Update on ObsBox and Loss maps studies

Speaker: Loïc COYLE

Acknowledgements:
T. Pieloni, M. Schenk, X. Buffat, M.Schenk, R. Bruce, R. De Maria, A. Mereghetti,
D. Mirarchi, T. Persson

ObsBox

Quick recap

The ObsBox:

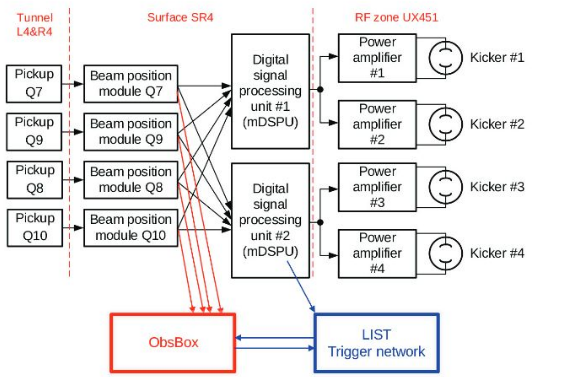


Figure 1: An overview of the LHC transverse feedback system (ADT).

→ rolling buffer & saves on trigger
→ 65536 turns
→ bunch by bunch
→ transverse position data

The problem:

- The trigger is not very accurate :

Most of the data does not contain any instabilities.
→ instabilities make up less than 1% !

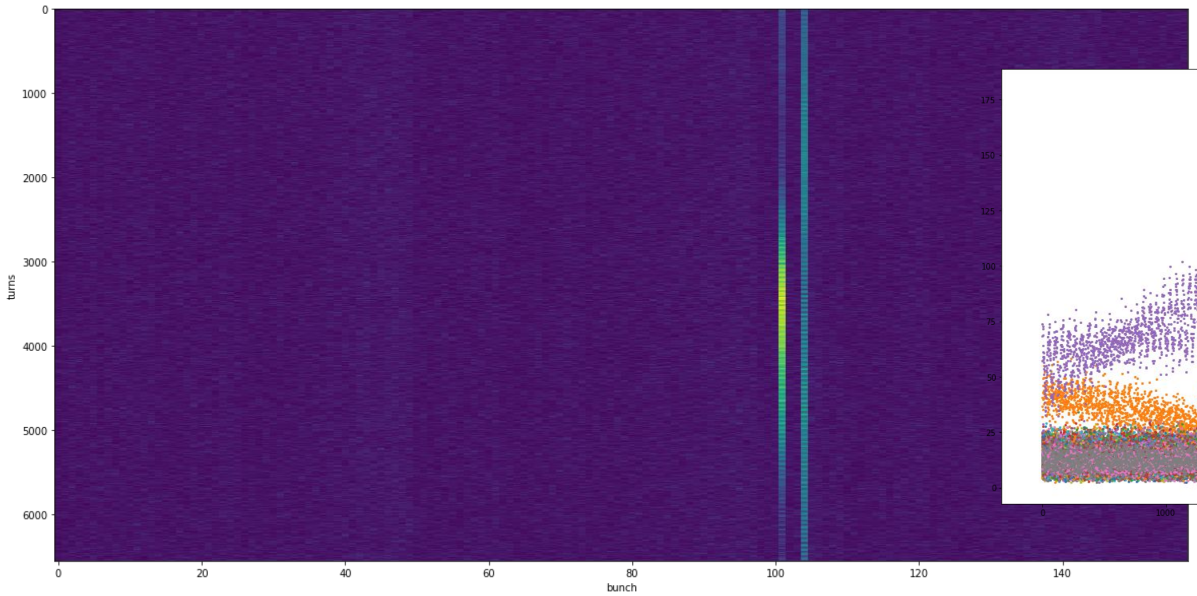
- Large amount of data ~4 TB
- Very little labeled data collected manually

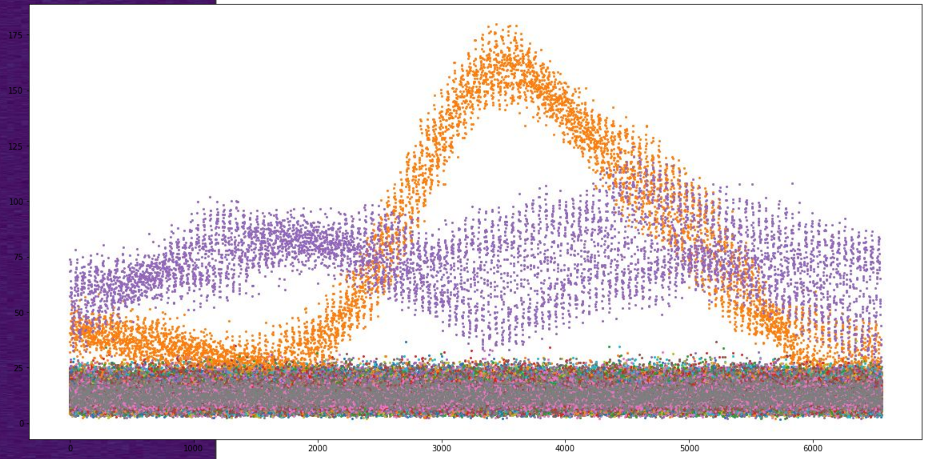
ADTObsBox

Quick recap

Raw beam amplitude data at a turn by turn and bunch by bunch resolution. → multivariate time series

