Forecasting and Root-Cause Analysis of Drifting Accelerator Systems with Continual Learning

SY-ABT-BTP

Algelly Malik

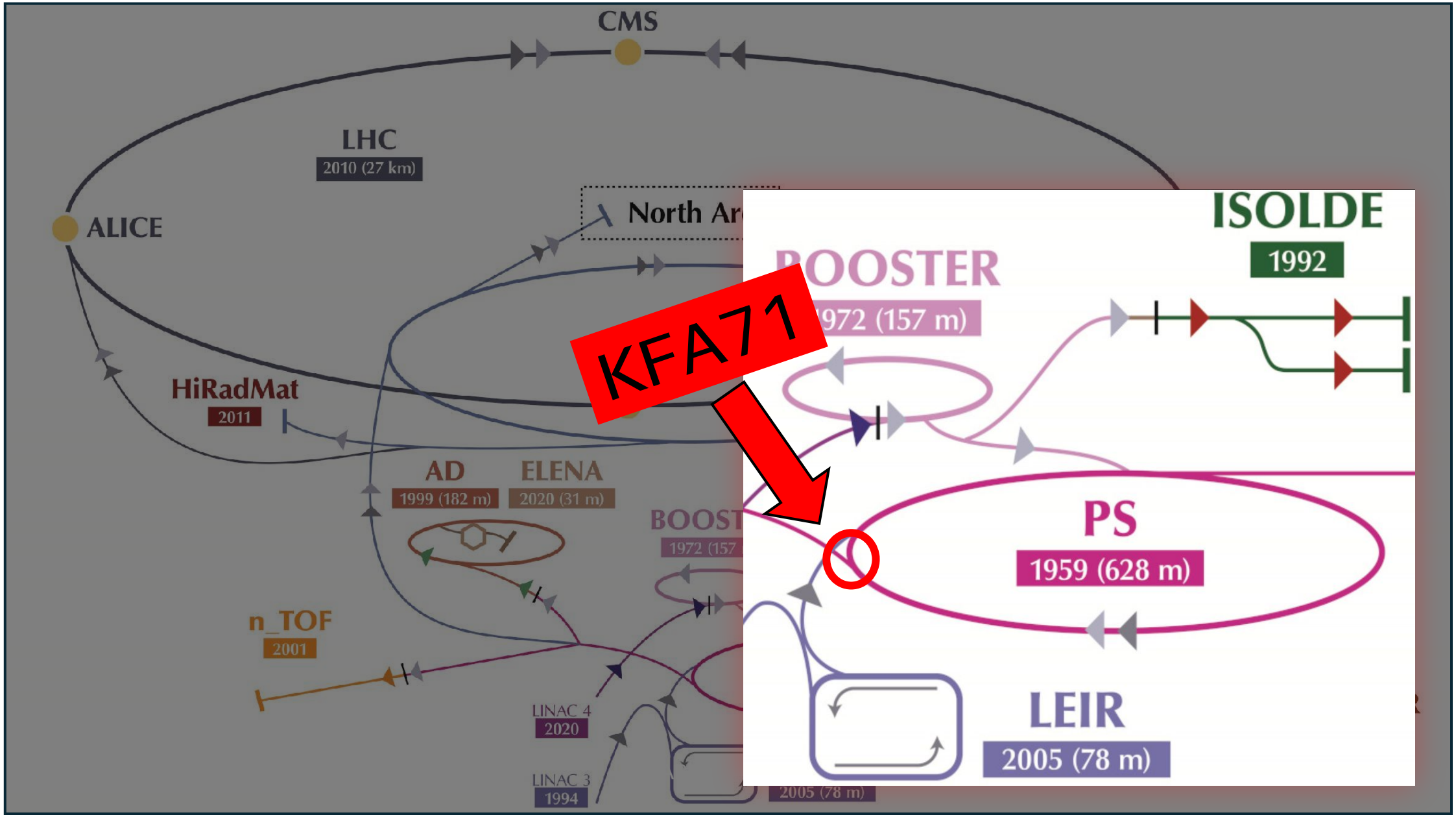
CERN Supervisors: *Francesco Velotti* & *Kostas Papastergiou*

UNIGE Supervisors*: Alexandros Kalousis & Stéphane Marchand-Maillet*

Content

1. System Overview
2. Data & Labeling
3. Anomaly Detection & Forecasting
4. Continual Learning
5. Root Cause Analysis
6. Conclusion & Outlook

System Overview – The KFA71/79 Extraction Kicker



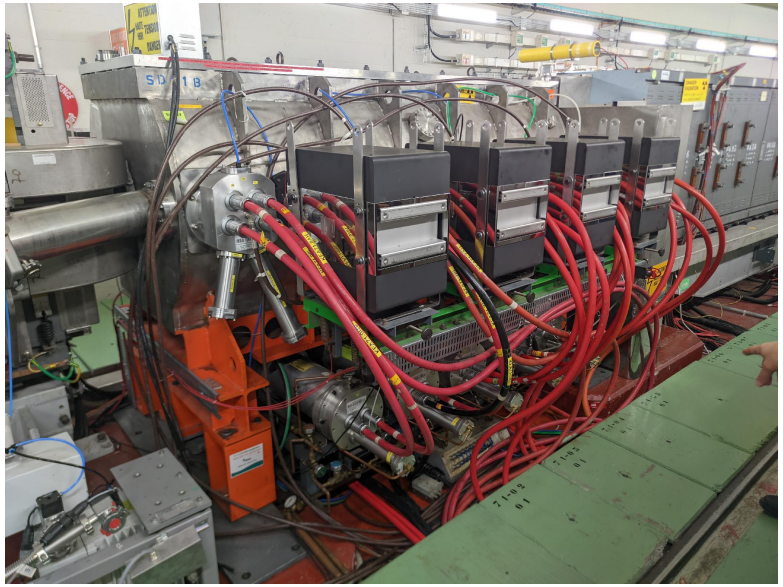
System Overview – The KFA71/79 Extraction Kicker

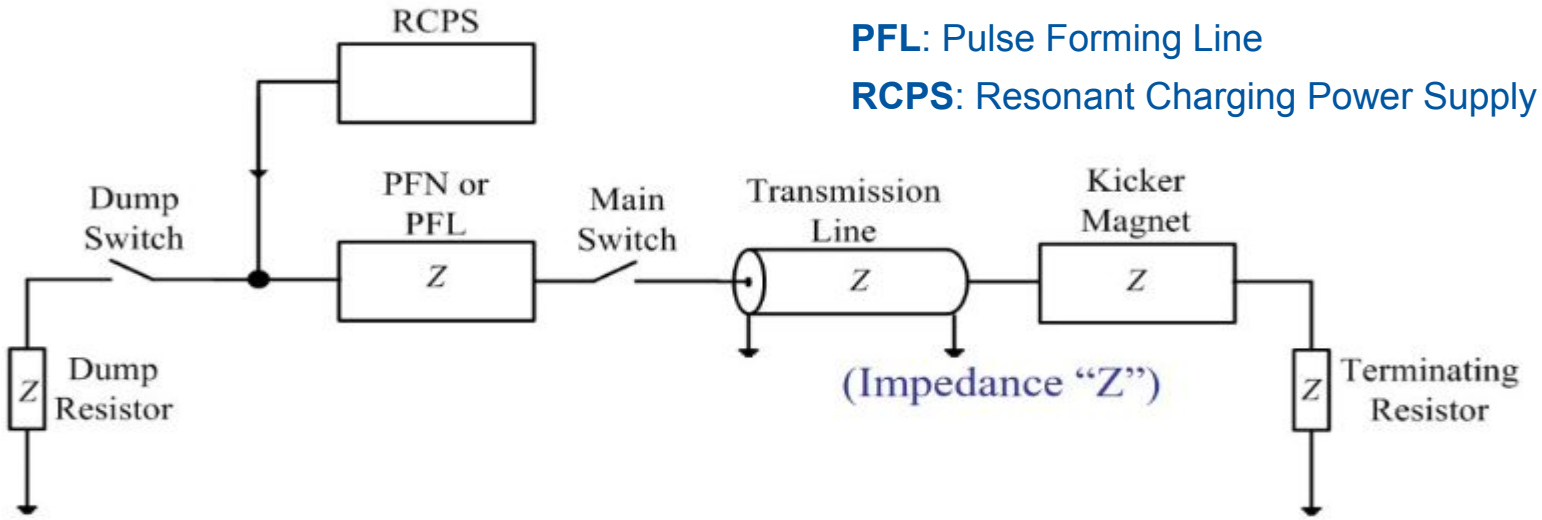
Purpose: Fast-pulsed magnet system to extract particle beams from the Proton Synchrotron (PS).

Components: 12 generator modules operating simultaneously in vacuum tanks.

Output:

High-voltage pulses (~80 kV peak).

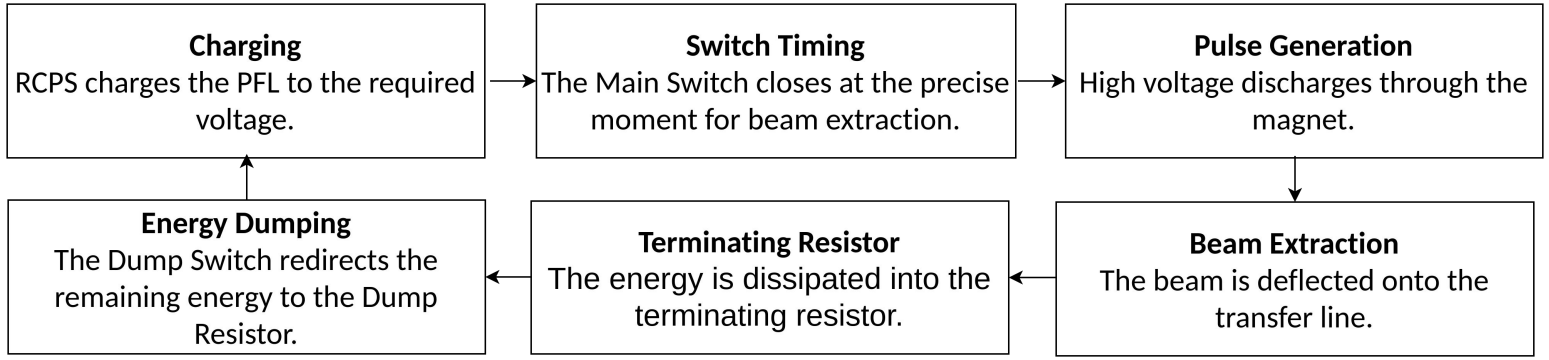




PFL: Pulse Forming Line

RCPS: Resonant Charging Power Supply

Schema: *Simplified schematic of a kicker module*[1]



[1] M.J. Barnes, *CAS: Beam Injection, Extraction & Transfer 2 Kicker Systems - Part 1 - Introduction and Hardware,* CERN TE/ABT

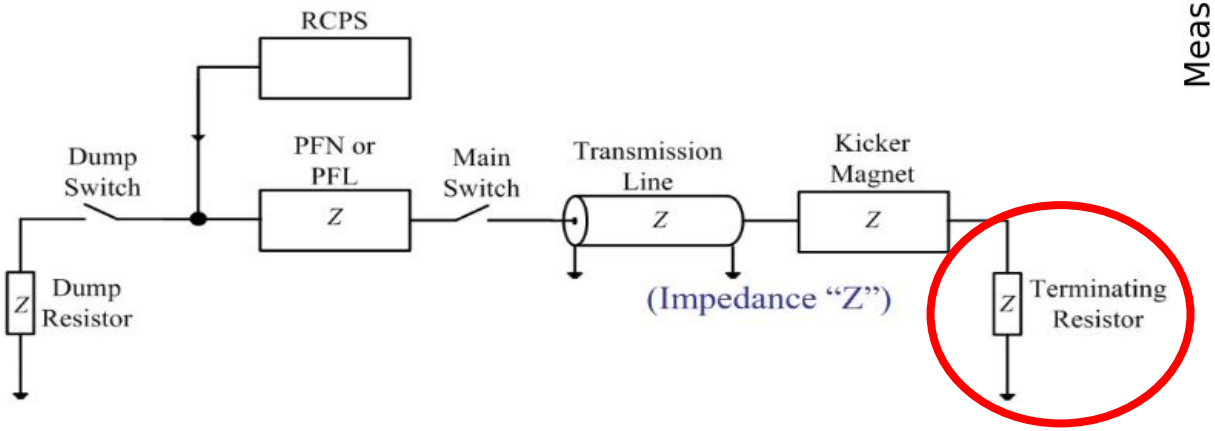
System Overview – The KFA71/79 Extraction Kicker

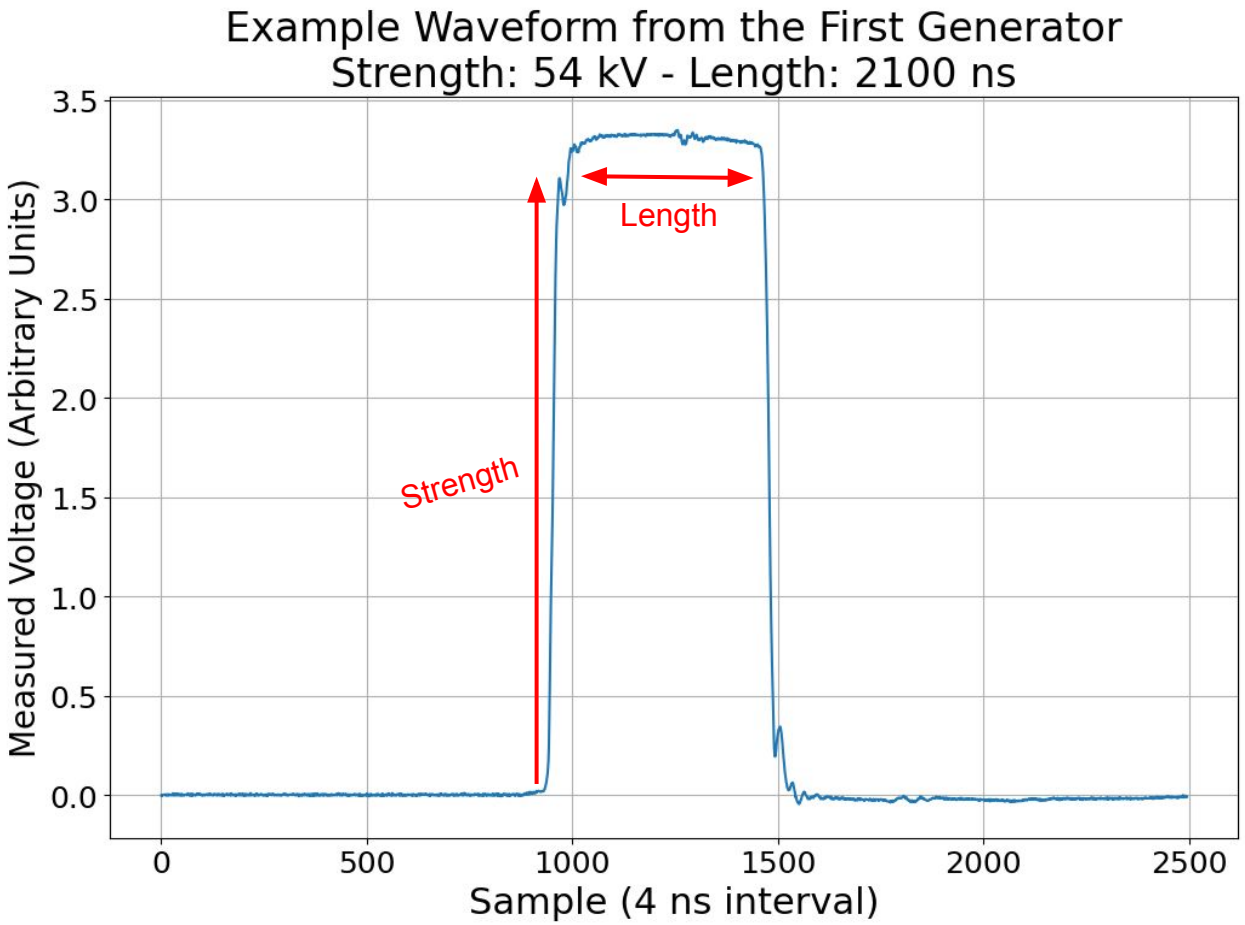
*Picture: CPS **PFL** DRUM Winding 2005*

