T2K 4th April 2022

# Data-Driven Characterization with Constrained Bottleneck Autoencoders

César JESÚS-VALLS cjesus@ifae.es IFAE

short paper review

# Introduction

## Data-driven detector signal characterization with constrained bottleneck autoencoders

**C.Jesús-Valls,**[a,1] **T.Lux,**[a] **F.Sánchez**[b]

[a] *Institut de Física d'Altes Energies (IFAE) - The Barcelona Institute of Science and Technology (BIST), Campus UAB, 08193 Bellaterra (Barcelona), Spain*

[b] *University of Geneva, Section de Physique, DPNC, 1205 Genève, Switzerland*

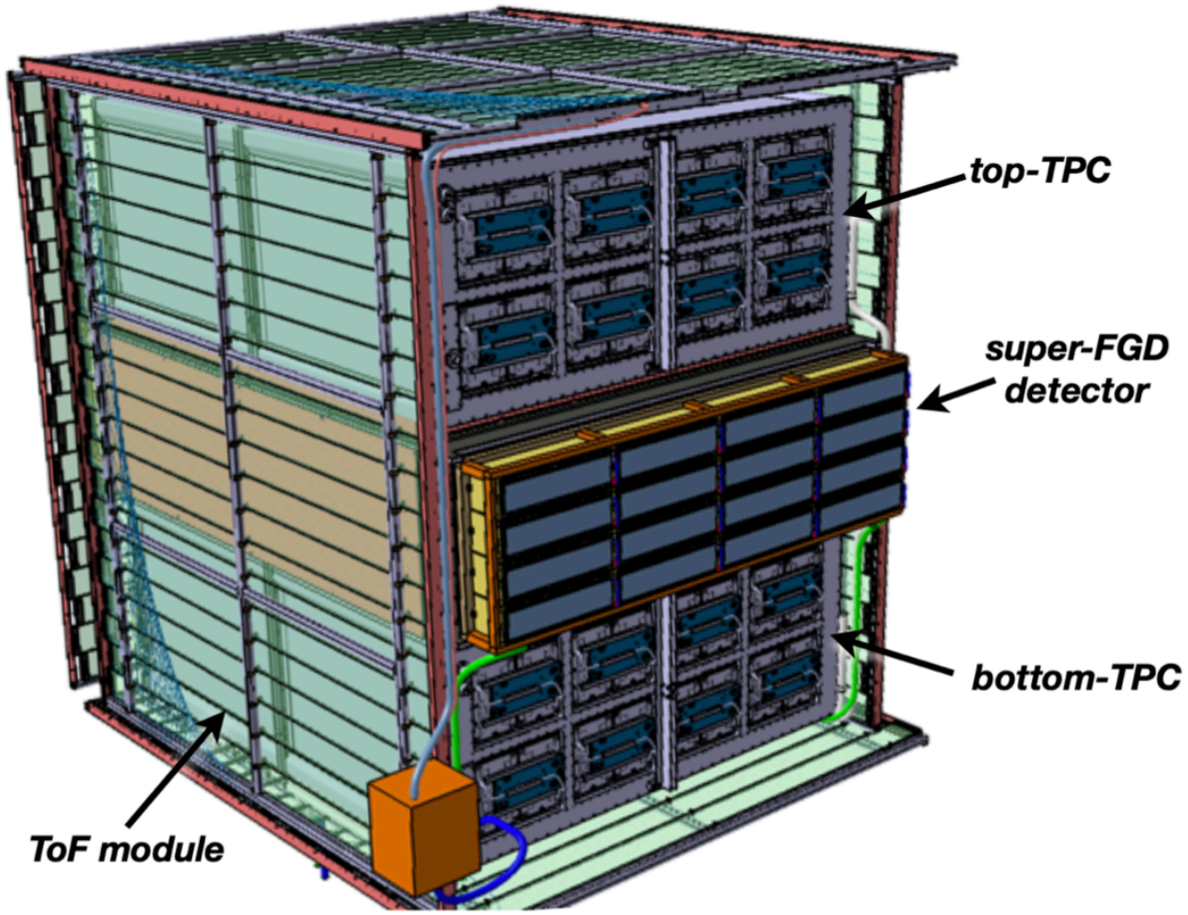
*E-mail:* cesar.jesus@cern.ch

*https://arxiv.org/pdf/2203.04604.pdf*

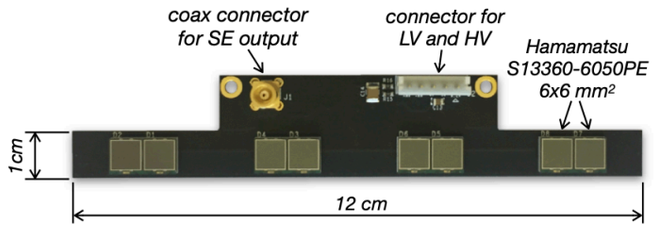
- As mentioned in previous meetings recently a paper was posted on arXiv and sent to JINST.
- This presentation does a quick review of the paper.