Example: 07169_Inst_B1V_Q7_20180914_08h53m08s → 2 unstable bunches

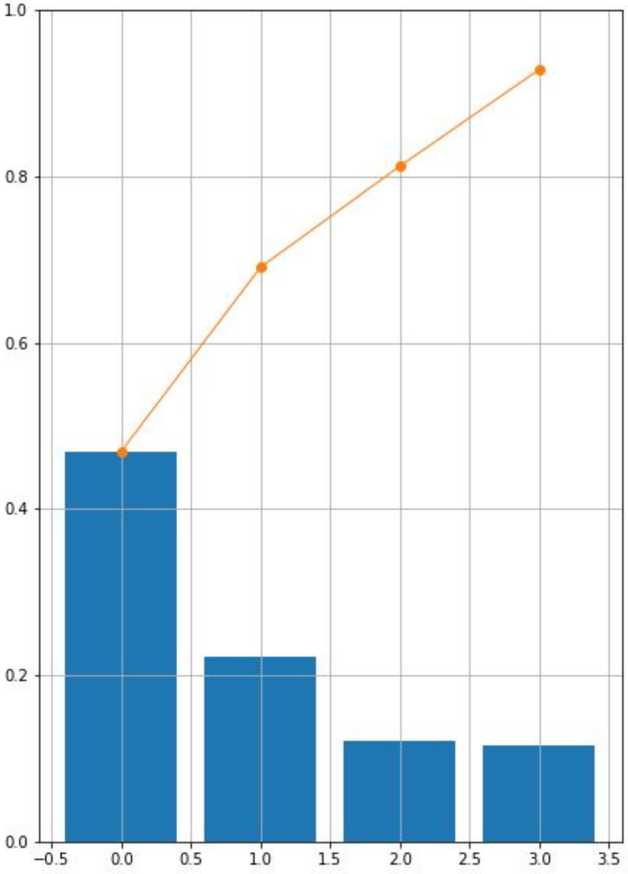


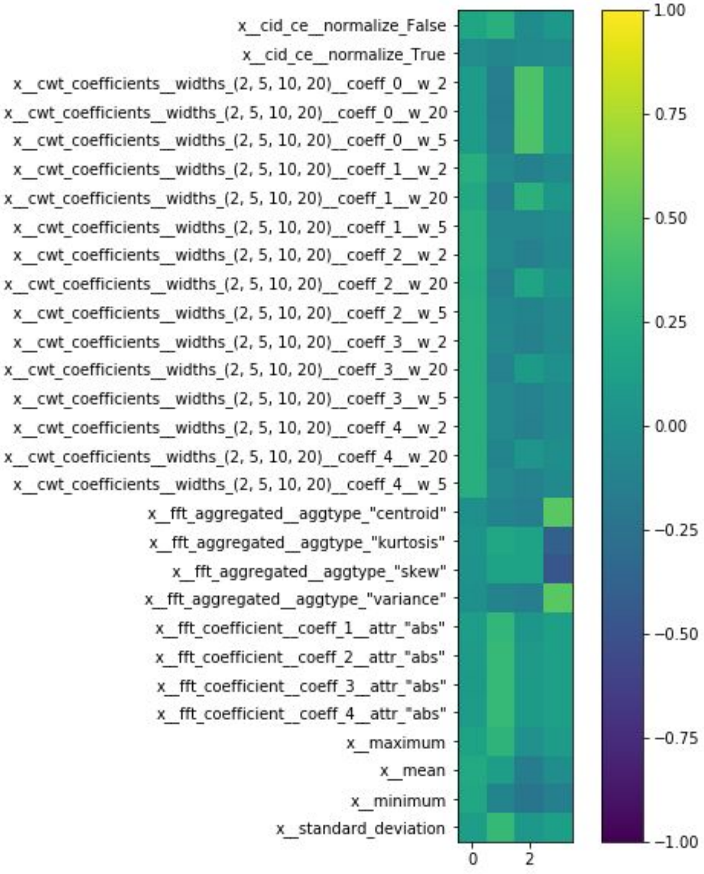


Principal Component Analysis

Quick recap

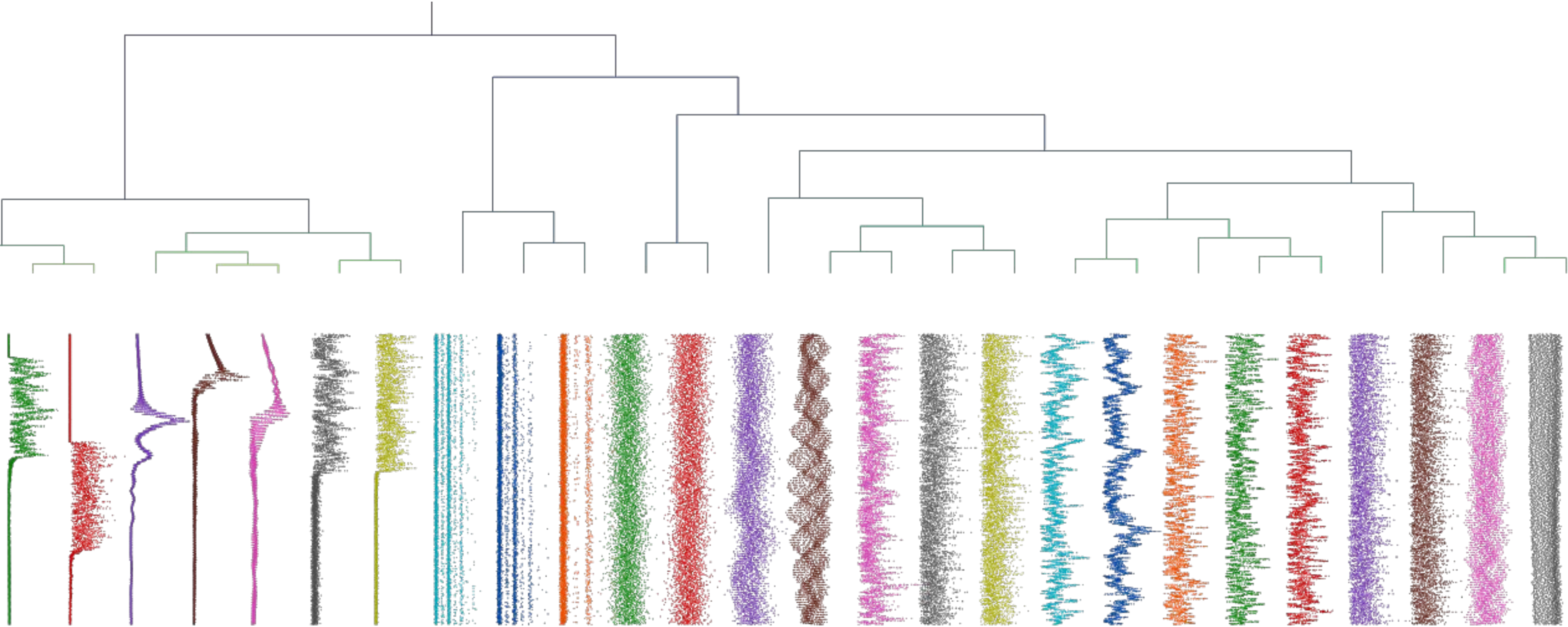
PCA vectors truncated to 4 components → ~93% variance explained





Linkage Tree

Quick recap



Full plot: https://cernbox.cern.ch/index.php/s/F6m2LQlVVBvCK79 or https://imgur.com/a/jeDk8ts

Conclusion

OBsBox:

- Anomaly detection for instability detection ~working
  - Refine the extracted features
  - Isolation forest hyper parameters
- Some preliminary (univariate) time series clustering ~working
- Proof of concept seems to produce coherent results
- Improvement:
  - More features → extend to run on cluster ~nearly working
  - Look into multivariate (multi-bunch) time series clustering
- Look into online use

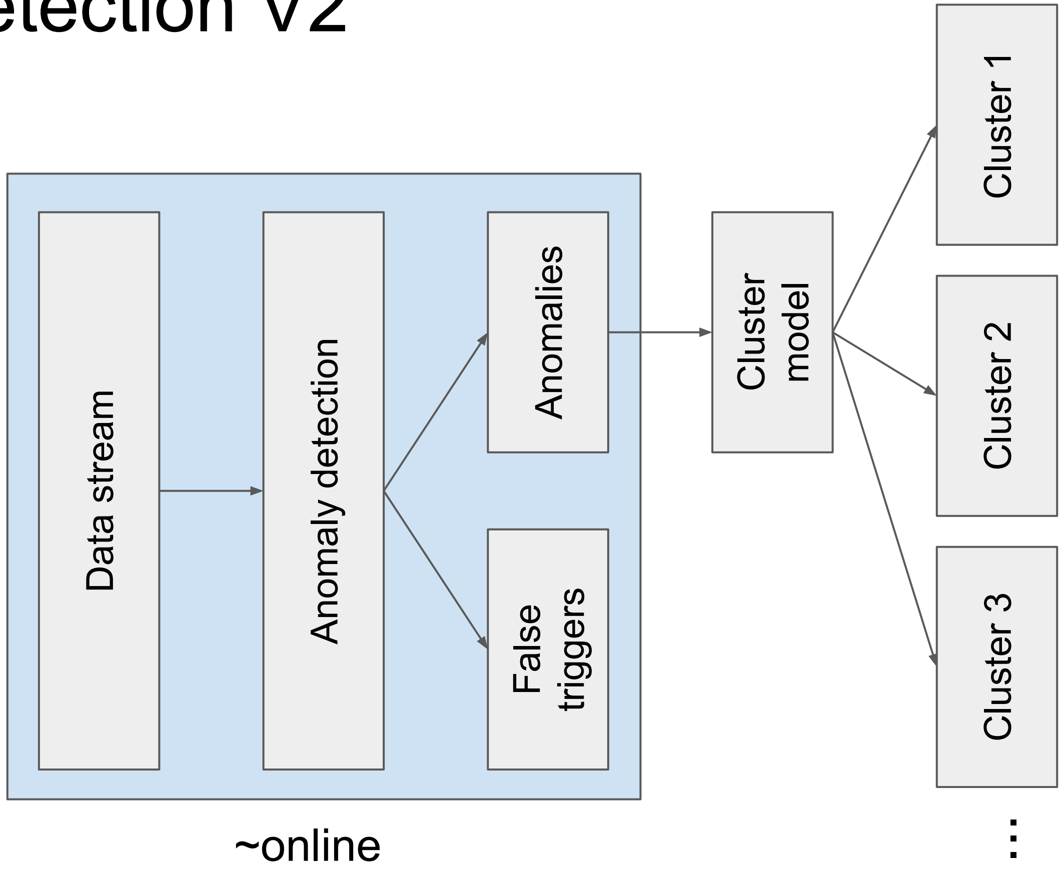
Conclusion

Quick recap

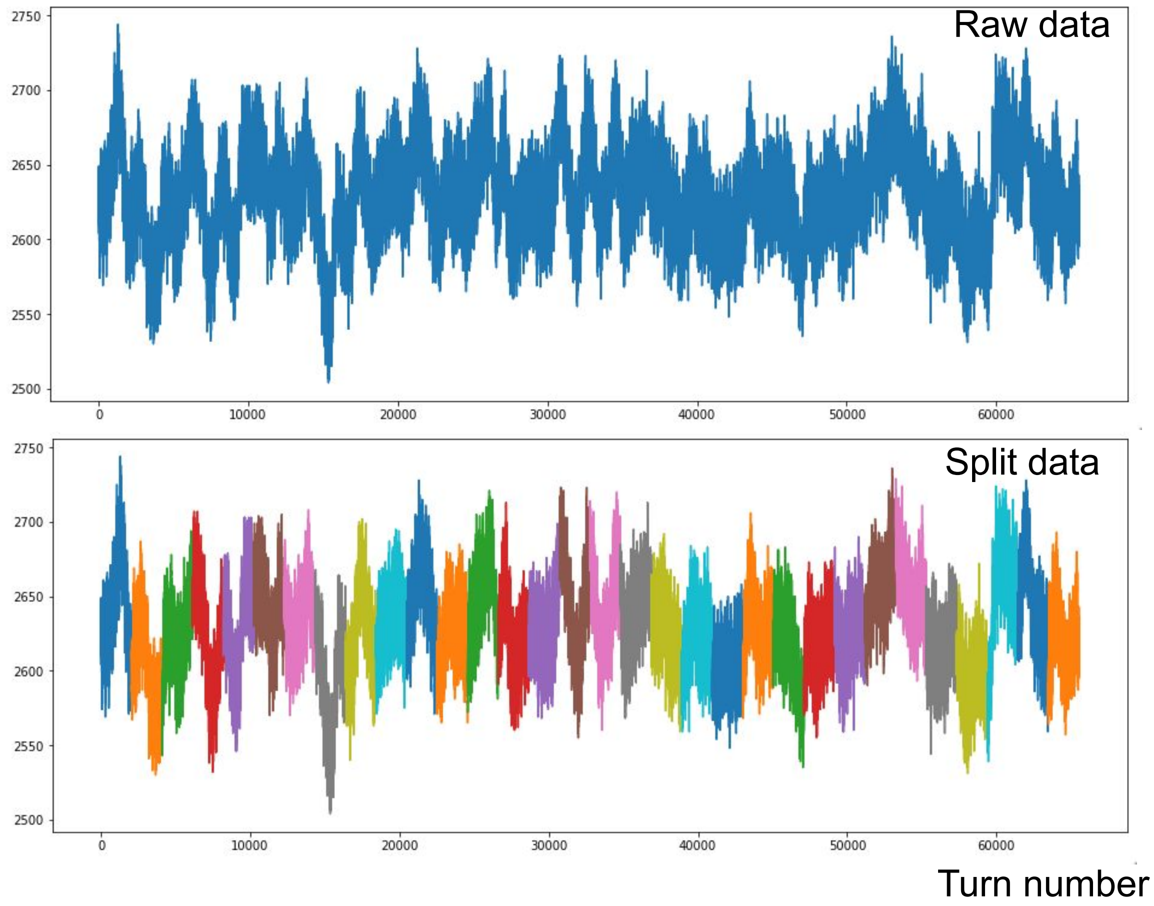
OBsBox:

- Anomaly detection for instability detection ~working
  - Refine the extracted features
  - Isolation forest hyper parameters
- Some preliminary (univariate) time series clustering ~working
- Proof of concept seems to produce coherent results
- Improvement:
  - More features → extend to run on cluster ~nearly working
  - Look into multivariate (multi-bunch) time series clustering
- **Look into online use**

Anomaly detection V2

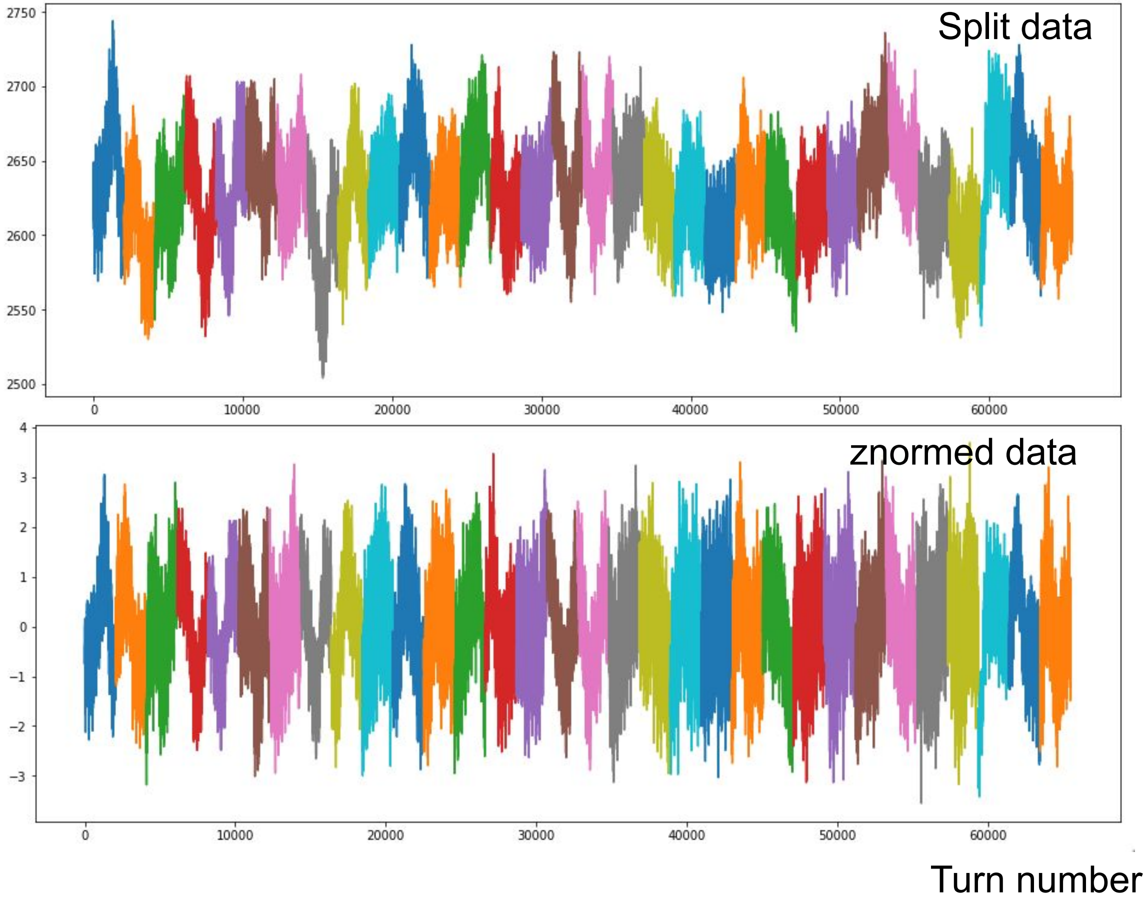


Anomaly detection V2 - Preprocessing



The 65536 turn buffer is split into 2048 smaller chunks.

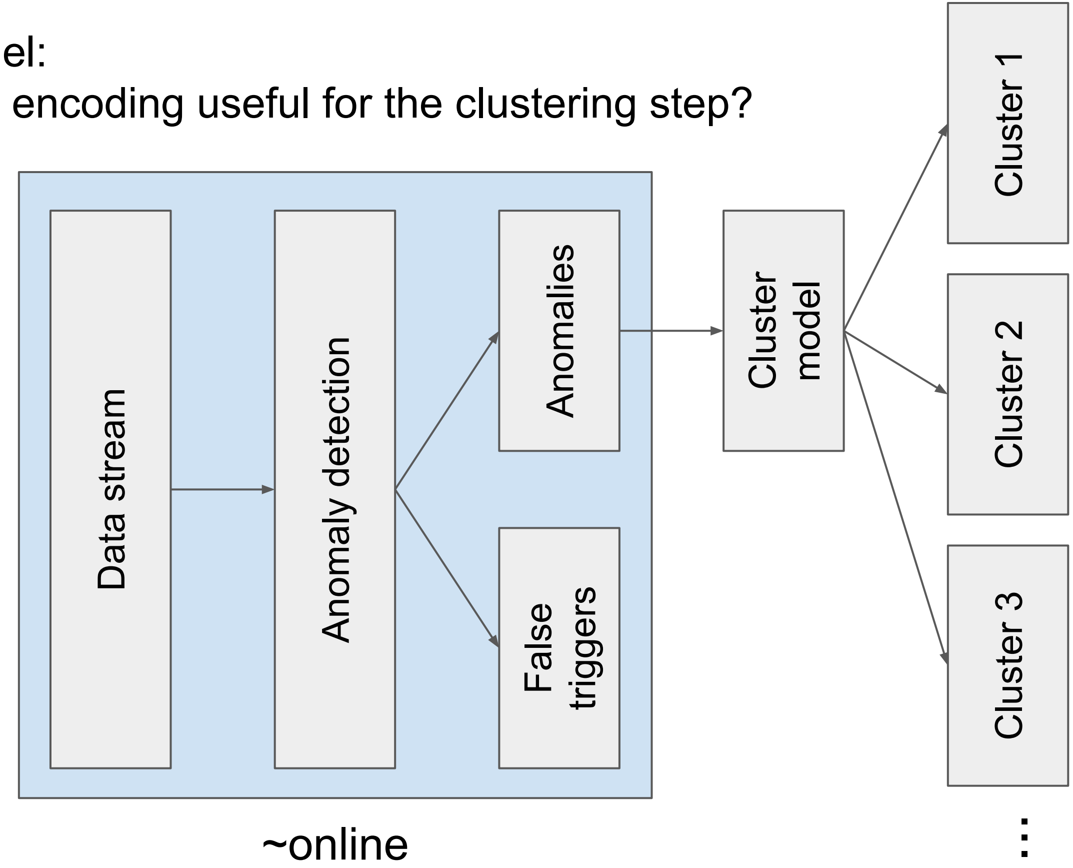
Anomaly detection V2 - Normalization



Each chunk is normalized

Anomaly detection V2 - Model

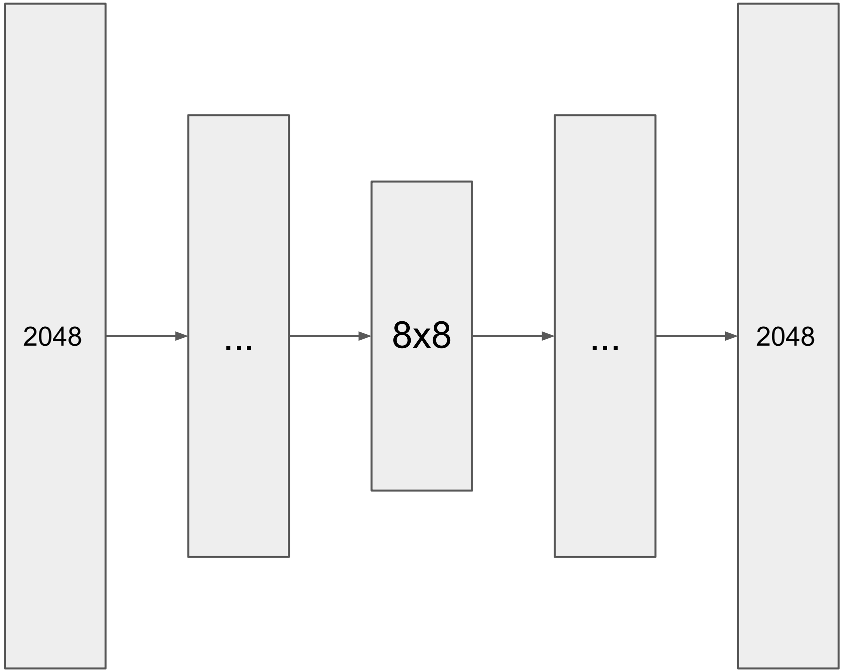
Autoencoder based model:
→ Learns a latent space encoding useful for the clustering step?