Data Description (1)

Waveform Characteristics:

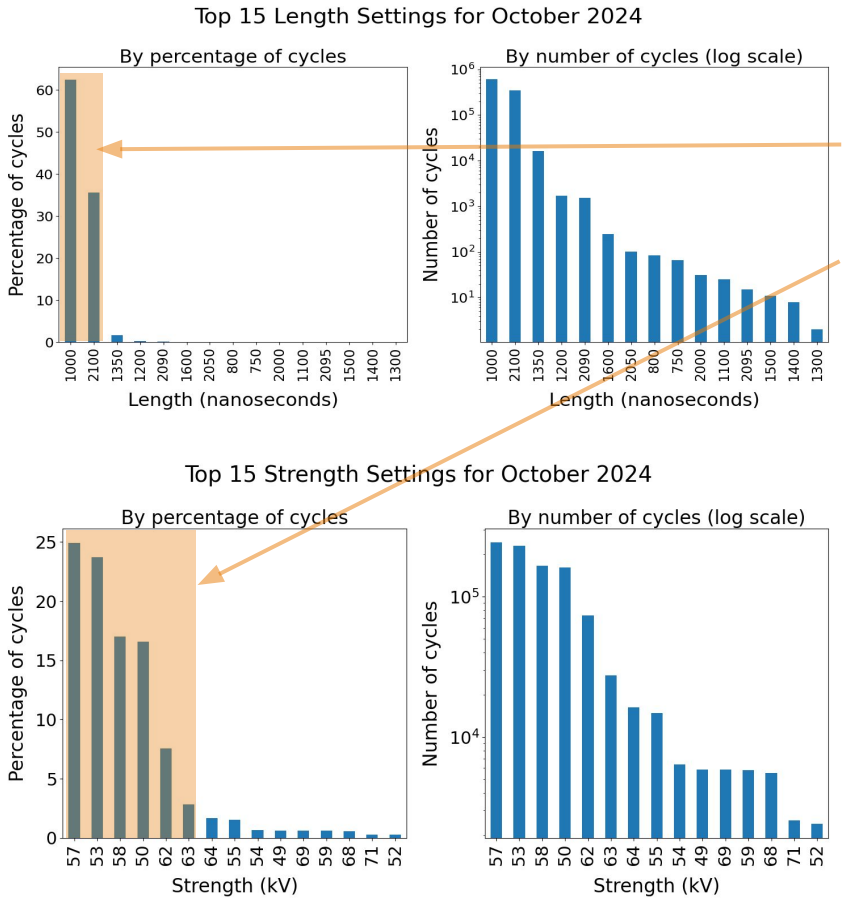
- ***Measured Voltage on the Terminating Resistor***
- ***Sampling Rate***: 1 sample every 4 ns for 10 µs → 2500 data points
- ***Signal Content***: Short rise and fall times, short plateau region.
- ***Pulse Settings***: Includes desired pulse **strength**, pulse **length**, enabled generators, etc.





Data Description (2)

Data Imbalance:



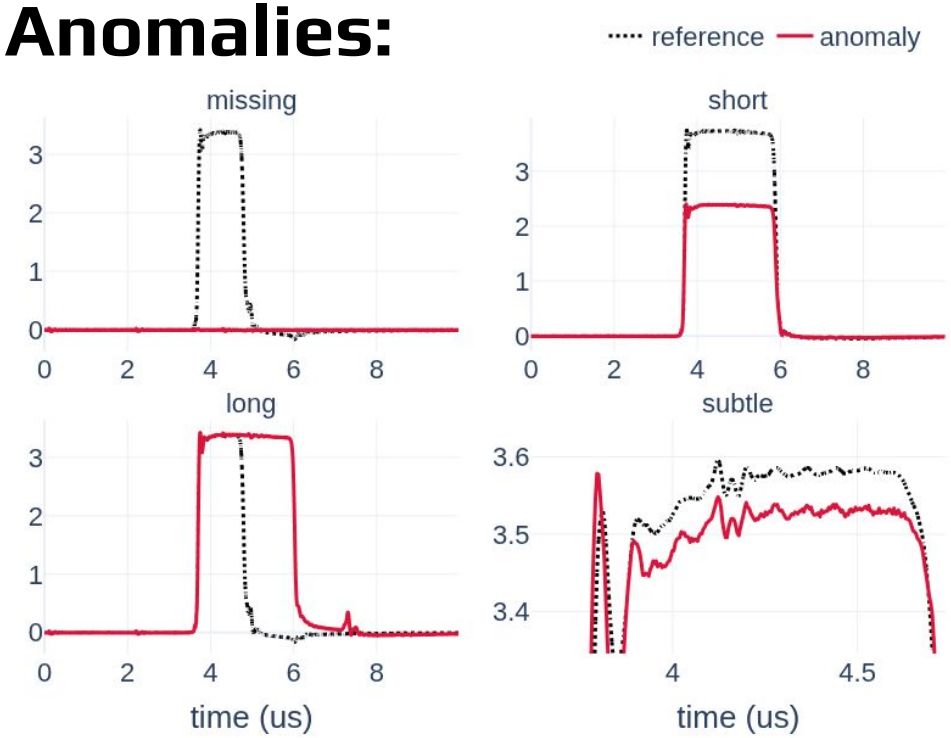
Settings distribution:

- **~98%** of waveforms share 2 length values.
- **~80%** of waveform share 6 voltage values.

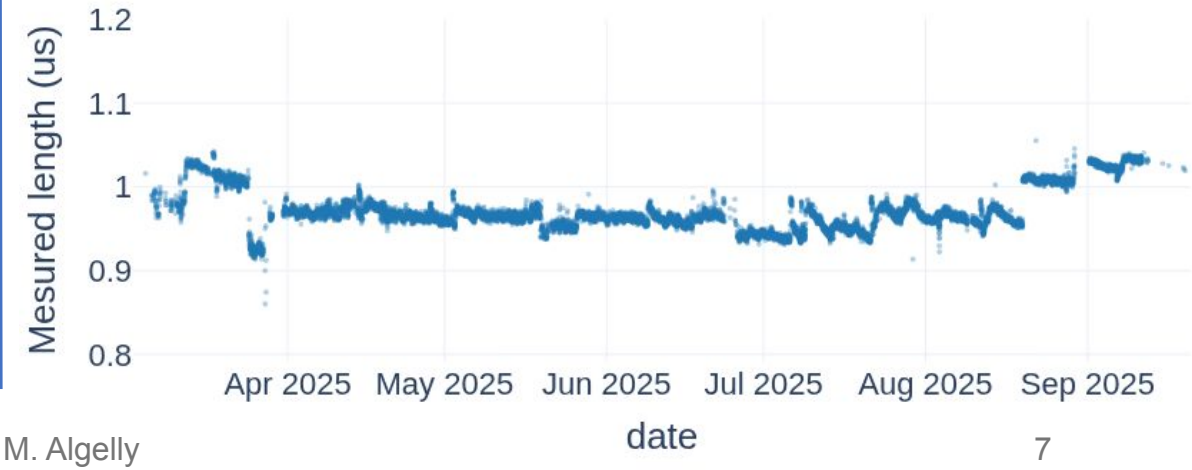
Consequences:

- Rare configurations **misclassified** as anomalies.
- Reduces detection accuracy and increased biases.

Anomalies:



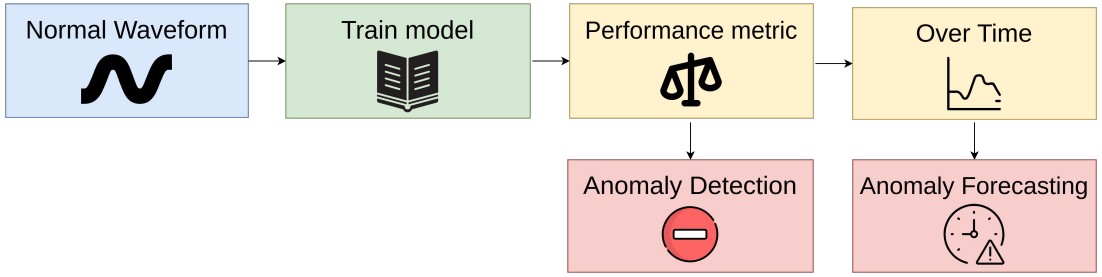
Length drift (1000 ns):



Anomaly Detection & Forecasting

General Idea:

1. Train a model on nominal waveforms.
2. Detect deviations using performance metrics.
3. Monitor trends to identify drifts or early anomalies.



Mathematical Formulation[1]:

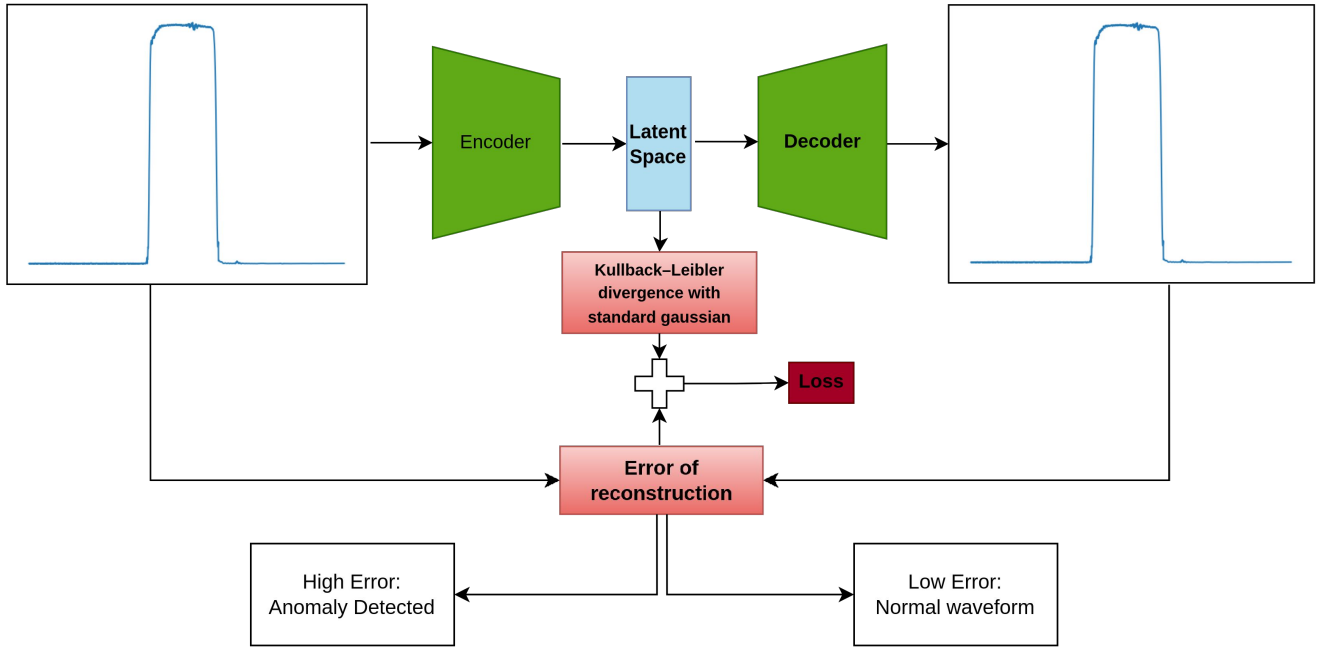
$$\mathbb{E}[L] = (1-\epsilon)\cdot\mathbb{E}[l_w(X_{good})] + \epsilon\cdot\mathbb{E}[l_w(X_{anom})]$$

- $l_w(X)$: Loss for waveform X with model's weights w.
- ϵ: Fraction of anomalies.

[1] F. Huhn et al., *Automating beam dump failure detection using computer vision*, IPAC 2023 doi:10.18429/JACoW-IPAC2023-THPL017

Model: VAE Components

- **Encoder**: Maps data to a compressed probabilistic representation. Output two vector: μ & σ^2.
- **Latent Space**: Probabilistic representation: $z \sim \mathcal{N}(\mu, \sigma^2)$.
- **Decoder**: Reconstructs data: $\text{Decoder}(z) = \hat{x}$
- **Loss**: Mean Squar Error + Kullback-Leibler divergence with $\mathcal{N}(0, \mathcal{I})$



Conditional Variational Autoencoders (CVAE)[1]

Contextual Data:

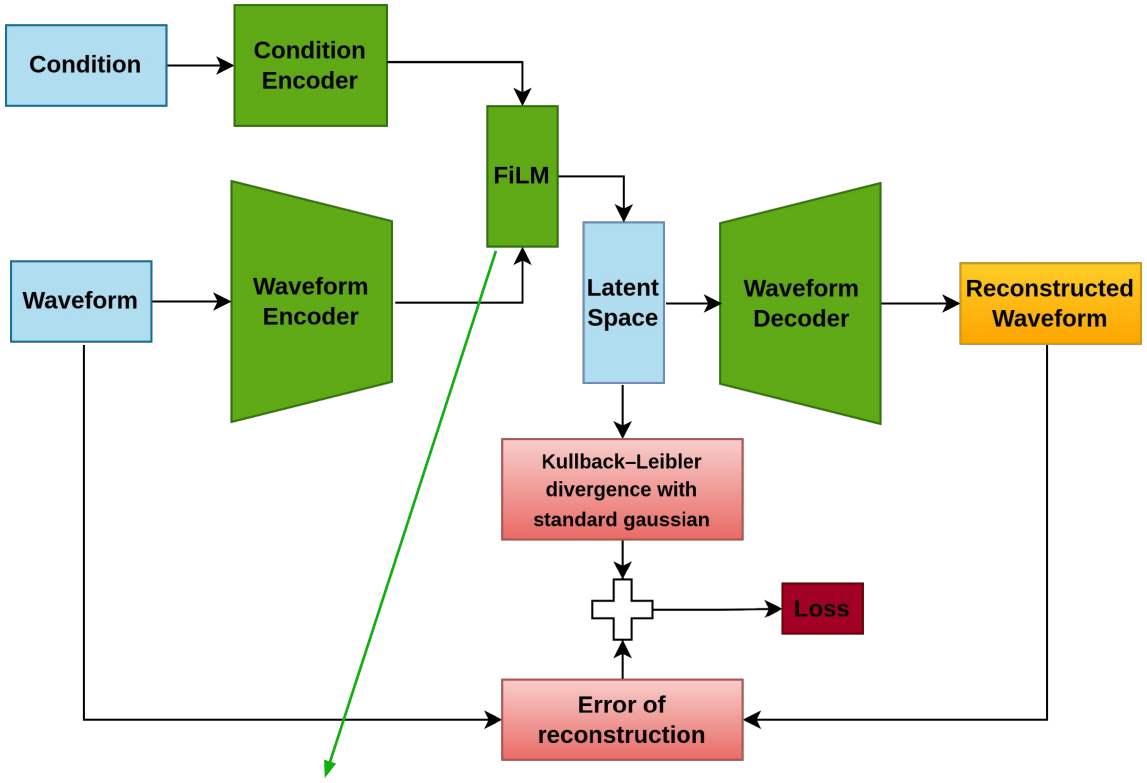
Use contextual information: **Settings** (e.g., strength, length)

Specificities:

- **Condition Encoder:** Encodes condition into its own latent space.
- **FiLM-style Fusion**[2]**:** Modulates waveform latent using condition (**γ**, **β**).
- **Latent Awareness:** Conditioning modulates the latent space using context, which could help encode condition-related structure

Advantages:

- **More general model:** More robust reconstruction, especially for rare pulses or unseen pulses.
- **Improved Robustness:** Higher anomaly score for subtle anomalies.



