy of the rule is close to one

CloudRCA

- Framework of CloudRCA
  - Advantages:
    - A general framework
    - Integrate multiple sources of data
    - Improve traditional algorithms
    - Infer root cause types directly

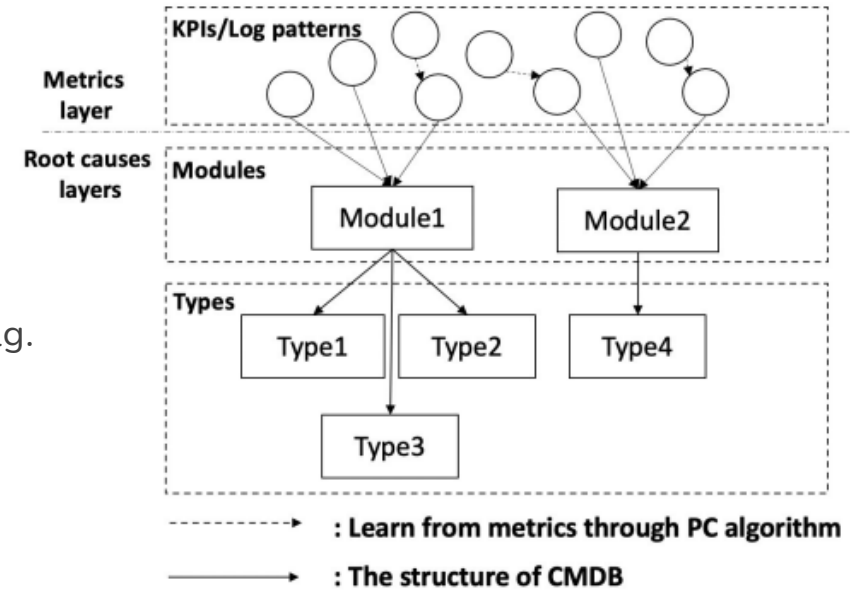


Yingying Zhang, Zhengxiong Guan, Huajie Qian, Leili Xu, Hengbo Liu, Qingsong Wen, Liang Sun, Junwei Jiang, Lunting Fan, Min Ke, "CloudRCA: A Root Cause Analysis Framework for Cloud Computing Platforms," in Proc. 30th ACM International Conference on Information and Knowledge Management (CIKM 2021), Queensland, Australia, Nov. 2021.

CloudRCA

- Knowledge-informed Hierarchical Bayesian Network
  - Integrate multi-sources data:
    - Key performance indicators (KPIs)
    - Log pattern
    - Tree-based topology data from CMDB
  - Structure of KHBN
    - Metrics layer + root cause layer
    - Causal discovery: CMDB knowledge + PC Alg.
  - Root cause identification
    - Conditional probability

$$\text{type}_i^* = \text{argmax}_{t_i} P(t_i | m_k, s_1, s_2, \dots, s_j) P(m_k | s_1, s_2, \dots, s_j)$$



Tutorial Summary

- **Introduction:** Real-world Challenges and Needs for Robust Time Series
- **Preliminaries:** Multiple Disciplines: Statistics, Signal Processing, Optimization, Deep Learning
- **Robust Time Series Processing Blocks**
  - Time Series Periodicity Detection
  - Time Series Decomposition: Trend Filtering, Seasonal-Trend Decomposition
  - Time Series Similarity
- **Robust Time Series Applications and Practices**
  - Time Series Forecasting
  - From Forecasting to Decision: Autoscaling
  - Time Series Anomaly Detection
  - From Anomaly Detection to Localization: Fault Cause Localization

Alibaba DAMO Academy

Thanks!
Q&A

Tutorial Website: https://github.com/DAMO-DI-ML/KDD2022-Tutorial-Time-Series
Alibaba DAMO Academy - Decision Intelligence Lab: https://damo.alibaba.com/labs/decision-intelligence

Hiring: Time Series, XAI

Research Interns, Postdocs, and
Research & Applied Scientists

Seattle (US), Hangzhou (China)

Email CV to

qingsong.wen@alibaba-inc.com
liang.sun@alibaba-inc.com