Anomaly detection V2 - Model

Autoencoder based model:
→ Convolutional autoencoder → image like representation in latent space

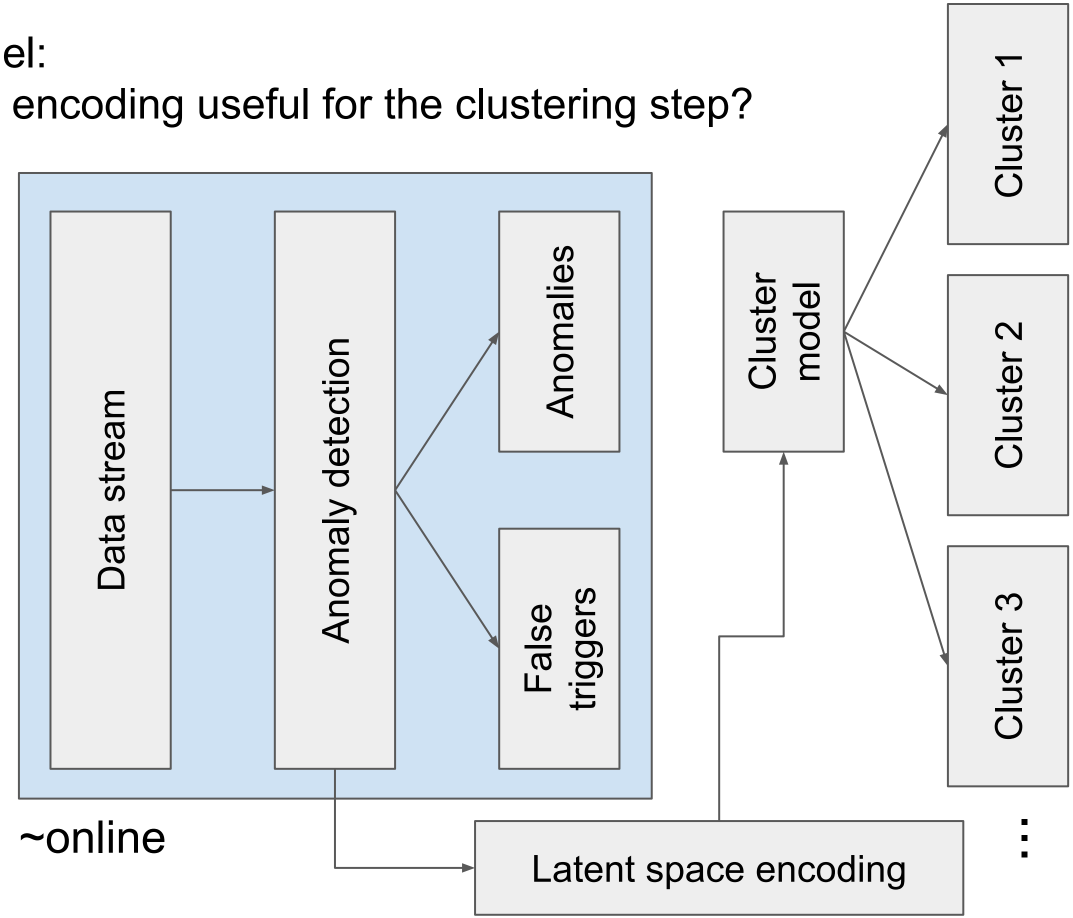
Reconstruct the input despite a bottle neck.




```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d (Conv1D)              (None, 512, 64)           512
_________________________________________________________________
dropout (Dropout)            (None, 512, 64)           0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 128, 32)           14368
_________________________________________________________________
dropout_1 (Dropout)          (None, 128, 32)           0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 32, 16)            3600
_________________________________________________________________
dropout_2 (Dropout)          (None, 32, 16)            0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 8, 8)              904
_________________________________________________________________
conv1d_transpose (Conv1DTran (None, 32, 8)             456
_________________________________________________________________
dropout_3 (Dropout)          (None, 32, 8)             0
_________________________________________________________________
conv1d_transpose_1 (Conv1DTr (None, 128, 16)           912
_________________________________________________________________
dropout_4 (Dropout)          (None, 128, 16)           0
_________________________________________________________________
conv1d_transpose_2 (Conv1DTr (None, 512, 32)           3616
_________________________________________________________________
dropout_5 (Dropout)          (None, 512, 32)           0
_________________________________________________________________
conv1d_transpose_3 (Conv1DTr (None, 2048, 64)          14400
_________________________________________________________________
conv1d_transpose_4 (Conv1DTr (None, 2048, 1)           449
=================================================================
Total params: 39,217
Trainable params: 39,217
Non-trainable params: 0
_________________________________________________________________
```

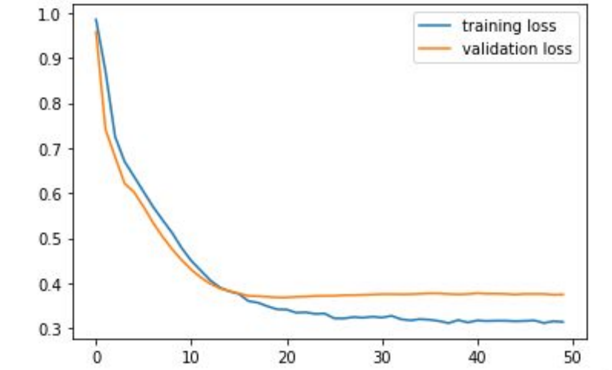
Anomaly detection V2 - Model

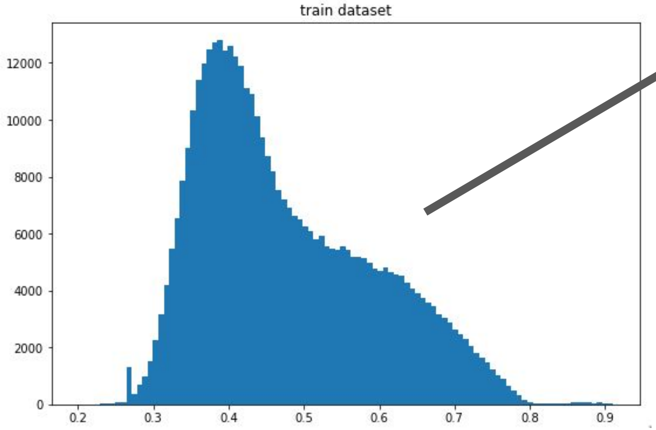
Autoencoder based model:
→ Learns a latent space encoding useful for the clustering step?



Anomaly detection V2 - Results

Very dependent on the normalization method: **znorm**

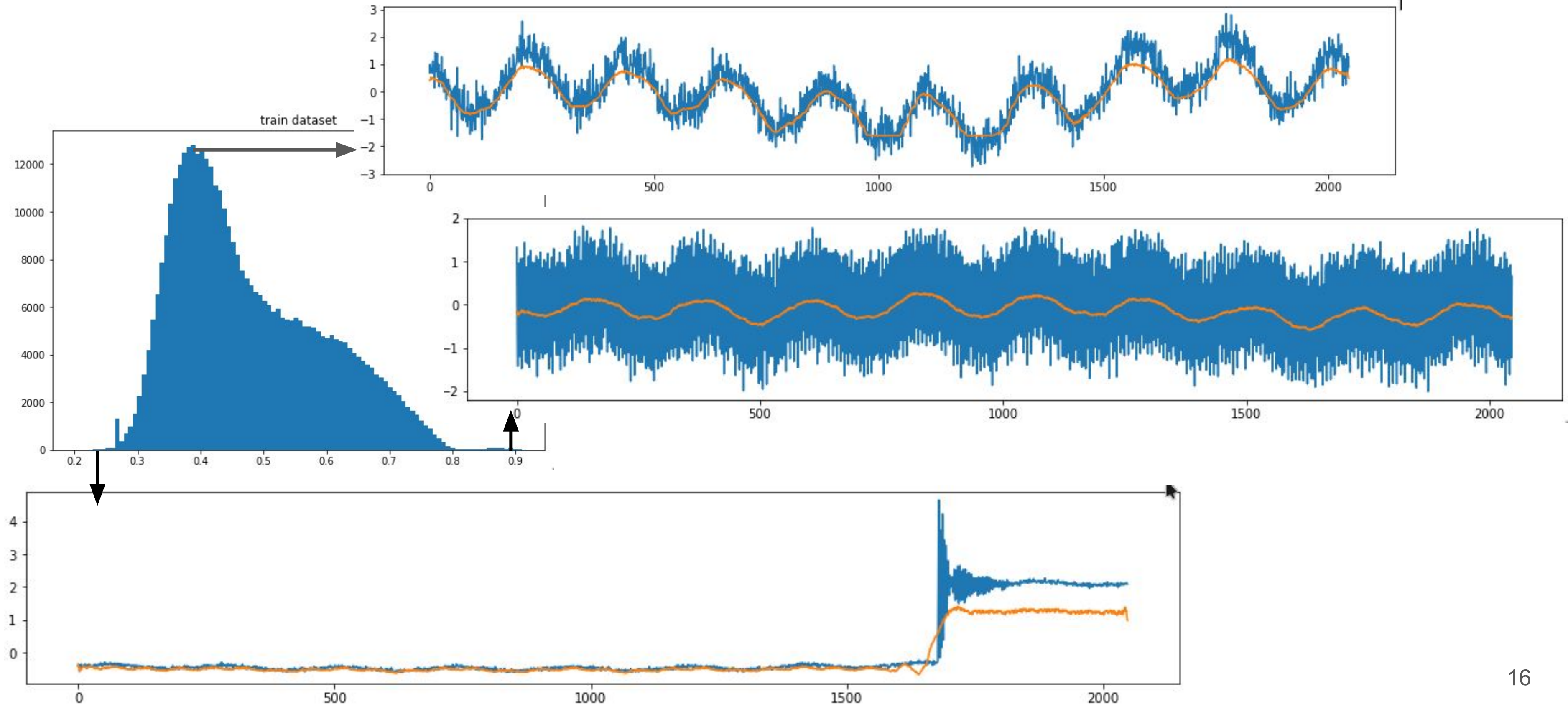




Distribution surprising, was expecting the most frequently occurring signals to have the lowest error.