Feature-wise Linear Modulation

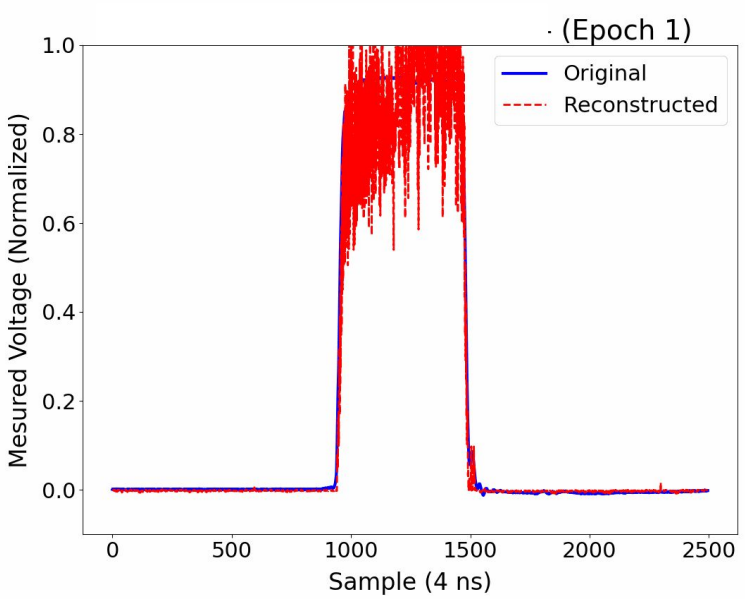
$$\mu = \gamma_\mu(\mu_{cond}) \cdot \mu_{wave} + \beta_\mu(\mu_{cond})$$

$$\sigma^2 = \gamma_\sigma(\sigma^2_{cond}) \cdot \sigma^2_{wave} + \beta_\sigma(\sigma^2_{cond})$$

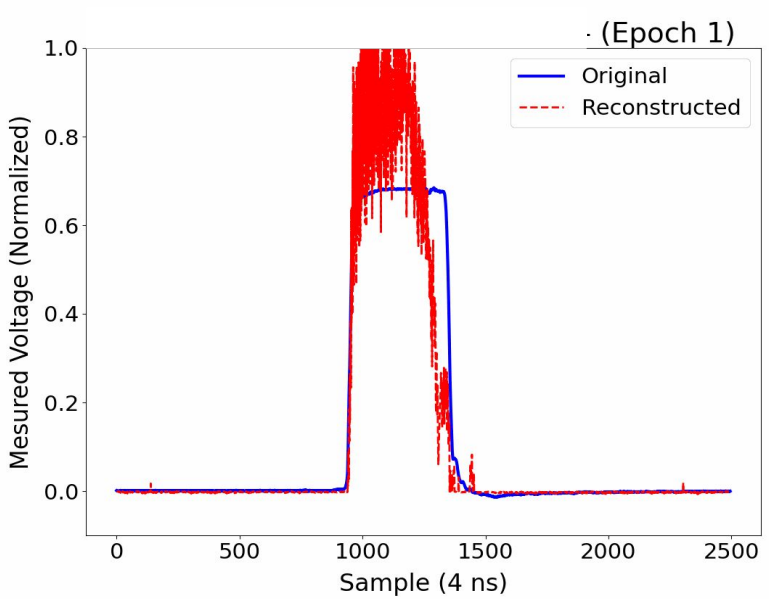
[1] Pol et al., *Anomaly Detection With CVAE*, arXiv:2010.05531 (2020)
[2] E. Perez et al., *FiLM: Visual Reasoning with a General Conditioning Layer*, arXiv:1709.07871 [cs.CV], 2017. doi:10.48550/arXiv.1709.07871

Results – Validation

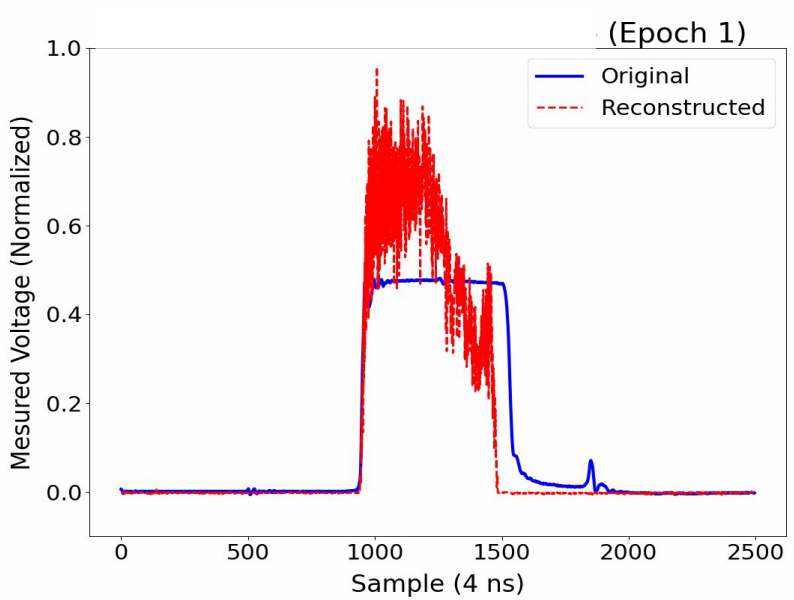
Normal waveform



New settings waveform

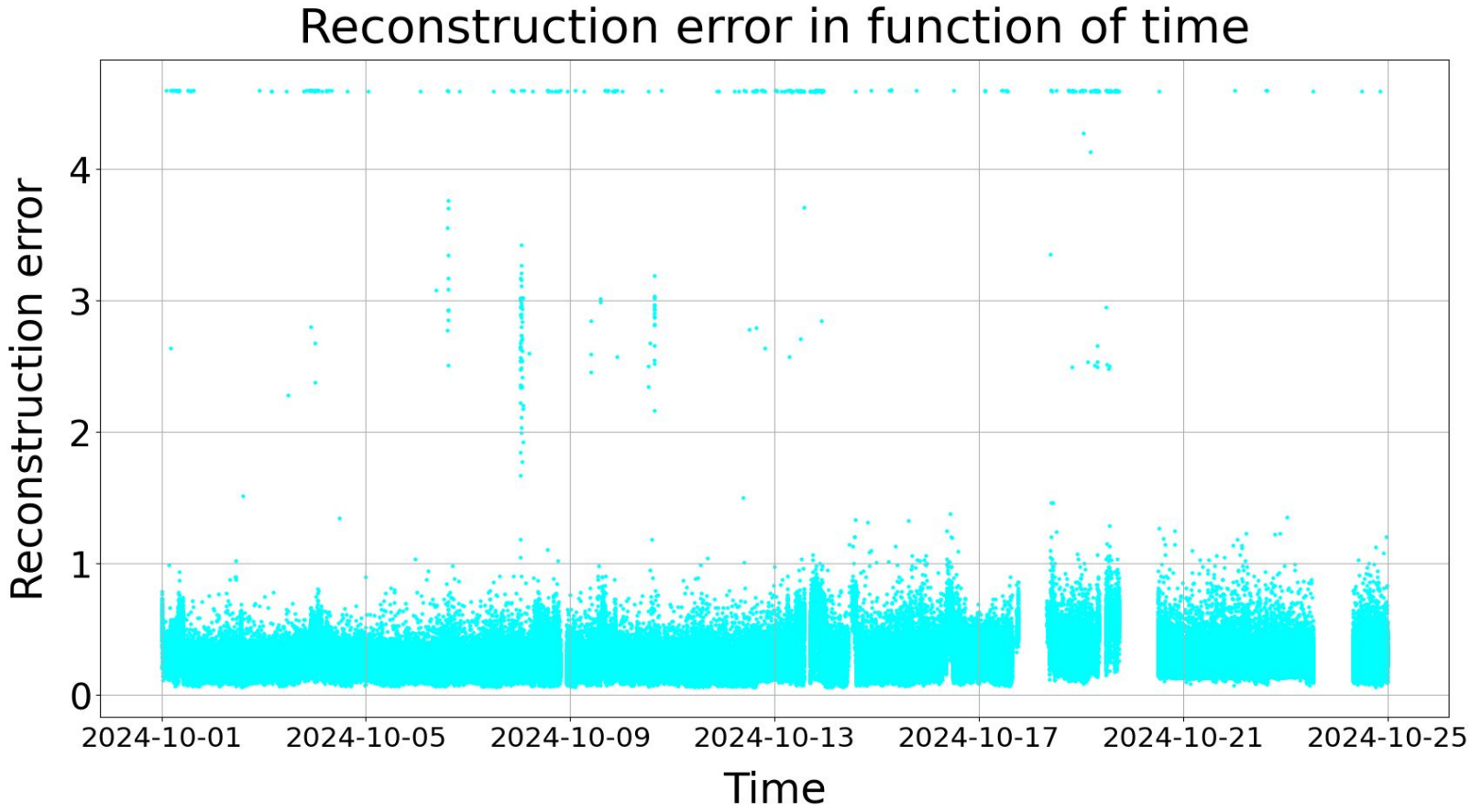


Anomalous waveform



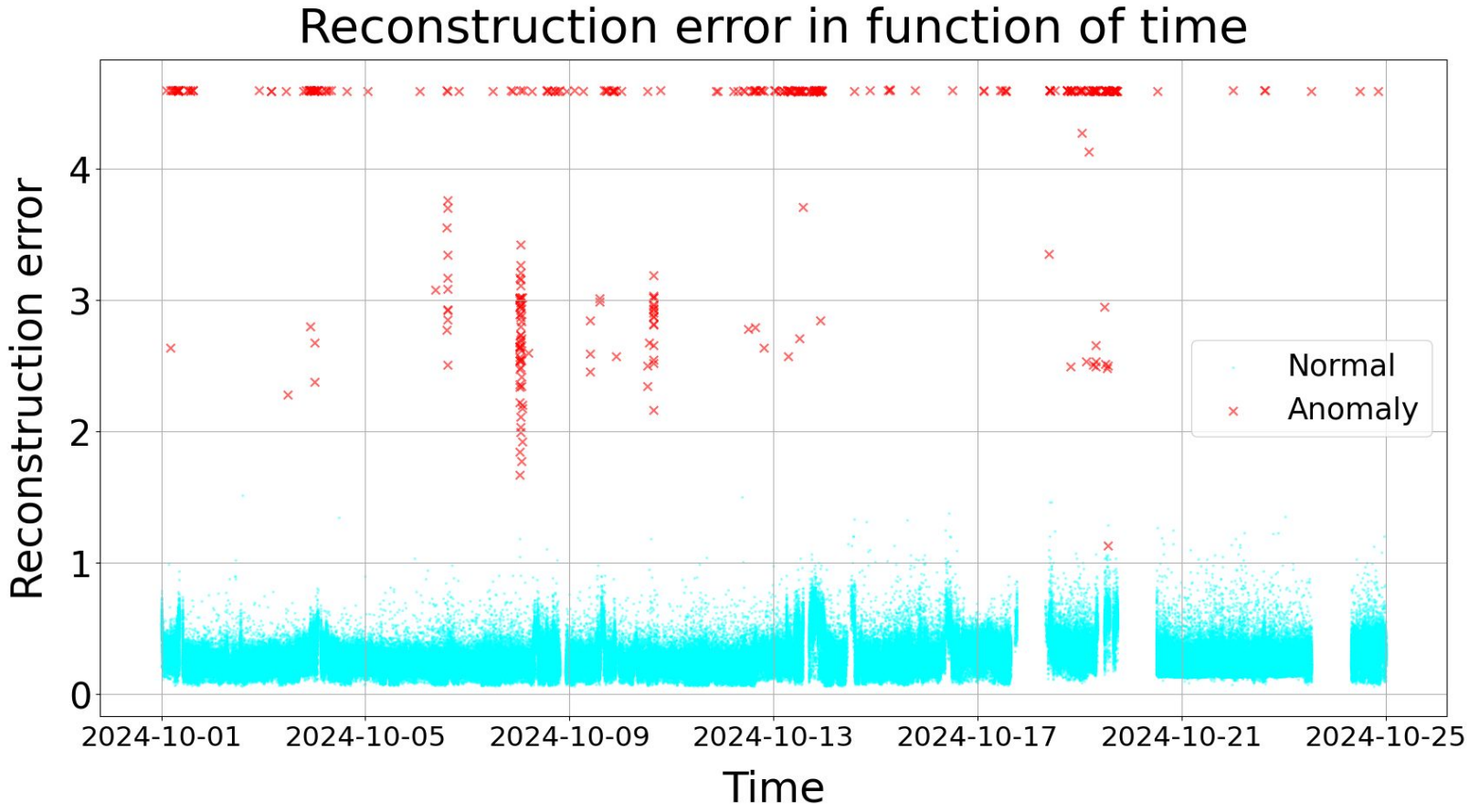
Reconstruction errors

- **High Reconstruction Errors**: Observed in some data.