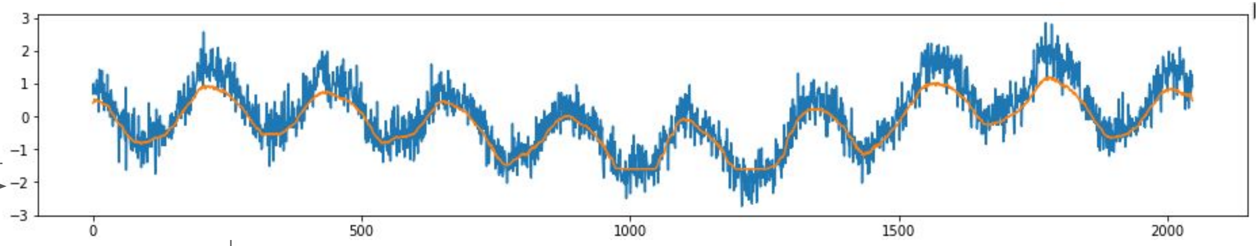
Anomaly detection V2 - Results

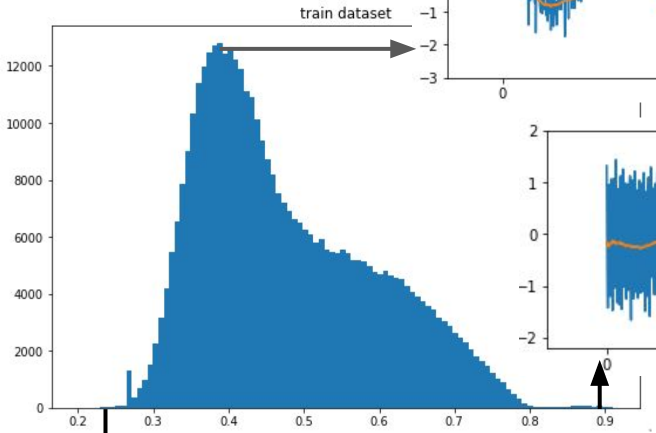
Very dependent on the normalization method: **znorm**

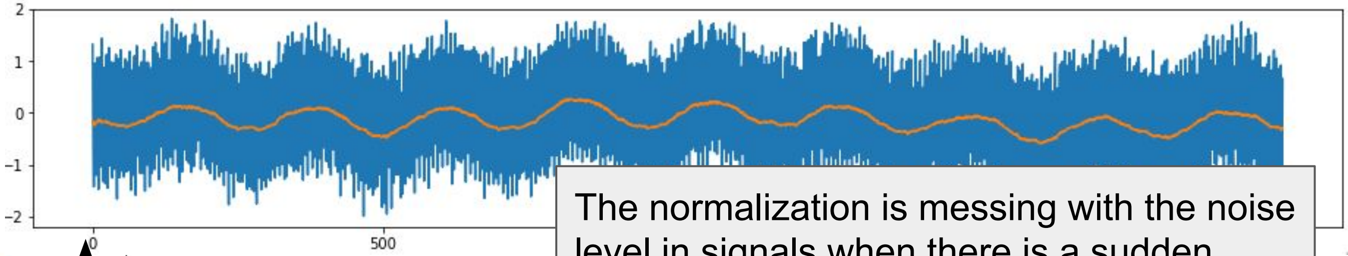


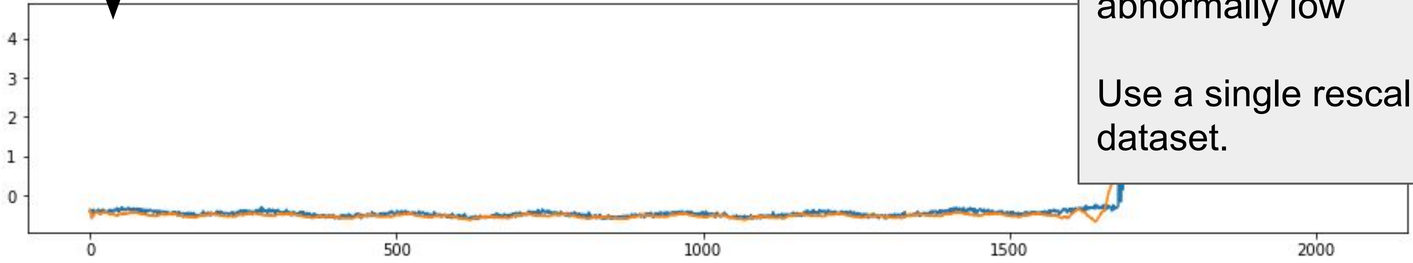
Anomaly detection V2 - Results

Very dependent on the normalization method: **znorm**







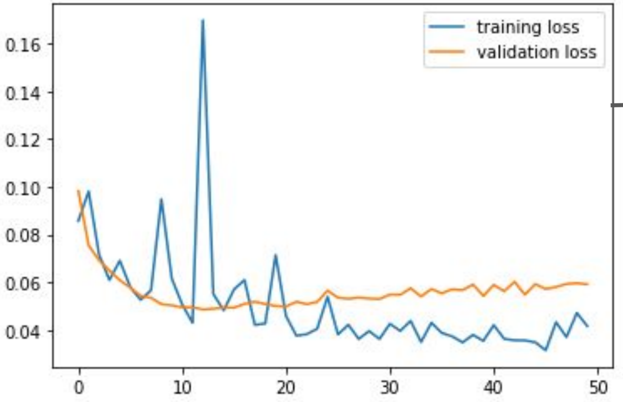


The normalization is messing with the noise level in signals when there is a sudden offset → the reconstruction error is abnormally low

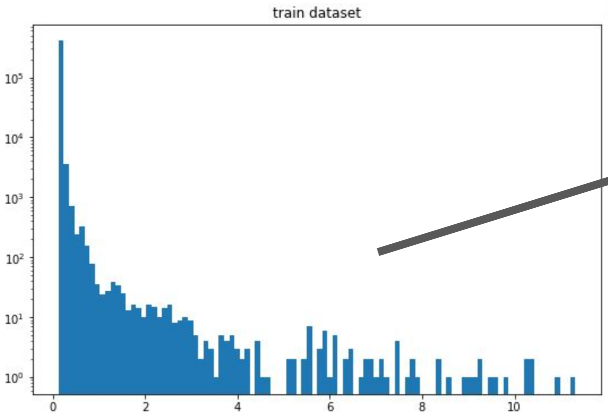
Use a single rescaling value for the entire dataset.

Anomaly detection V2 - Results

Alternative normalization: mean 0 constant normalization factor **(center scale)**



Losses is much more erratic, to be expected.

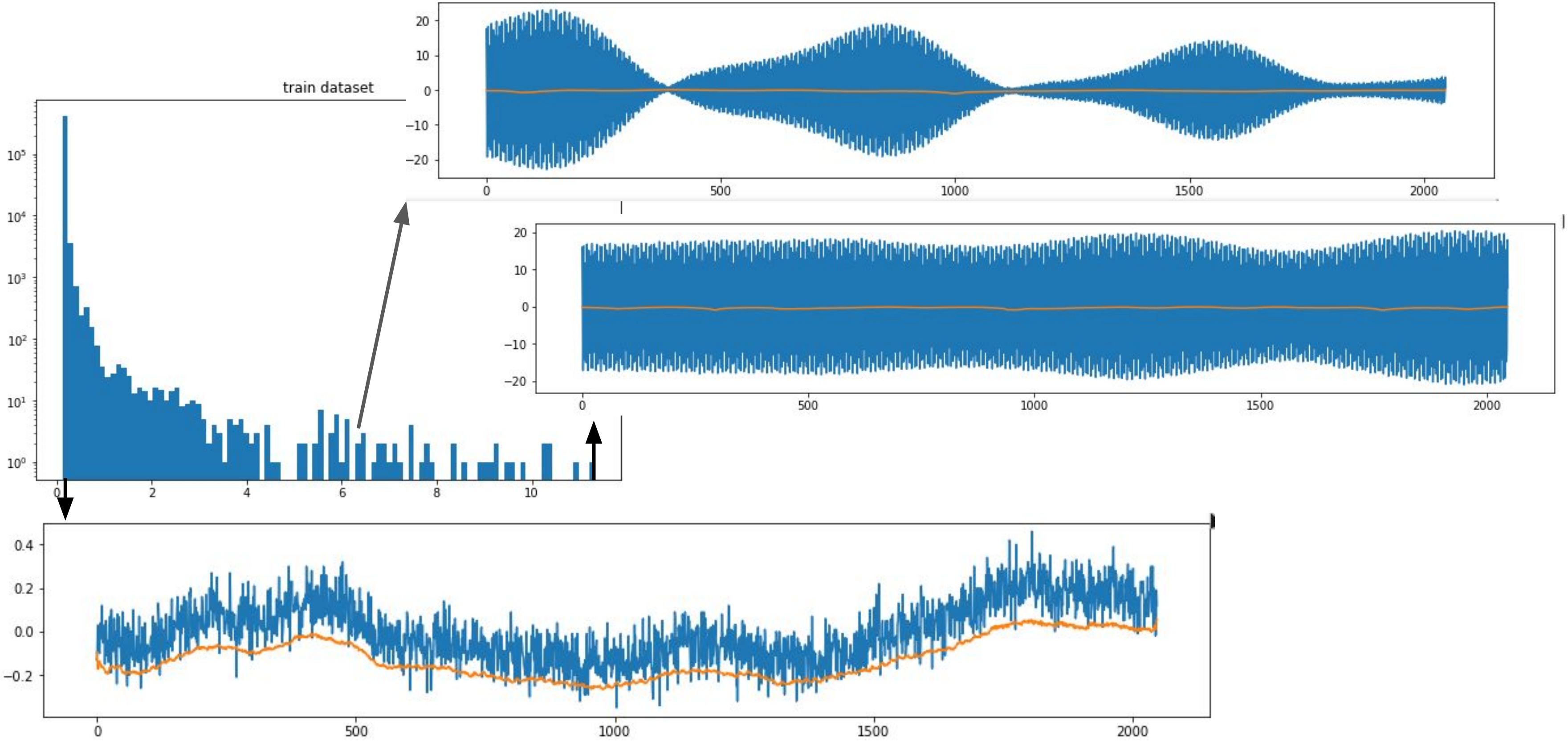


More like what I was expecting, lower errors occur more frequently.

Note the log scale.

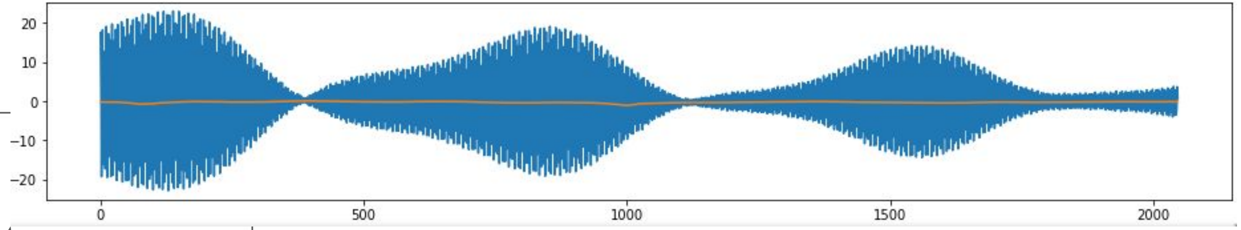
Anomaly detection V2 - Results

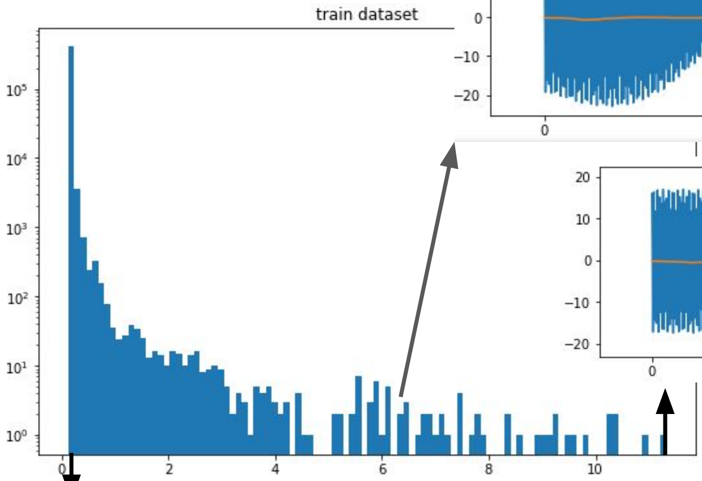
Very dependent on the normalization method: **center scale**

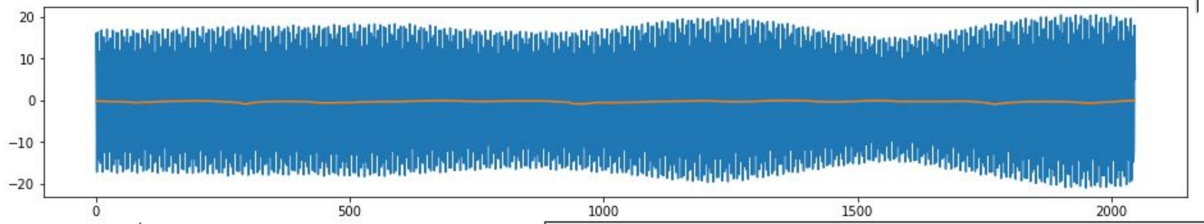


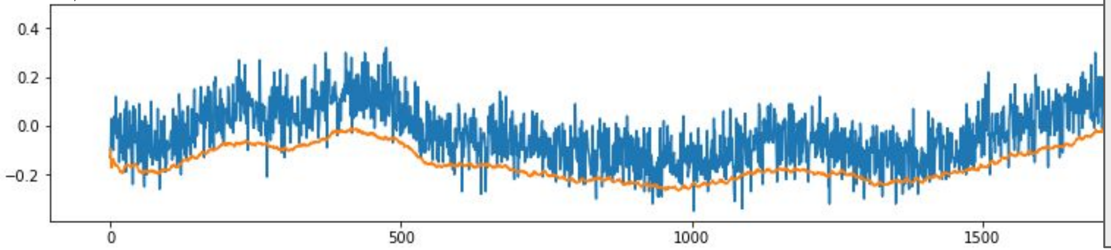
Anomaly detection V2 - Results

Very dependent on the normalization method: **center scale**









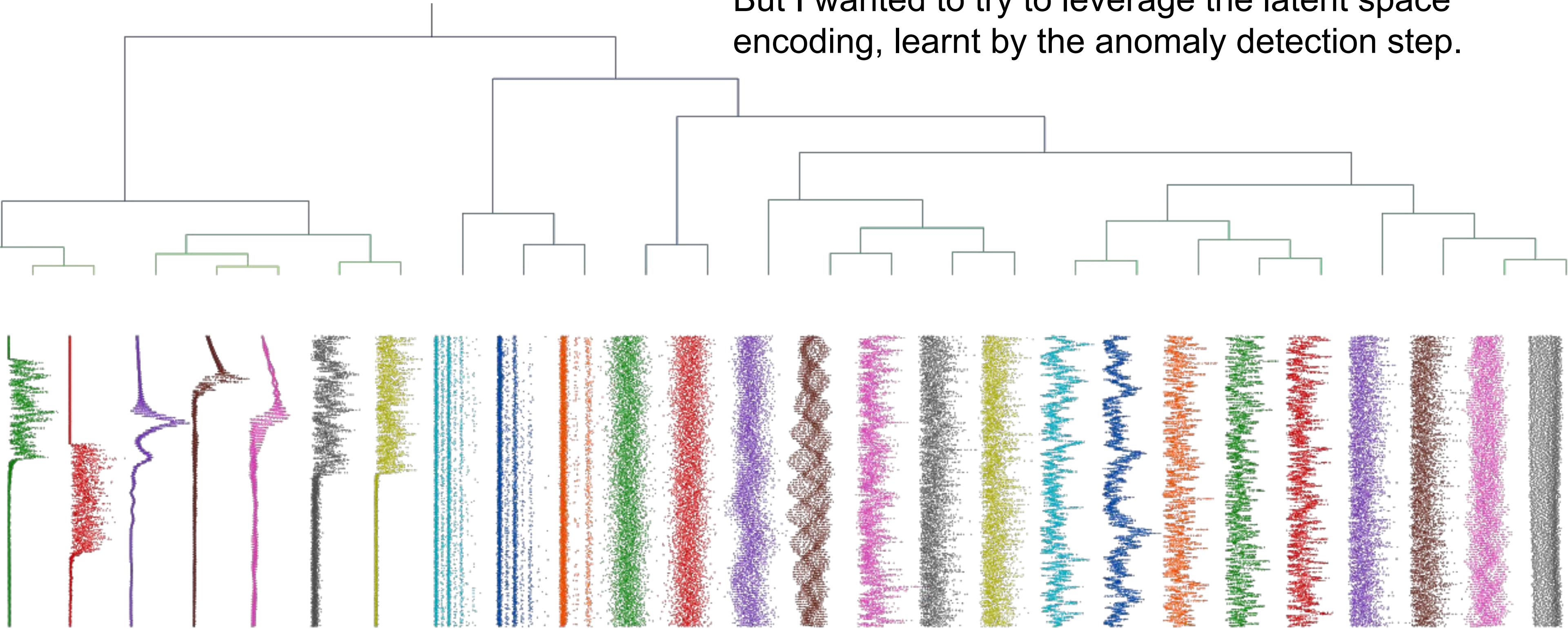
Using a different normalization method, seems to produce more coherent results. At the cost of more erratic training.

I’m also looking into using the derivative of the signal instead.

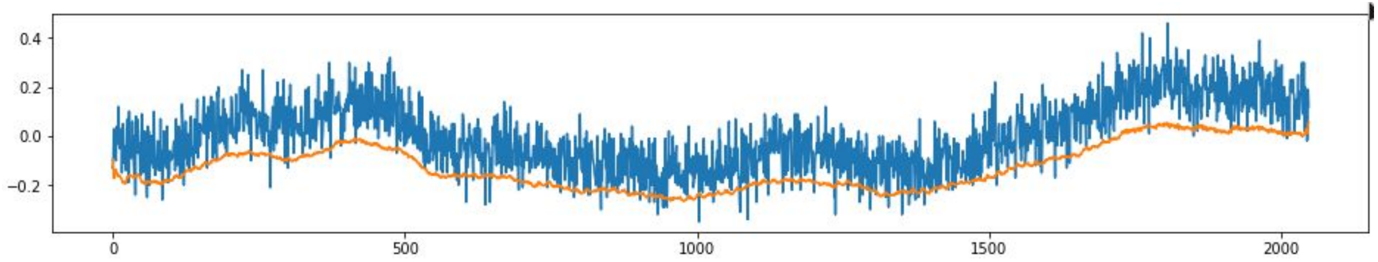
Clustering

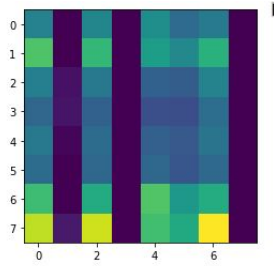
We could use the same Hierarchical clustering (using DTW) method as previously.