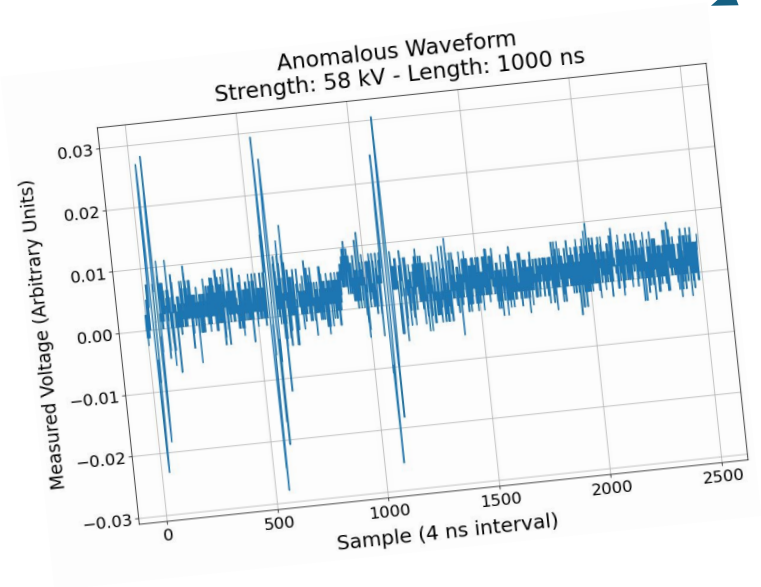
Reconstruction errors

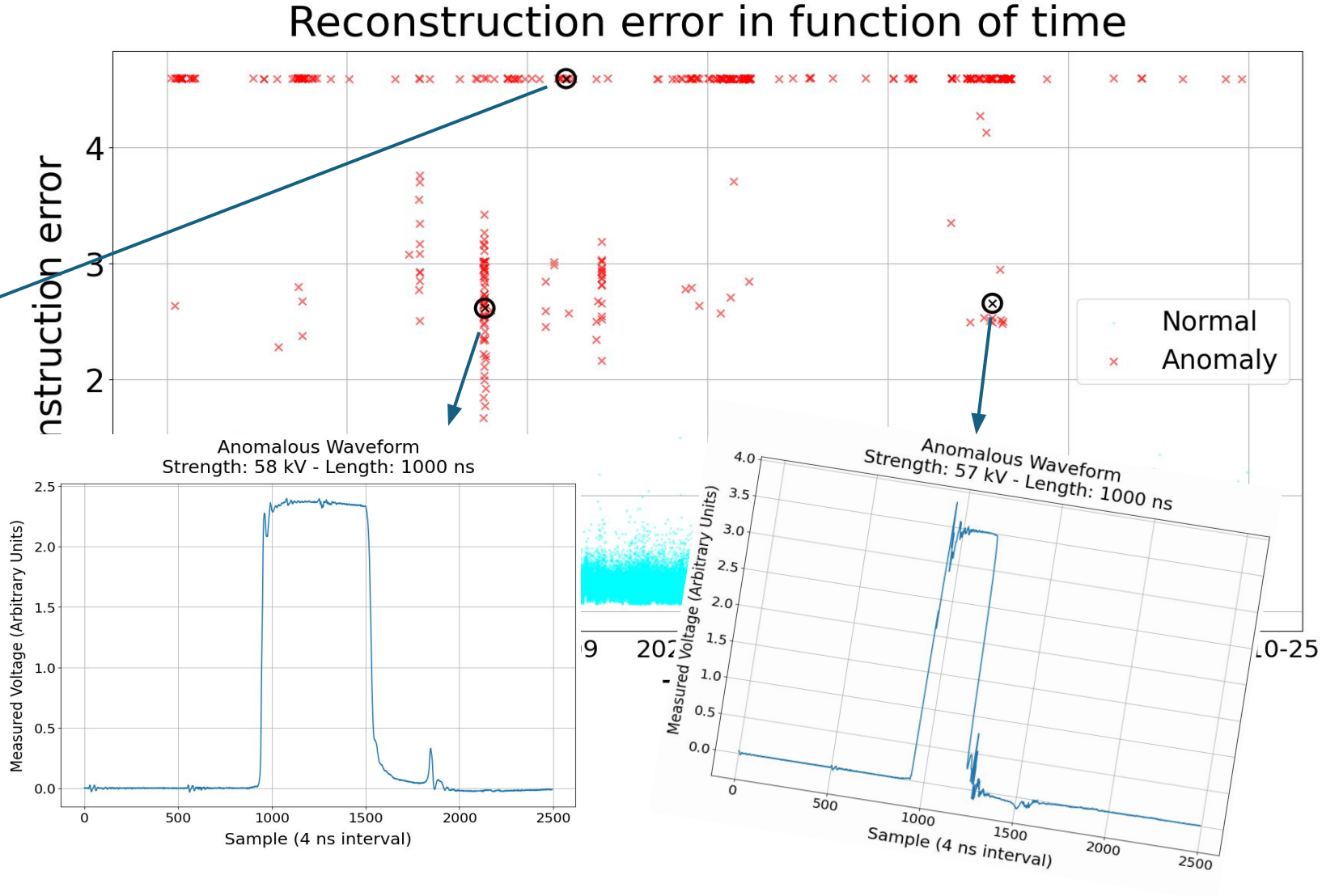
- **High Reconstruction Errors**: Observed in some data.
- **Anomalies**: Associated with high errors.



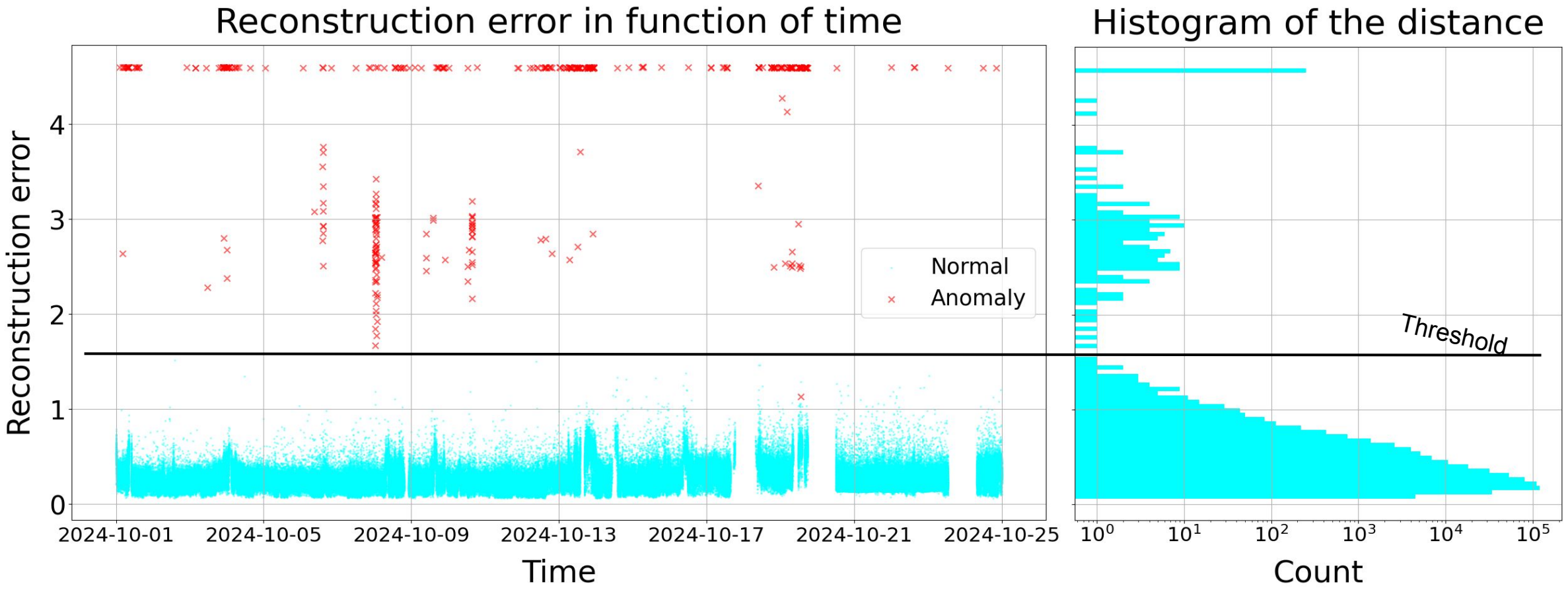
Reconstruction errors

- **High Reconstruction Errors**: Observed in some data.
- **Anomalies**: Associated with high errors.



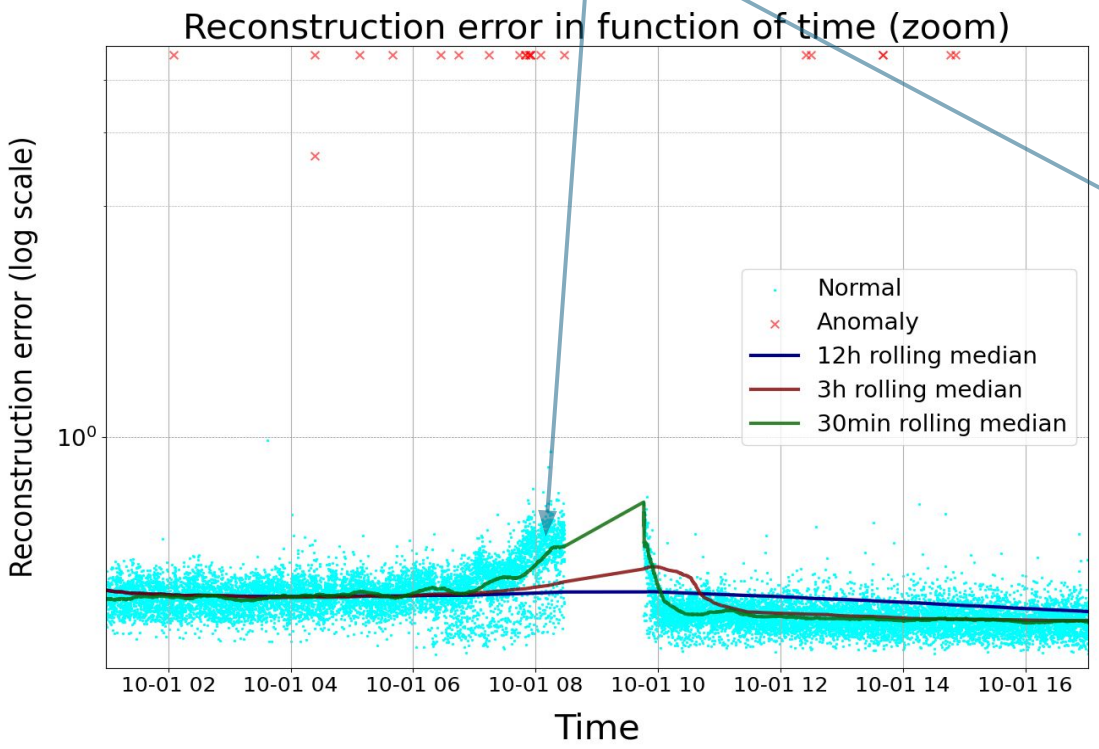


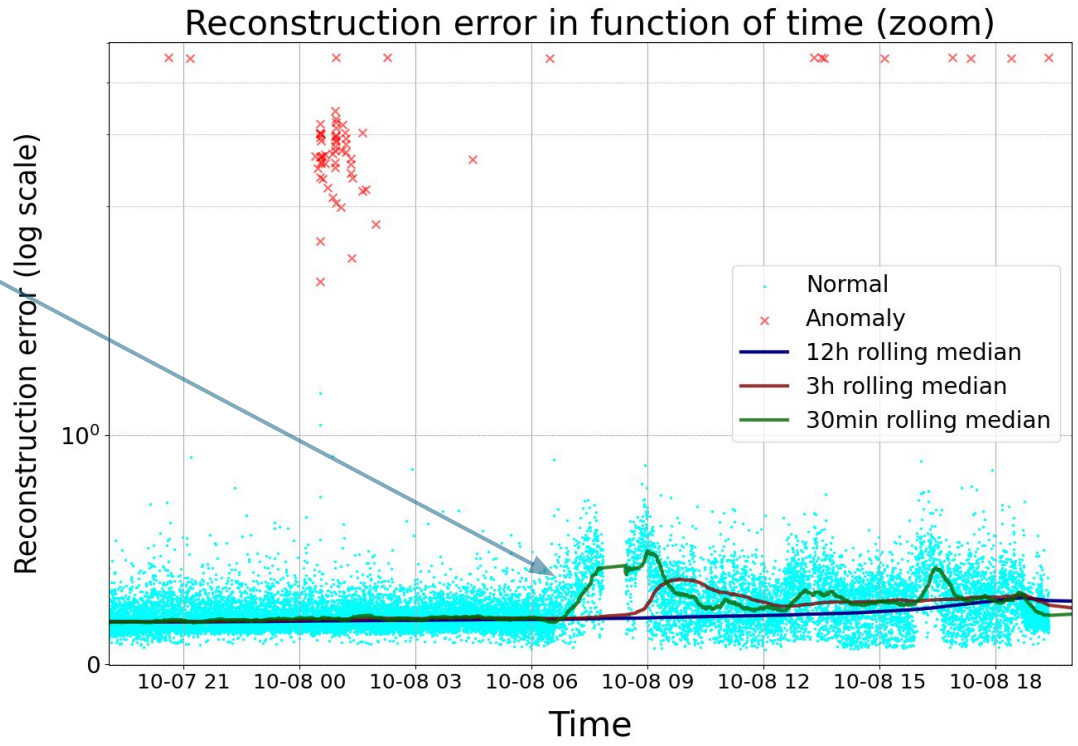
Reconstruction errors



Anomaly Forecasting

Observation: **Gradual rise** in reconstruction error before failures.

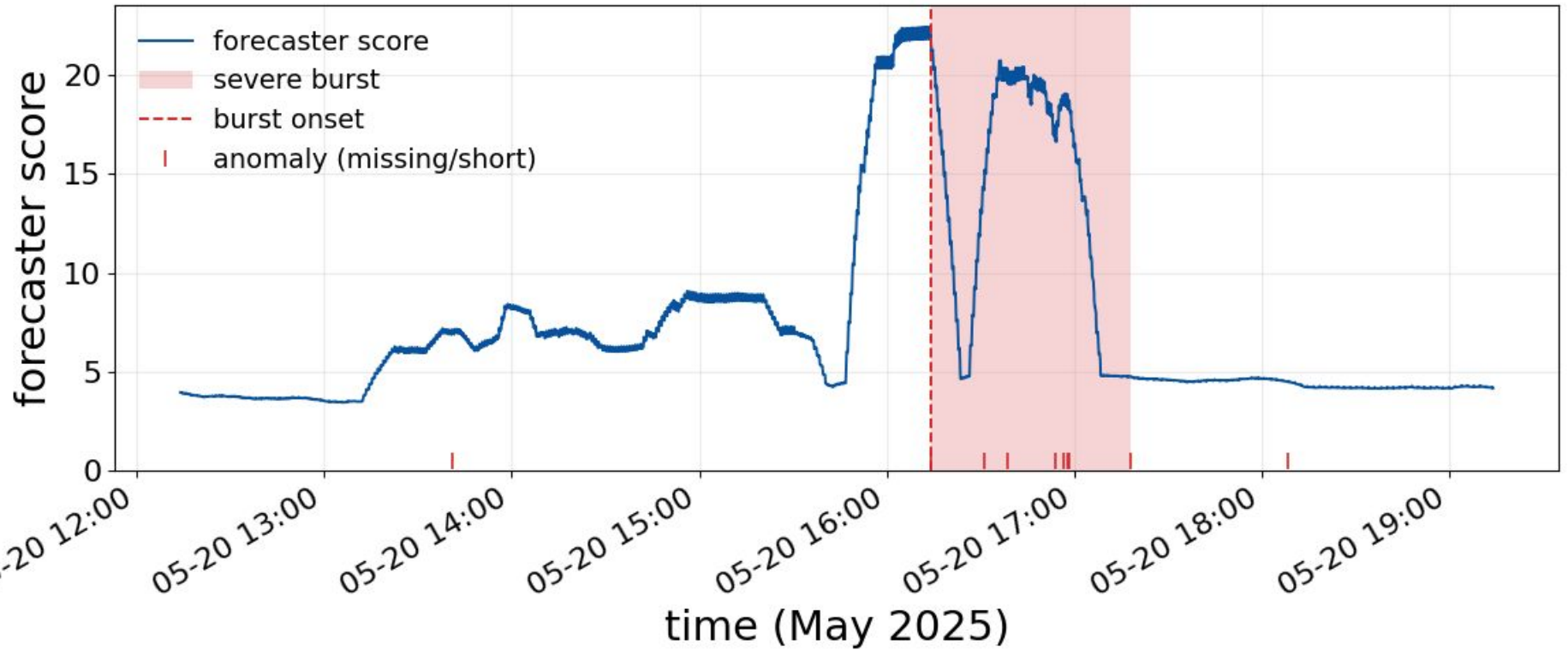




Anomaly Forecasting

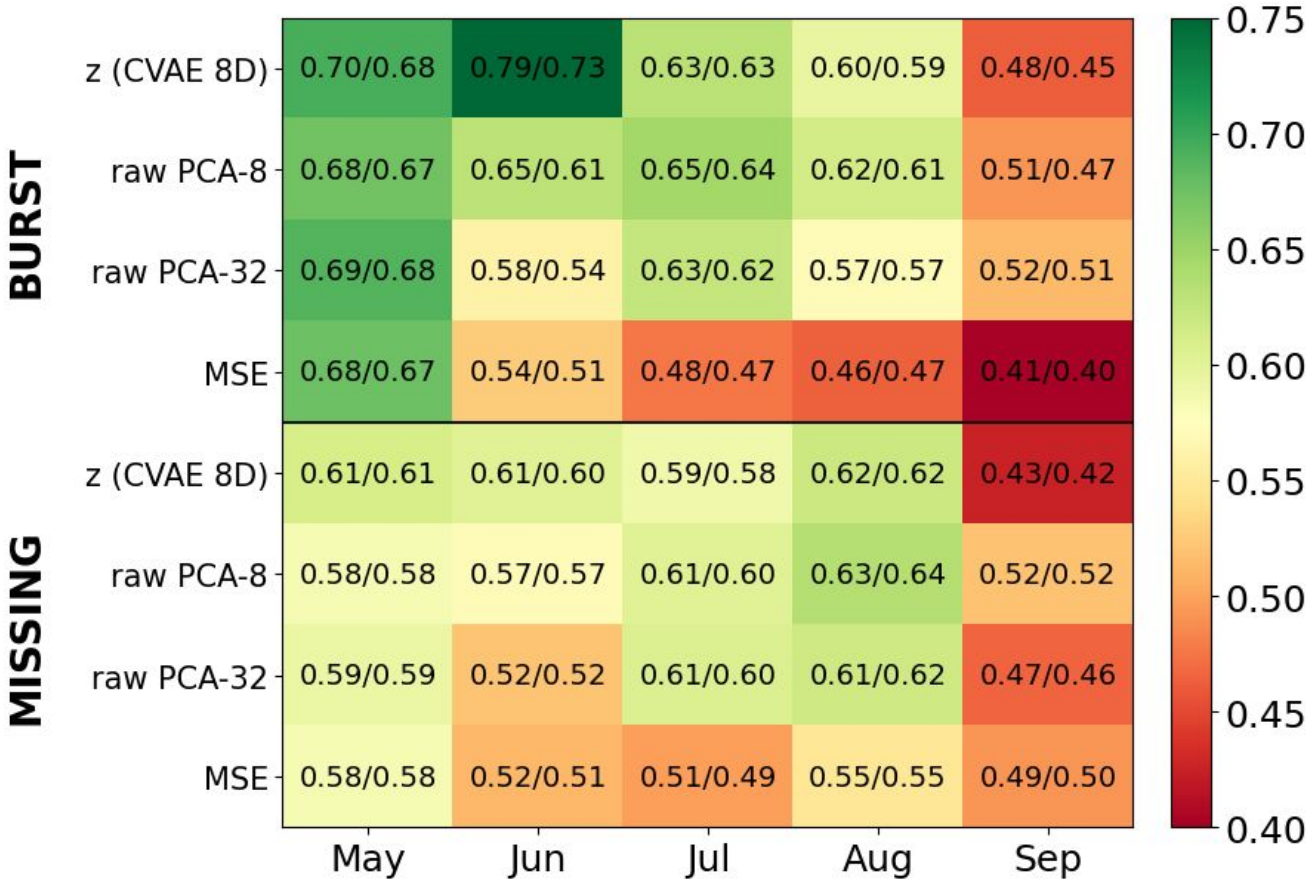
One-class novelty detection in the CVAE latent space [1]

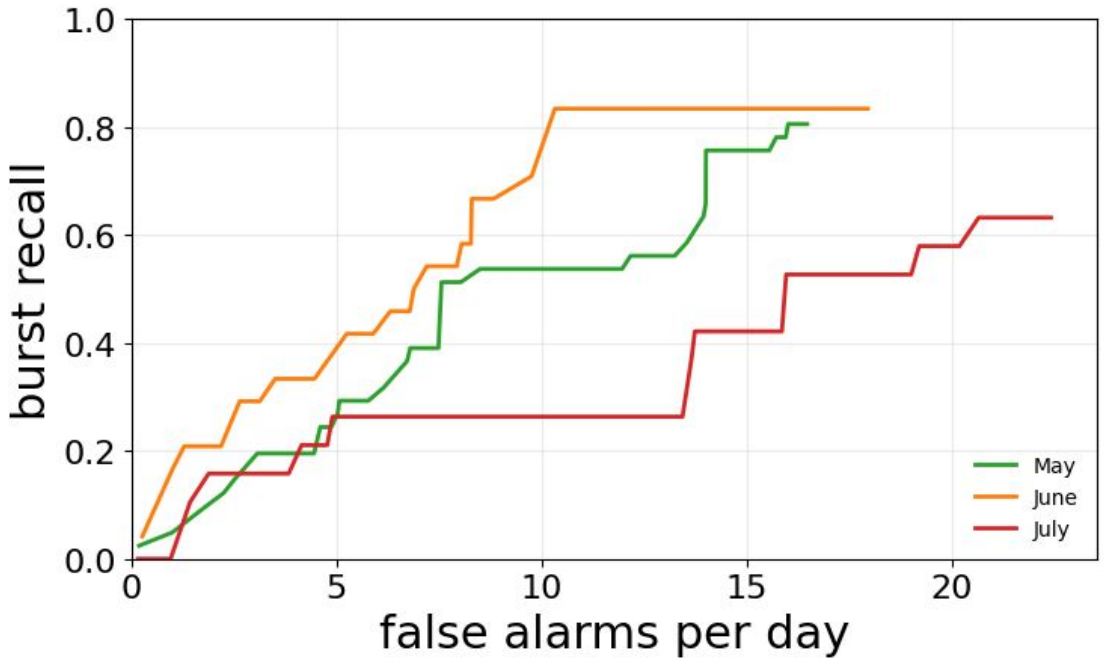
- Trained on normal pulses only (anomalies removed)
- Each pulse → latent vector z (CVAE encoder)
- Score = Mahalanobis distance of z to a fixed normal reference (April), 10-min smoothed
- Rising score → onset predicted 10 to 30 min ahead



[1] Denouden 2018; cf. Lee 2018, DAGMM

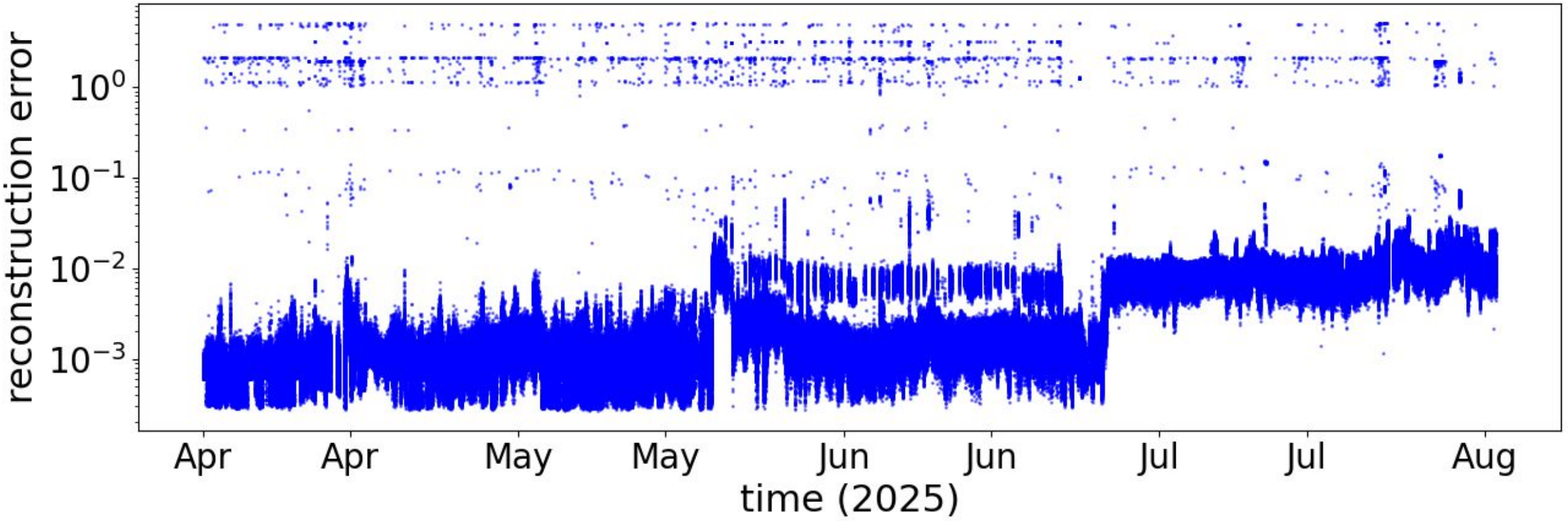
Anomaly Forecasting





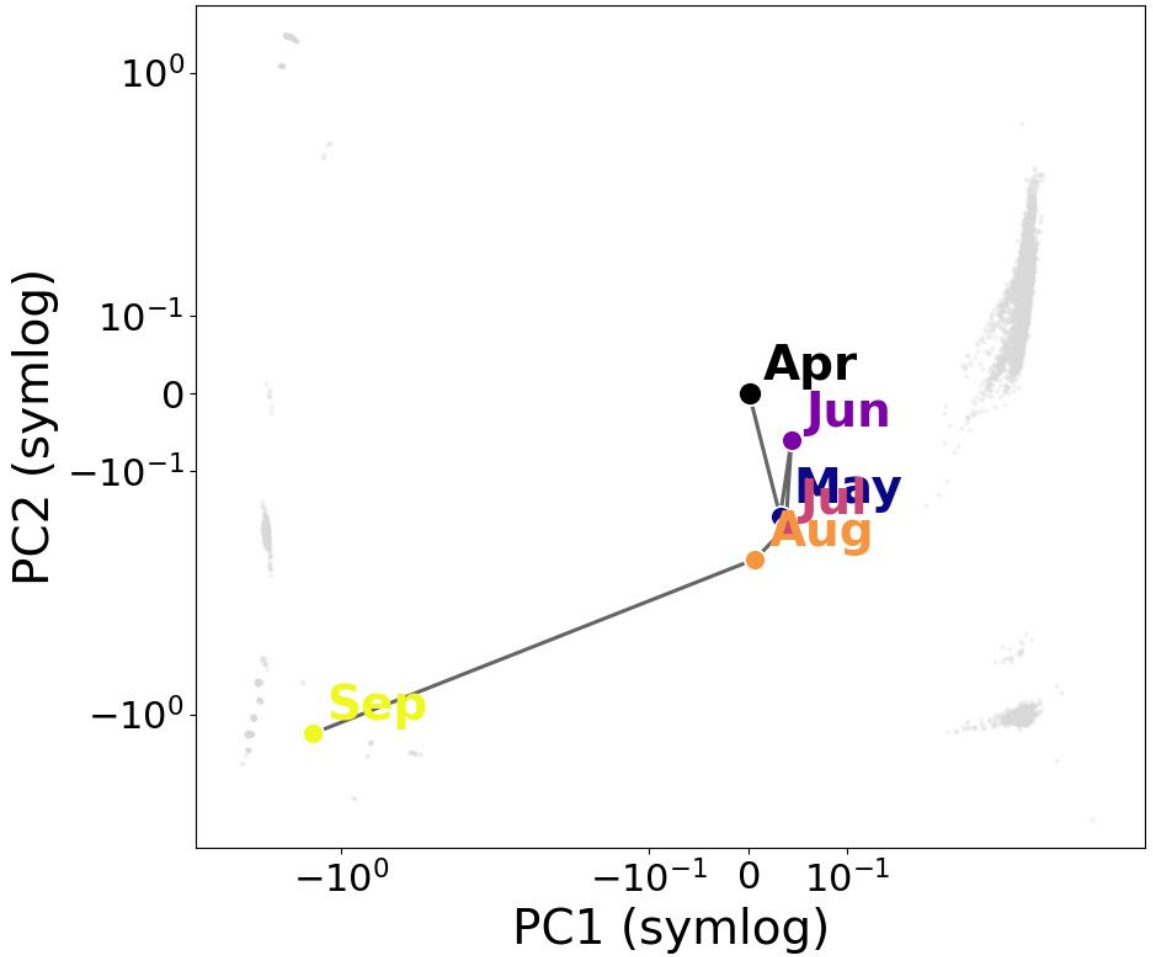
Anomaly Forecasting

The system's drifting, the model's getting outdated...



Anomaly Forecasting

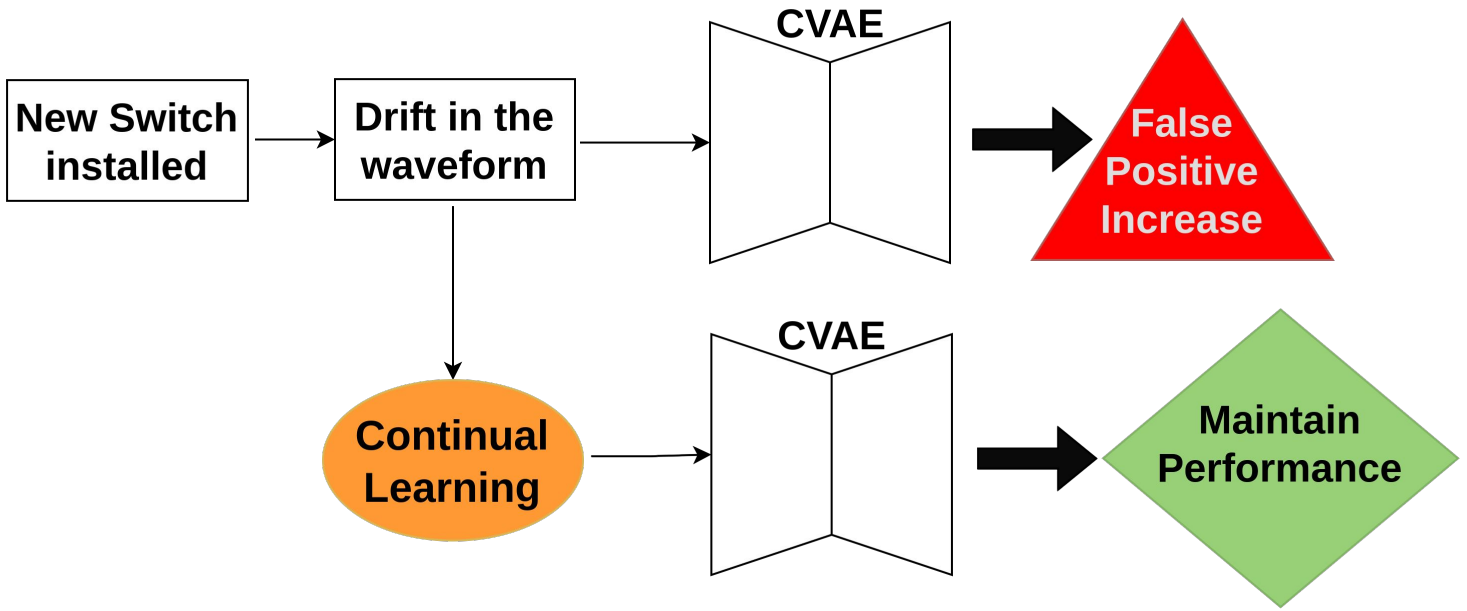
… and so is the centroid of our latent space.