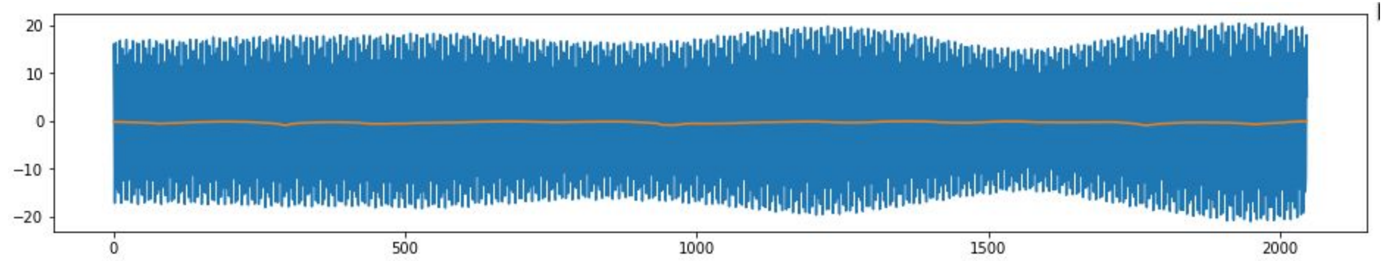
But I wanted to try to leverage the latent space encoding, learnt by the anomaly detection step.

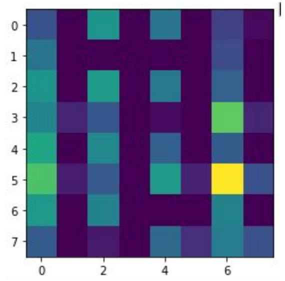


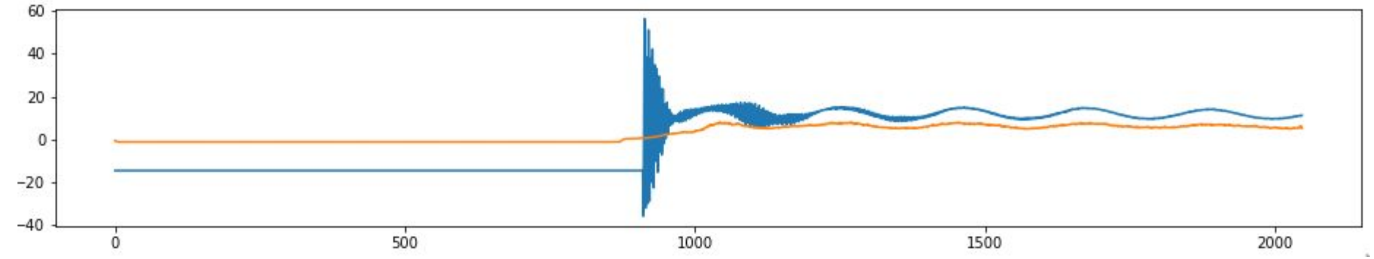
Clustering - Latent space

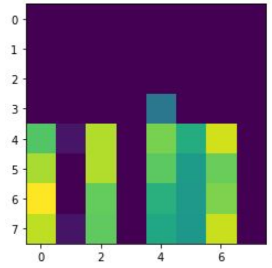




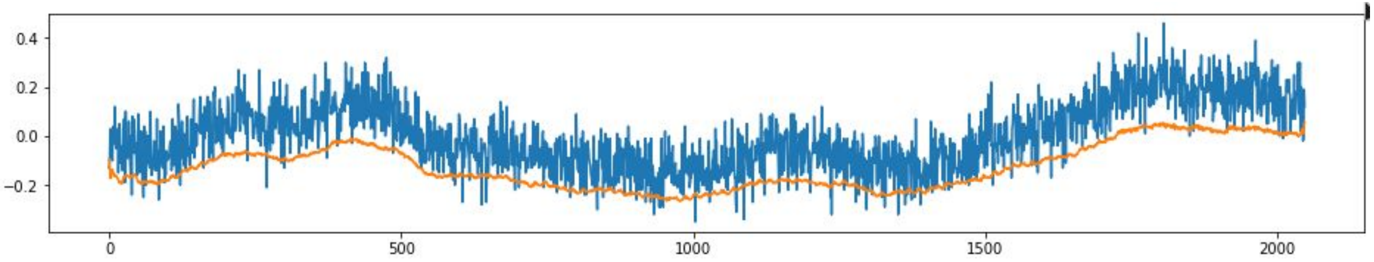


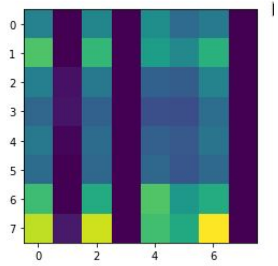


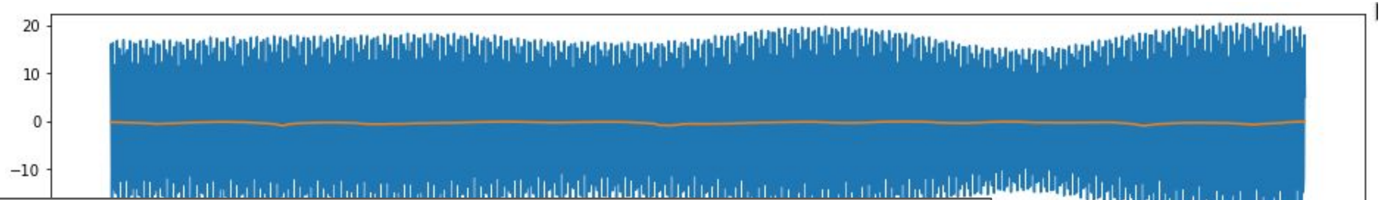


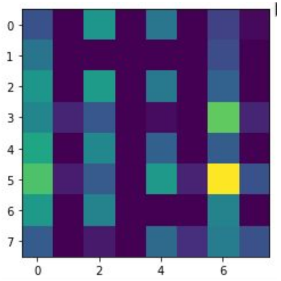


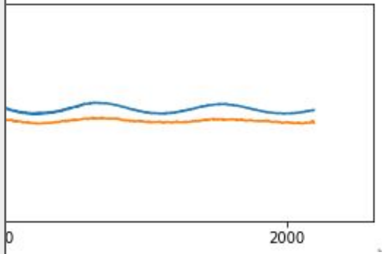
Clustering - Latent space











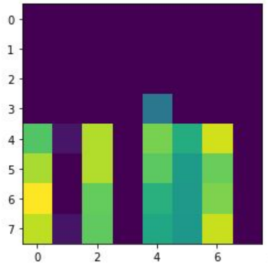


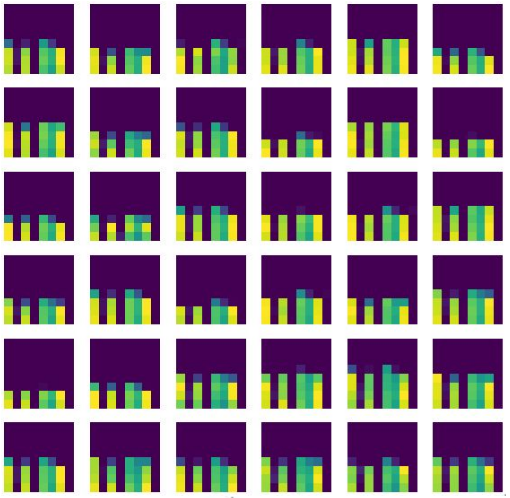
Image like representation in latent space
→ image clustering methods