Continual Learning

What is continual learning?

- Standard ML: train once on a fixed dataset, then **freeze** and deploy.
- Continual learning: keep learning from a **stream without retraining from scratch** and **without storing all past data**.
- The catch: each update tends to **overwrite** what was learned before.



Continual Learning

Two families, opposite goals

the stability ↔ plasticity trade-off

Anti-forgetting (lifelong learning)

- The past stays valid
- Goal: do **not** forget
- One-way control: protect old knowledge
- → most of the field
  (Parisi 2019; De Lange 2021)

Tracking the present (concept drift)

- The past goes **stale**
- Goal: **forget** the stale, follow the current normal
- → **our setting**
  (Gama 2014; Lu 2018)

The toolbox (anti-forgetting)

- **Regularization**: protect the weights that matter
- **Replay / rehearsal**: keep or regenerate old samples
- **Architecture**: dedicate parameters per task

Toolbox (concept drift)

- **Drift detection**: flag when the stream changes
- **Windowing / forgetting**: keep recent data, decay the old
- **Ensembles**: add / drop / reweight learners on drift

Continual Learning

Initialise the normal cloud on April normals

$$\mu_0 = \text{mean}(z), \qquad \Sigma_0 = \text{cov}(z)$$

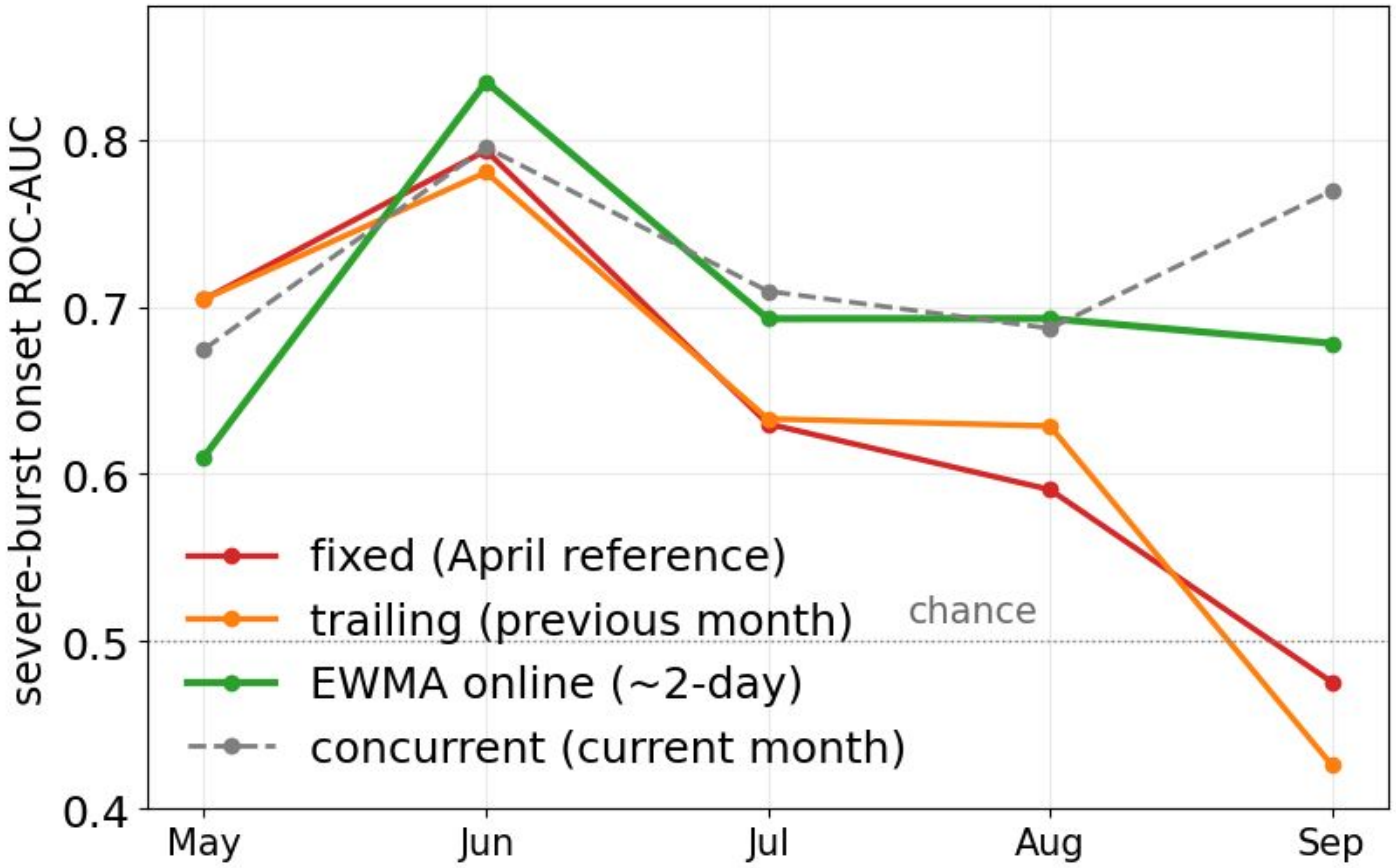
Score of pulse z_t with the current reference (Mahalanobis)

$$d_t = \sqrt{(z_t - \mu)^\top \Sigma^{-1} (z_t - \mu)}$$

Update on each normal pulse only $(\delta = z_t - \mu)$*:*

$$\mu \leftarrow \mu + \alpha\,\delta$$

$$\Sigma \leftarrow (1 - \alpha)(\Sigma + \alpha\,\delta\,\delta^\top)$$



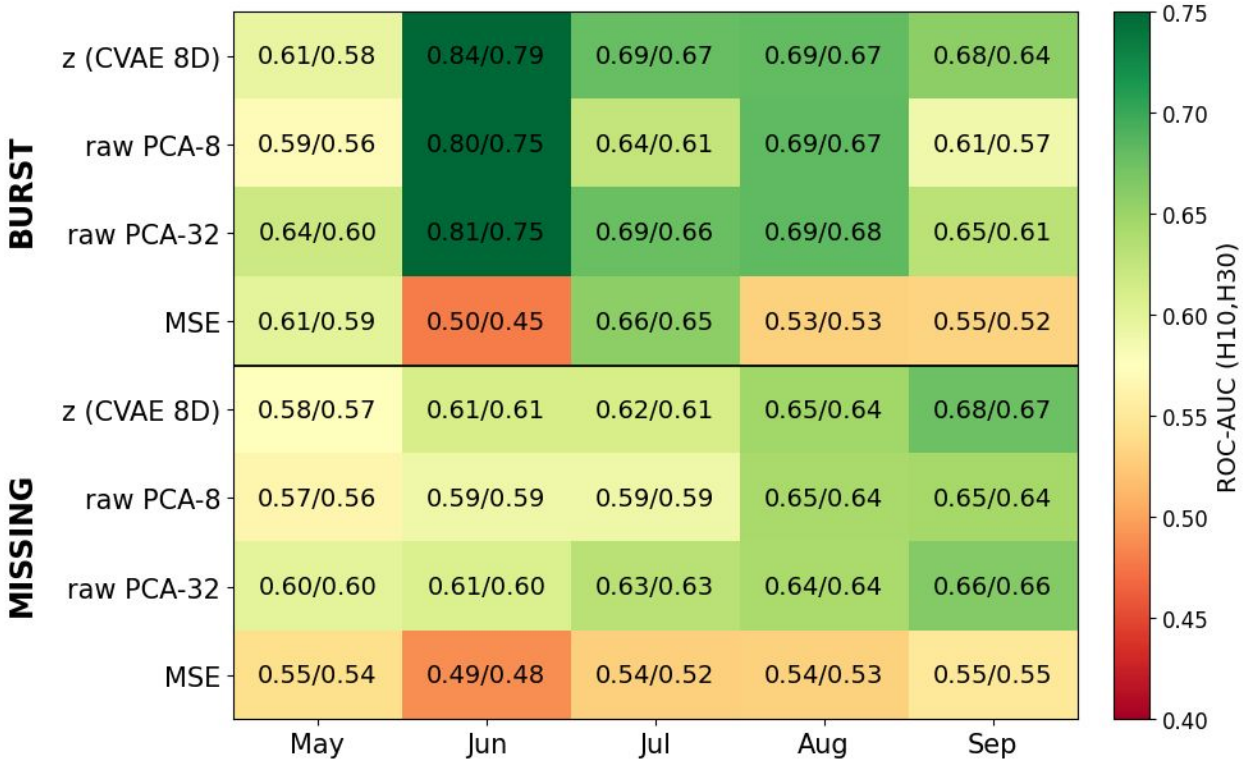
α = forgetting factor (effective memory $\approx 1/\alpha$ pulses; $\alpha = 2 \times 10^{-5} \approx$ 2 days)

Tony Finch. Incremental calculation of weighted mean and variance. University of Cambridge, 4(11-5):41–42, 2009.

Continual Learning

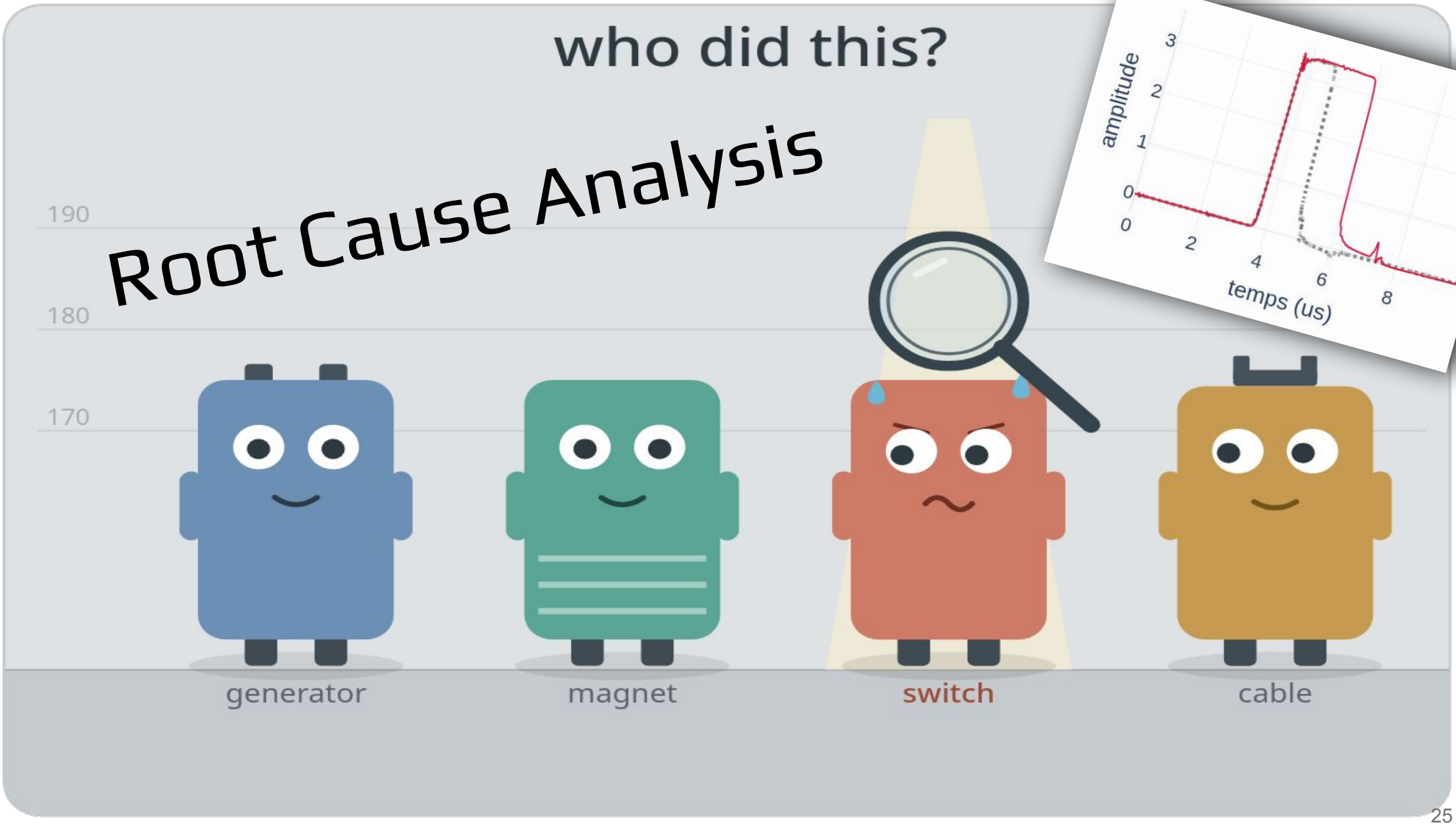
Without EWMA update

With EWMA update



Root Cause Analysis

Root Cause Analysis



WORKFLOW: OPTIMIZATION & ROOT CAUSE ANALYSIS

Create SPICE model of KFA71 generators

Generate data by varying key parameters

Train NN surrogate on generated data

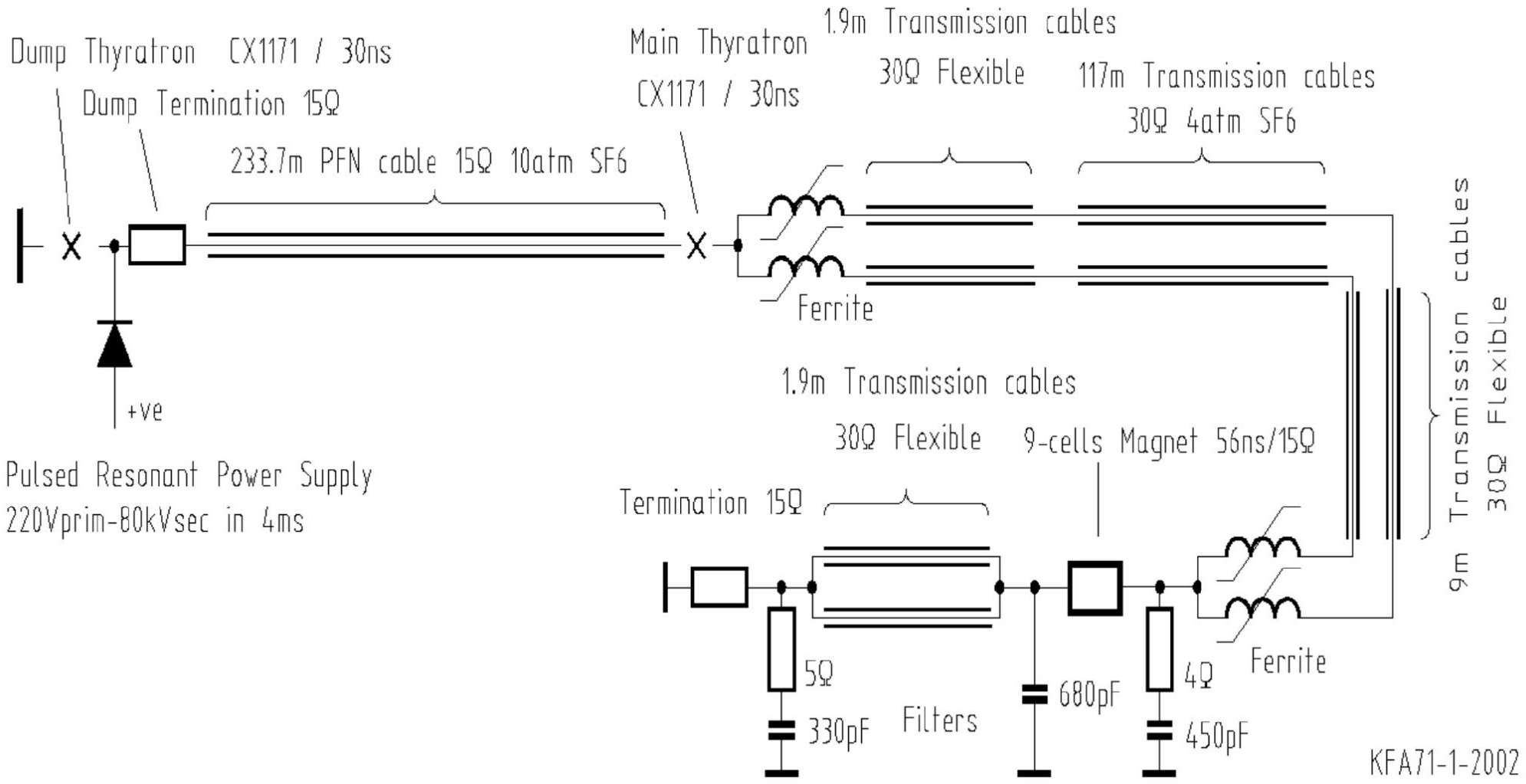
Optimize parameters (surrogate + MCMC + real data)

Root cause analysis with optimized SPICE + surrogate

WORKFLOW
STEP 1

Create SPICE model of KFA71 generators

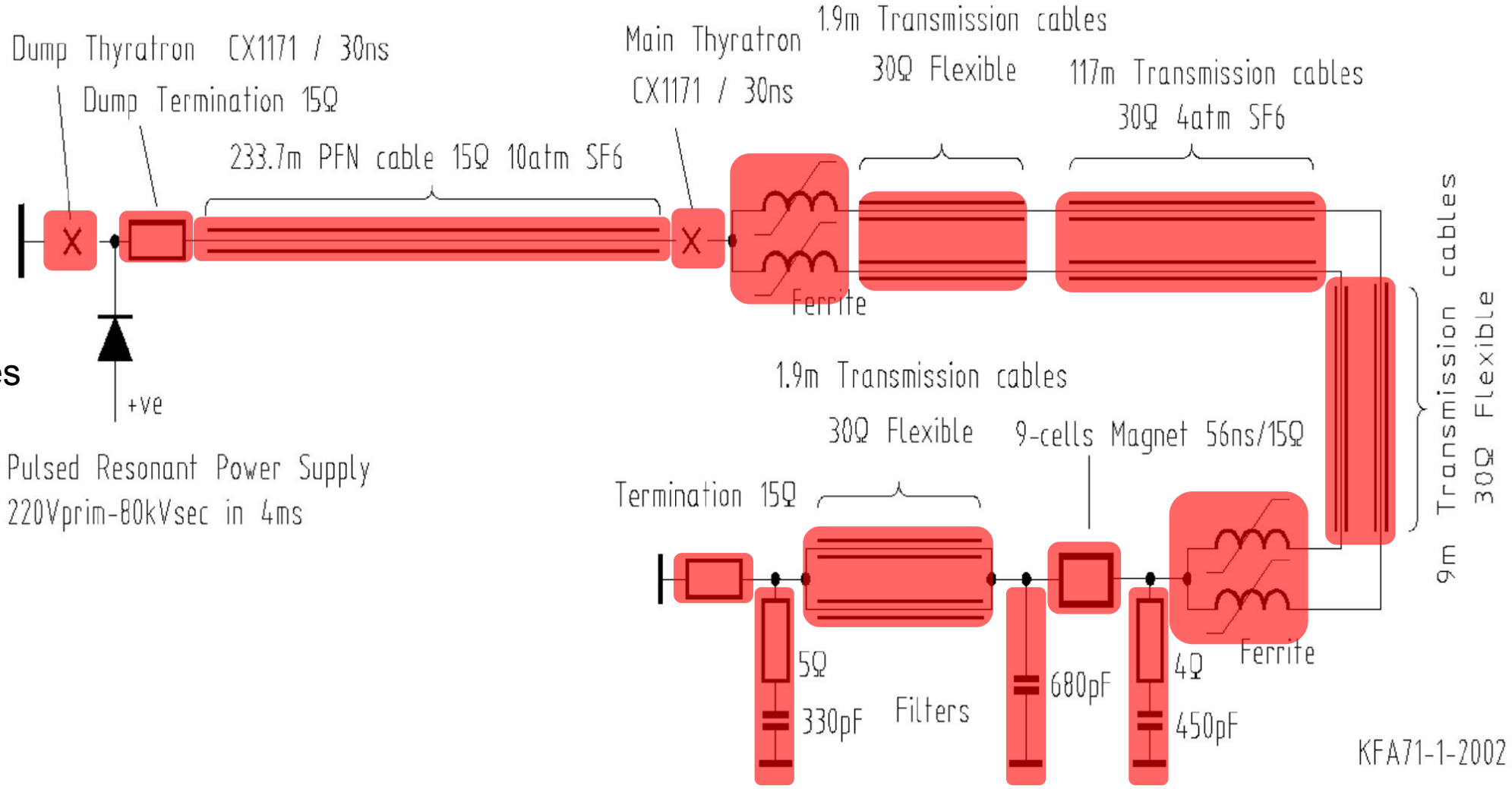
SCHEMATIC – KFA71 GENERATOR



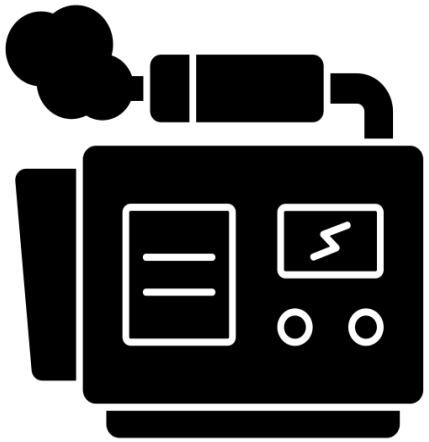
STEP 1: SPICE MODEL

Modelized:

- 2 Thyratrons
- 1 PFL
- 2 Ferrites
- 4 Transmission cables
- 1 Magnet
- 2 Terminations
- 3 Filtres



WORKFLOW
STEP 2



Generate data by varying key parameters

STEP 2: PARAMETER & GENERATION

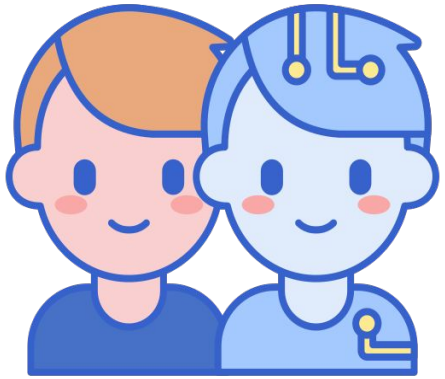
Parameter selection: from 58 to 27

Subsystem	Parameters
PFL	LEN, LOSS_DC, LOSS_AC
Source filters	R, C, L (× 2 filters)
Transmission lines	Z_branch, LEN, R_DC, loss, return Z_branch, return TD
Image filter (output)	R/C left, C center, R/C right
Magnet (kicker)	Z0, TD
Trigger timing	T_TRIG_MAIN, SETTING_LENGTH
Charging	VCH

Dataset:

- ~125k **SPICE** simulations (~20s each)
- **Sobol sequence** over the 27 parameters
- Each simulation → **5000-point** waveform (0-10 us, dt = 2 ns)
- **Training set** for the surrogate model

WORKFLOW
STEP 3



Train NN surrogate on generated data

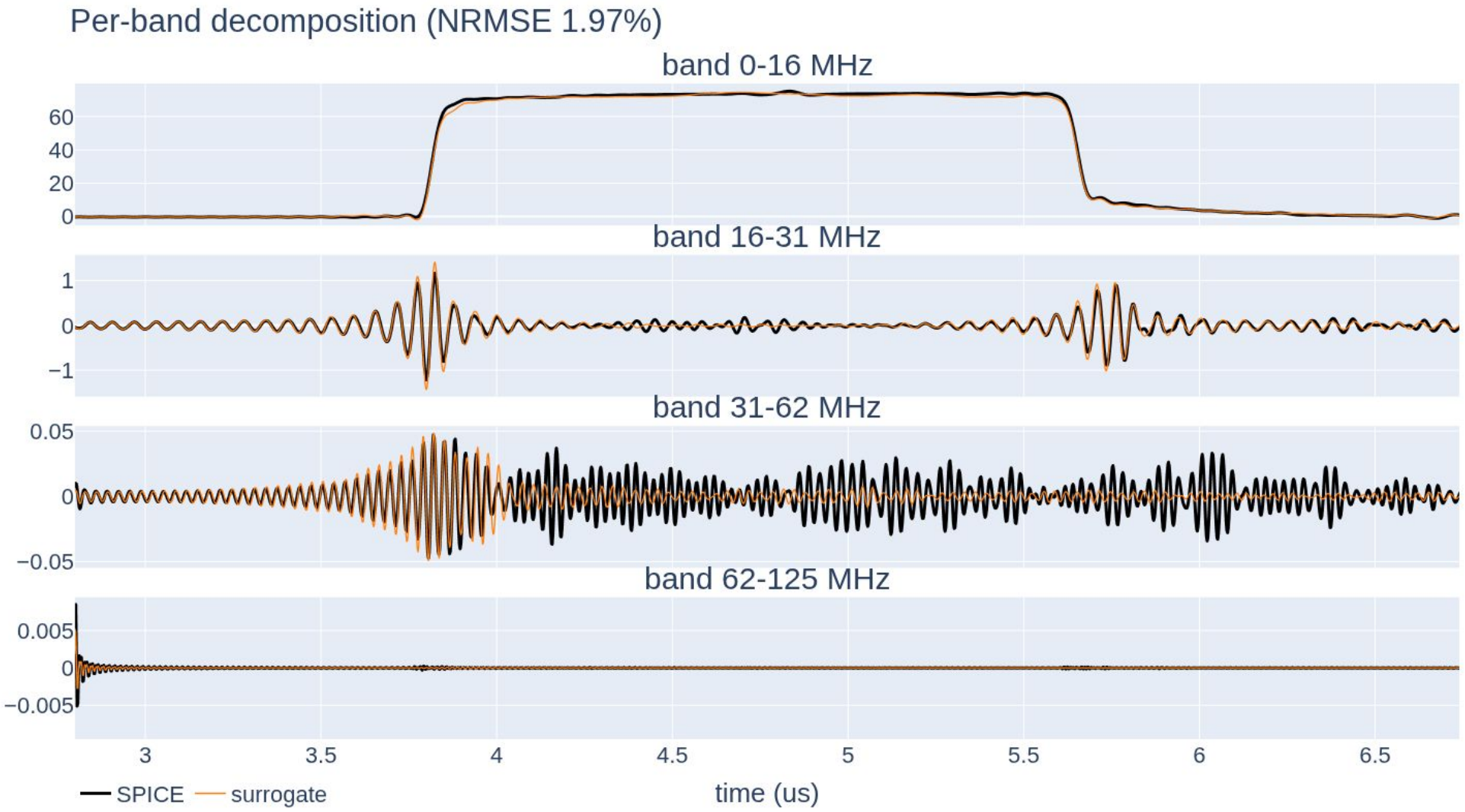
STEP 3: SURROGATE

Input	**27 parameters (standardized)**
Network	Residual MLP: 27 → Linear(384) → *4 × ResBlock* (LayerNorm, SiLU[1,2]) → 327
Output head	**327 multi-band POD coefficients**
Bands	**4 frequency bands**: 0-16, 16-31, 31-62, 62-125 MHz (k = **119/80/64/64**)
Reconstruction	per-band POD inverse (mean + Σ coeff·mode), **fixed linear basis**
Output	**2800-point waveform** (2.8-8.4 µs, dt = 2 ns)
Parameters	**~1 M** trainable
Inference	~1 ms (vs ~20 s ngspice) → ~20 000×
Loss	**MSE on the 327 standardized per-band POD coefficients**

Setting	Value
Hidden dim	384
Residual blocks	4
Activation	SiLU
Optimizer	Adam
LR schedule	ReduceLROnPlateau
Epochs / patience	≤ 600 / 60
Batch size	256
Train/val/test	**~82k / 25k / 18k**

[1] Ramachandran et al. (2017)
[2] Dinh et al. (2024, NeurIPS)

STEP 3: SPICE vs SURROGATE



WORKFLOW
STEP 4

Optimize SPICE parameters (surrogate + MCMC + real data)

STEP 4: Bayesian parameter identification

$$p(\theta \mid \text{data}) = \frac{p(\text{data} \mid \theta) \times p(\theta)}{p(\text{data})}$$

Posterior — Likelihood — Prior

What we want to know:
Parameter estimates
+ uncertainty

What we observe:
NN(θ) vs TMR waveform

What we know:
Nominal values,
physical bounds

Posterior is analytically intractable
→ need to sample it

What is the evidence:
How likely is this data

STEP 4: Prior – What we know

Prior

$p(\theta)$ ⟶ We need to define interval for each parameters

Prior = what we know before seeing the data:

- Physical bounds (a capacitance cannot be negative)
- Nominal values from design documents and datasheets (PSPICE model, …)

In our case: uniform priors within physical bounds
→ "any value in the interval is equally likely"

Examples:

- MAG_Z0 ∈ [13, 17] Ω
- PFL_LEN ∈ [230, 245] m
- C_IMG_FILT_LEFT ∈ [0.1, 0.6] nF

The posterior combines the prior with information from the data:

- Sensitive parameter
  → data narrows the prior
- Insensitive parameter
  → posterior stays as wide as the prior

STEP 4: Likelihood – What we observe

Likelihood

$p(\text{data} \mid \theta)$ ⟶ How well does a set of parameters θ fit the data?

Scoring the fit:

1. Run NN(θ) → predicted waveform
2. Compare to TMR point by point: $r_t = \mathrm{NN}(\theta)_t - \mathrm{TMR}_t$
3. Residuals → need to turn them into a single score: the likelihood
4. We must choose a statistical model for the residuals:
   - Minimising MSE = maximising i.i.d. Gaussian likelihood [1]
   - i.i.d. Gaussian → simple but wrong (residuals are correlated)
   - AR(1) Gaussian → accounts for temporal correlation ?