Could also be very useful for multibunch instabilities, stacking of the images

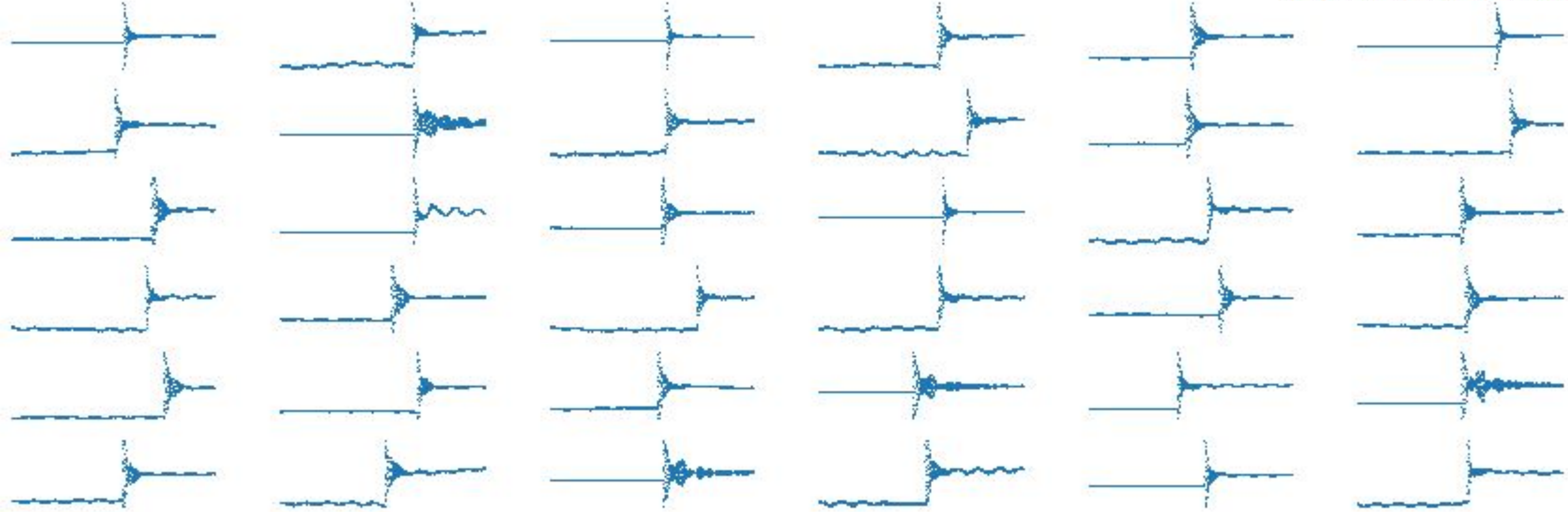
For now, standard K-Means to illustrate

Clustering - Latent space

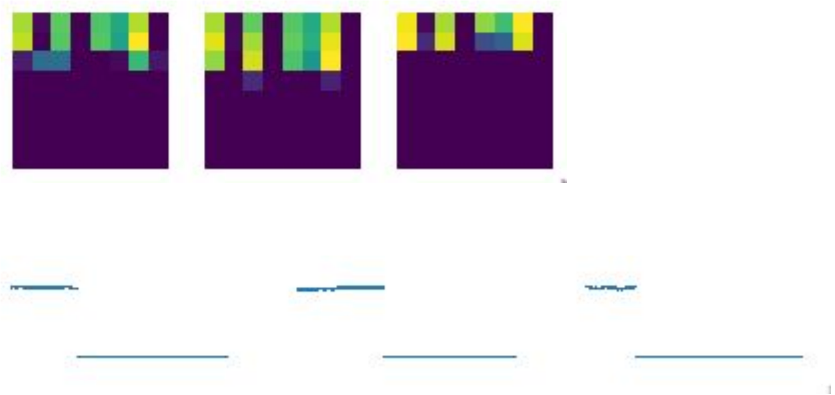
KMeans on the top anomalous signals:



Probably injections, or orbit feedback turn on.



Clustering - Latent space

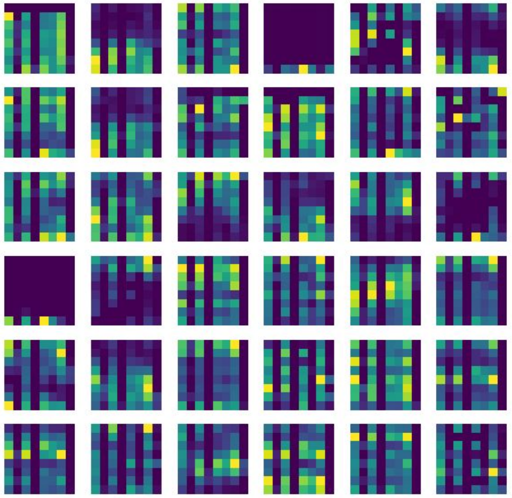


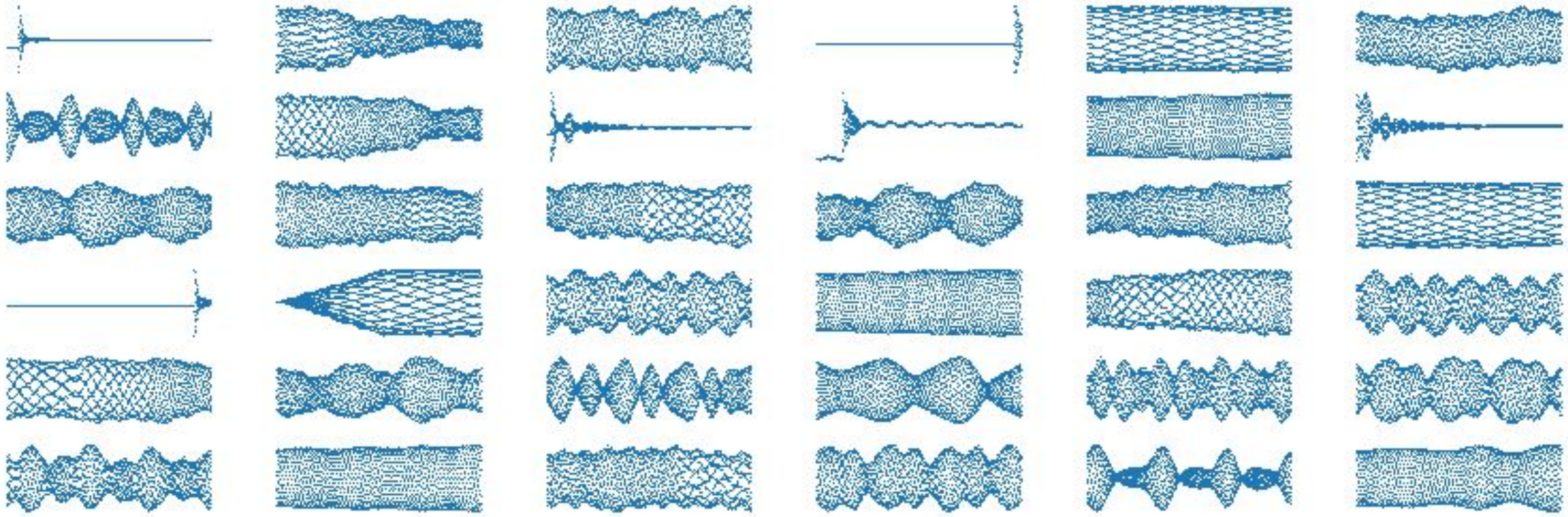
Probably beam dumps ?

Clustering - Latent space

The interesting stuff, unfortunately, the different looking signals don’t get clustered together.

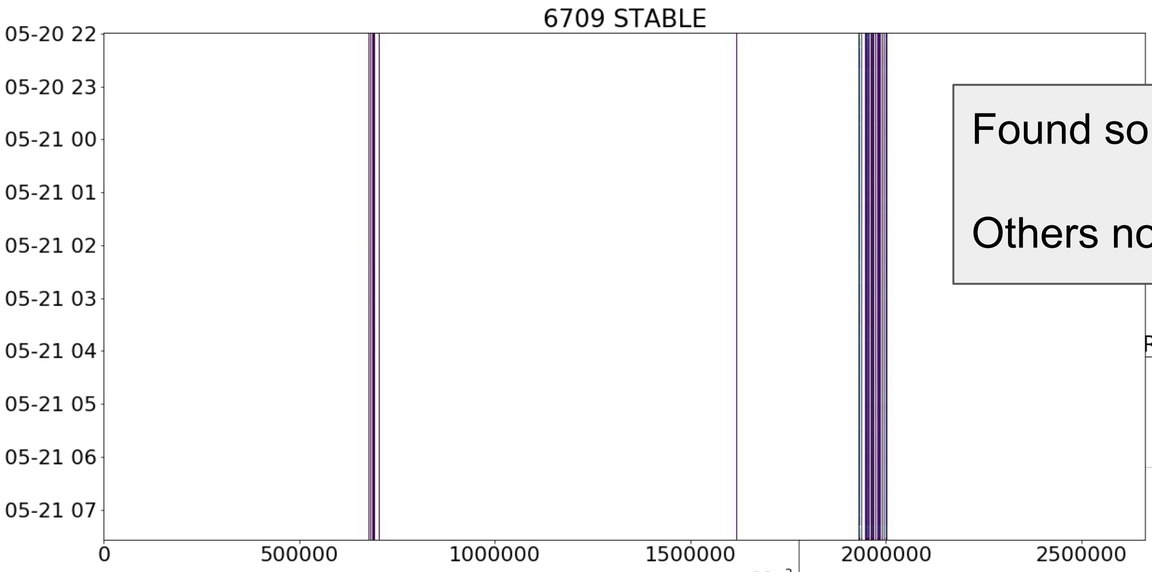
But it is at least illustrative.





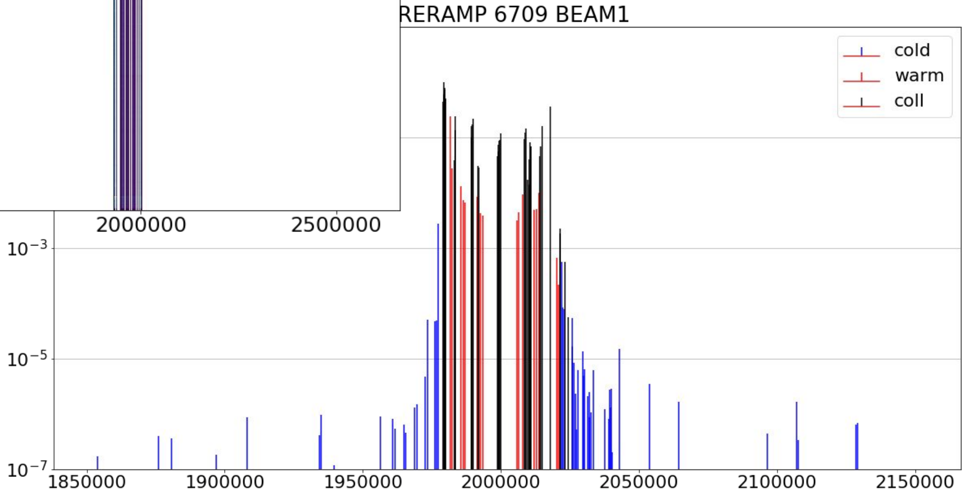
Loss maps

Loss map



Found some confirmed anomalies.

Others not very clear, hard to interpret.



Loss map

Tried many variations, no eureka moment.

Add cold region BLMs and cross check with known UFO events.

Ran into some technical problems (data storage issues), so it took a back seat to the ObsBox study.

Will pick it back up soon.

Conclusion

ObsBox:

- Online anomaly model
- Clustering with latent space encoding promising
- Figure out proper clustering model (instead of KMeans)
  - Hierarchical Clustering in latent space ?
  - Image based clustering methods ?
- Multibunch instabilities feasible