[1] Bishop (2006), §1.2.5 & §3.1.1

STEP 4: Posterior – What we want to know

Posterior

$p(\theta \mid \text{data})$ ⟶ We cannot compute the posterior, but we can sample it !

Markov chain Monte Carlo (MCMC):

1. Start at θ_current
2. Propose a new point θ'
3. Compute ratio $r = \frac{p(\text{data} \mid \theta') \times p(\theta')}{p(\text{data} \mid \theta_{\text{current}}) \times p(\theta_{\text{current}})}$
   - If r ≥ 1 → accept (θ' is better)
   - If r < 1 → accept with probability r
4. Repeat thousands of times

Two key choices determine performance:
the **likelihood function** (how we score a fit)
and the **sampler** (how we propose new points)

The acceptance rate measures how often proposals are accepted:
Too low → chain is stuck.
Too high → steps are too small.

STEP 4: Sampler – Hamiltonian Monte Carlo (HMC)

In high dimensions, probability concentrates on a thin shell[1]. Random proposals miss it, gradient-guided proposals follow it.

Standard MCMC:
random jumps → inefficient in 27 dimensions

HMC:
uses the gradient of the NN to propose new points intelligently

At each iteration:

1. Give a random kick (random direction and strength)
2. Follow the gradient
   → the proposal rolls along the posterior surface
3. Where it stops = the new proposal

Gradient computed via autograd through the NN

How long does it roll?
In HMC: fixed by the user, hard to tune

NUTS[2] (No-U-Turn Sampler): stops automatically when the trajectory turns back on itself. Then picks one of the visited points as the new sample

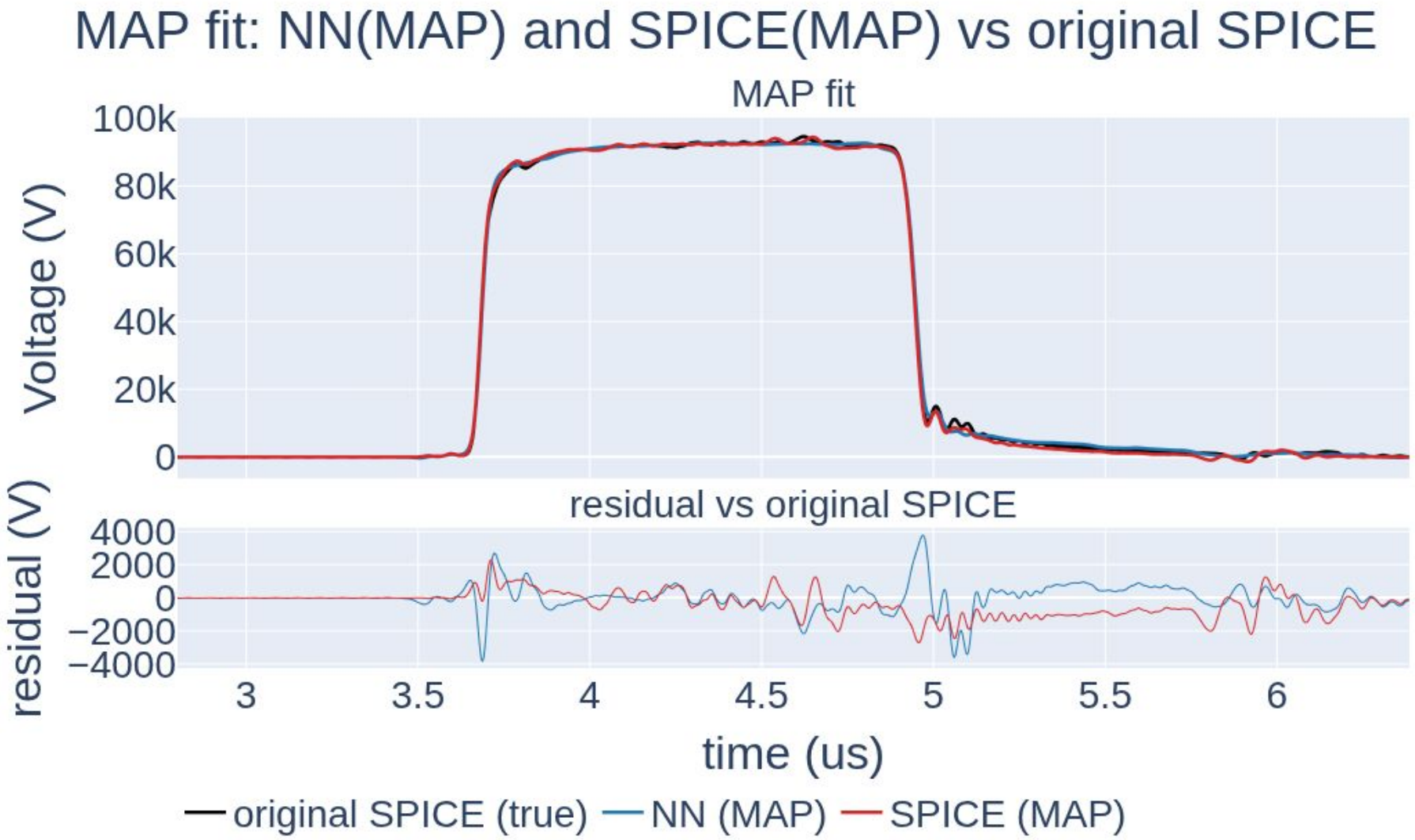
→ **No manual tuning required**

[1] Betancourt, 2018: In high dimensions, probability concentrates on a thin shell called the "typical set"; random-walk samplers miss it, while HMC uses gradient information to follow it efficiently.

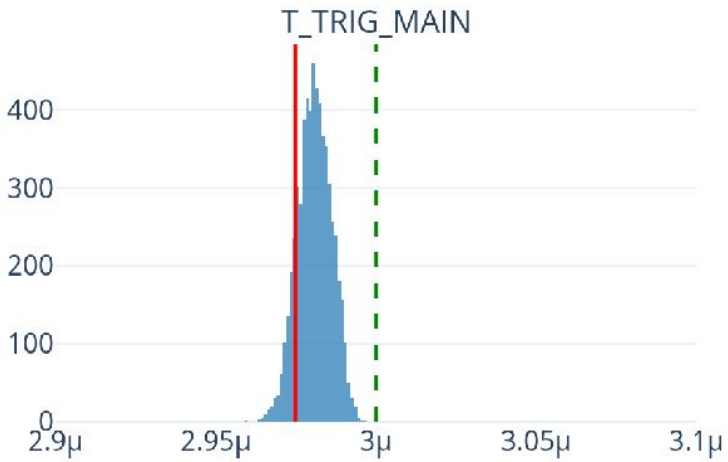
[2] Hoffman & Gelman, JMLR 2014: NUTS removes HMC's main tuning burden: it automatically sets the number of leapfrog steps by building a trajectory that stops when it starts doubling back (U-turn criterion). Also adapts the step size during warmup.

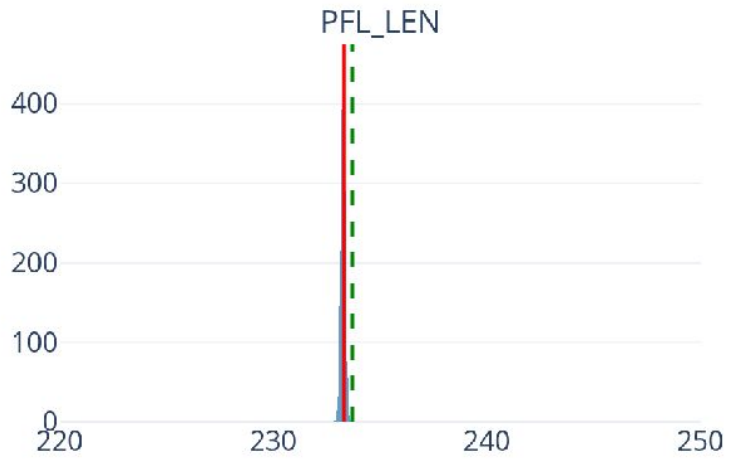
STEP 4: RESULTS

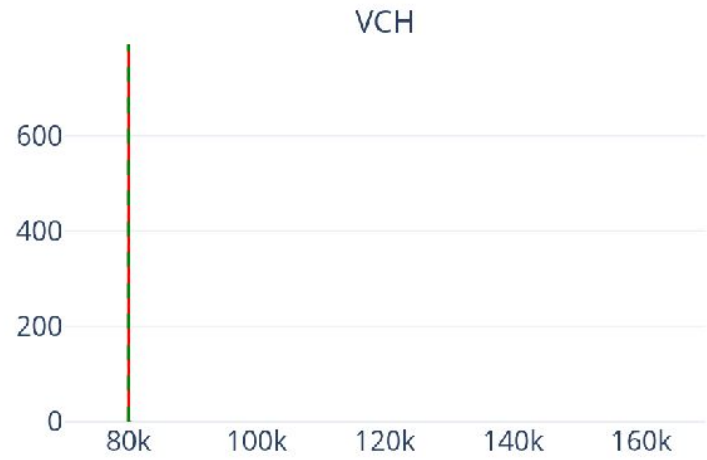
NUTS sampling: 4 chains, 2000 draws each



STEP 4: RESULTS – POSTERIOR HISTOGRAM

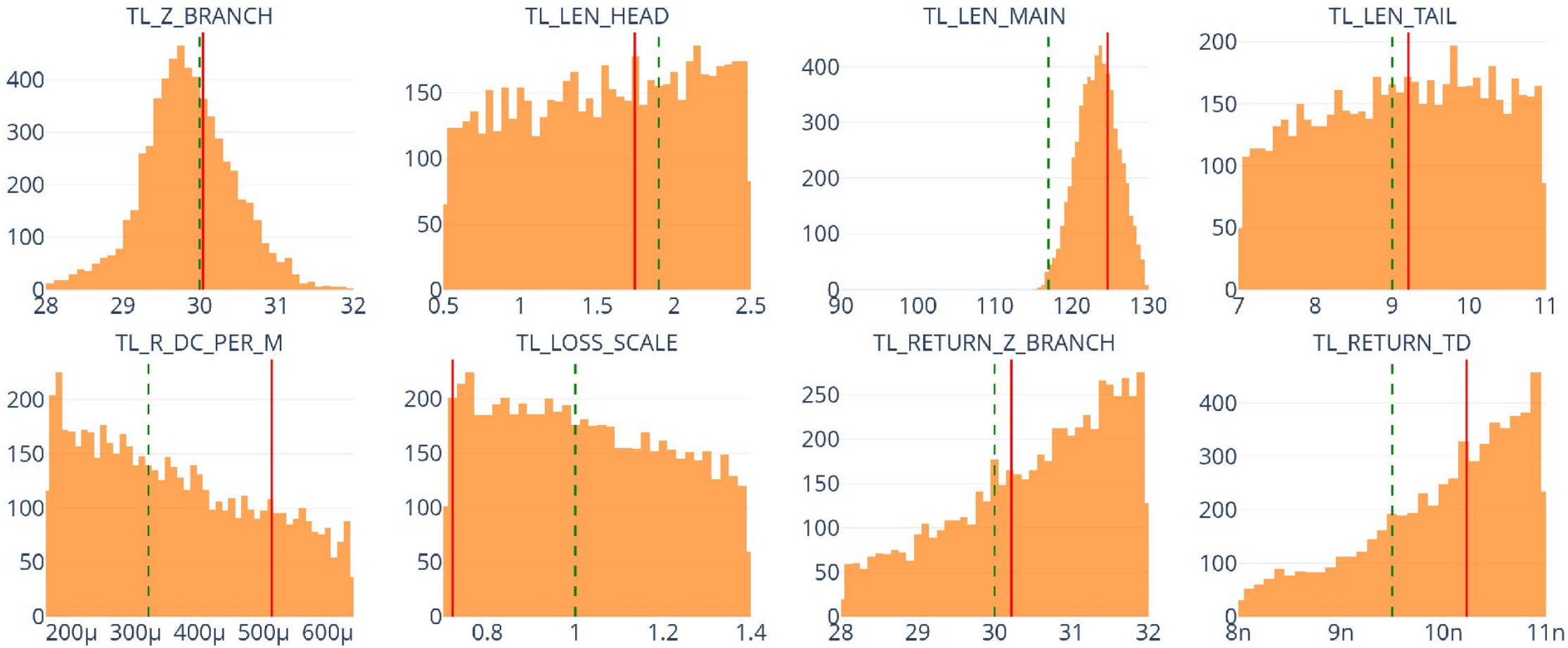






STEP 4: RESULTS – POSTERIOR HISTOGRAM

STEP 4: RESULTS – POSTERIOR HISTOGRAM



STEP 4: RESULTS – NOMINAL VS MAP

Parameter	Nominal	MAP	95% CI low	95% CI high	Error (%)
R_IMG_FILT_LEFT	5	6.782	2.734	9.852	+35.6%
C_IMG_FILT_LEFT	3.300e-10	1.992e-10	1.029e-10	3.861e-10	-39.6%
C_IMG_FILT_CENTER	6.800e-10	4.251e-10	2.244e-10	5.702e-10	-37.5%
R_IMG_FILT_RIGHT	4	5.294	2.297	7.91	+32.3%
C_IMG_FILT_RIGHT	4.500e-10	7.181e-10	3.171e-10	8.307e-10	+59.6%
R_FILT_MS1_VAL	15	11.43	7.644	27.99	-23.8%
C_FILT_MS1_VAL	3.300e-10	1.486e-10	1.273e-10	6.136e-10	-55.0%
L_FILT_MS1	4.000e-08	4.325e-08	2.119e-08	7.839e-08	+8.1%
R_FILT_MS2_VAL	30	29.75	16.56	58.92	-0.8%
C_FILT_MS2_VAL	7.200e-10	5.463e-10	2.788e-10	7.059e-10	-24.1%

STEP 4: RESULTS – NOMINAL VS MAP

Parameter	Nominal	MAP	95% CI low	95% CI high	Error (%)
L_FILT_MS2	4.000e-08	3.506e-08	2.090e-08	7.721e-08	-12.3%
PFL_LEN	233.7	233.3	233.1	233.5	-0.2%
PFL_LOSS_DC	1	1.602	0.643	1.985	+60.2%
PFL_LOSS_AC	1	1.017	0.9055	1.272	+1.7%
TL_Z_BRANCH	30	30.05	28.6	31.09	+0.2%
TL_LEN_HEAD	1.9	1.745	0.5591	2.456	-8.2%
TL_LEN_MAIN	117	124.7	118	128.5	+6.6%
TL_LEN_TAIL	9	9.213	7.138	10.91	+2.4%
TL_R_DC_PER_M	3.200e-04	5.126e-04	1.670e-04	6.207e-04	+60.2%
TL_LOSS_SCALE	1	0.7206	0.7155	1.375	-27.9%
TL_RETURN_Z_BRANC	30	30.22	28.28	31.94	+0.7%

WORKFLOW
STEP 5

Root cause analysis with SPICE + surrogate

STEP 5: Root cause analysis

Next step:

- Apply MCMC to OOD latent vector z
- Compare inferred parameters in time
- Parameter deviations show the most likely failing component(s).

Thank you !