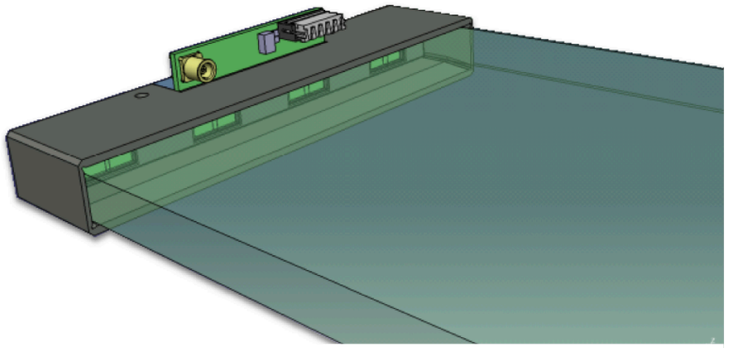
# Introduction

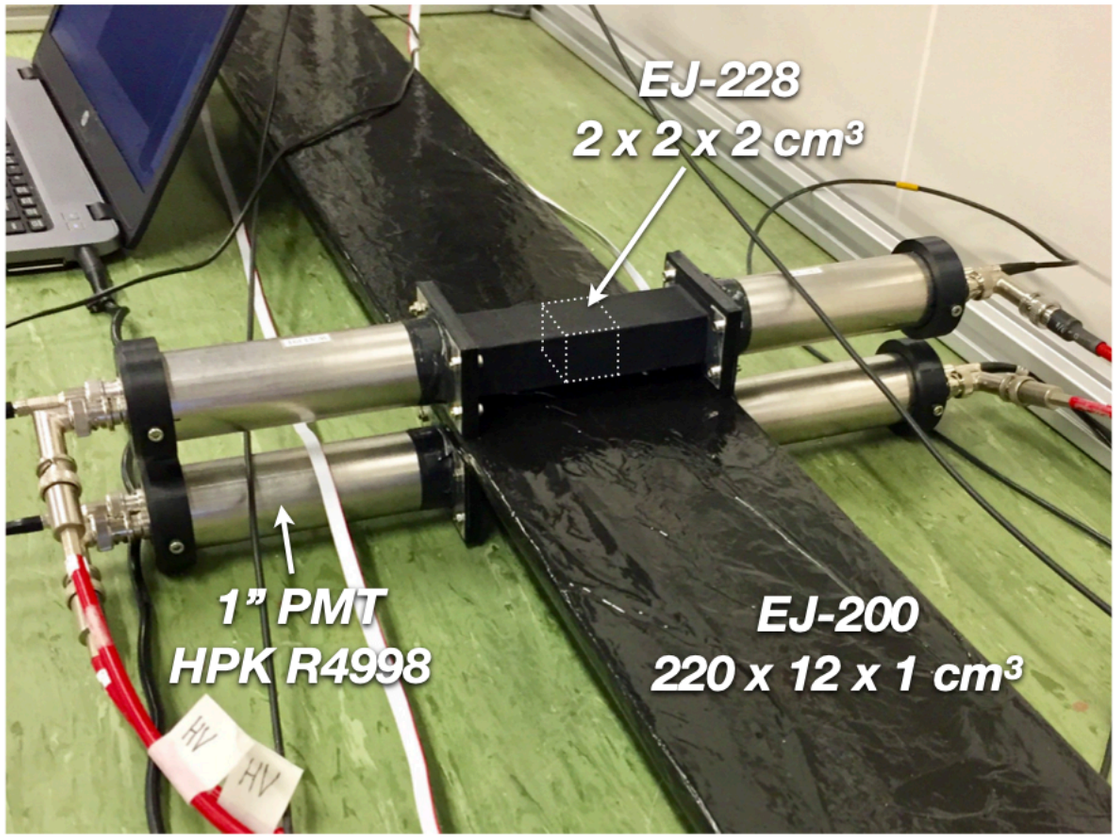
- In HEP, often one can measure the response of a detector, but modeling it is difficult. E.g  the ToF detector we are preparing for the ND280 upgrade

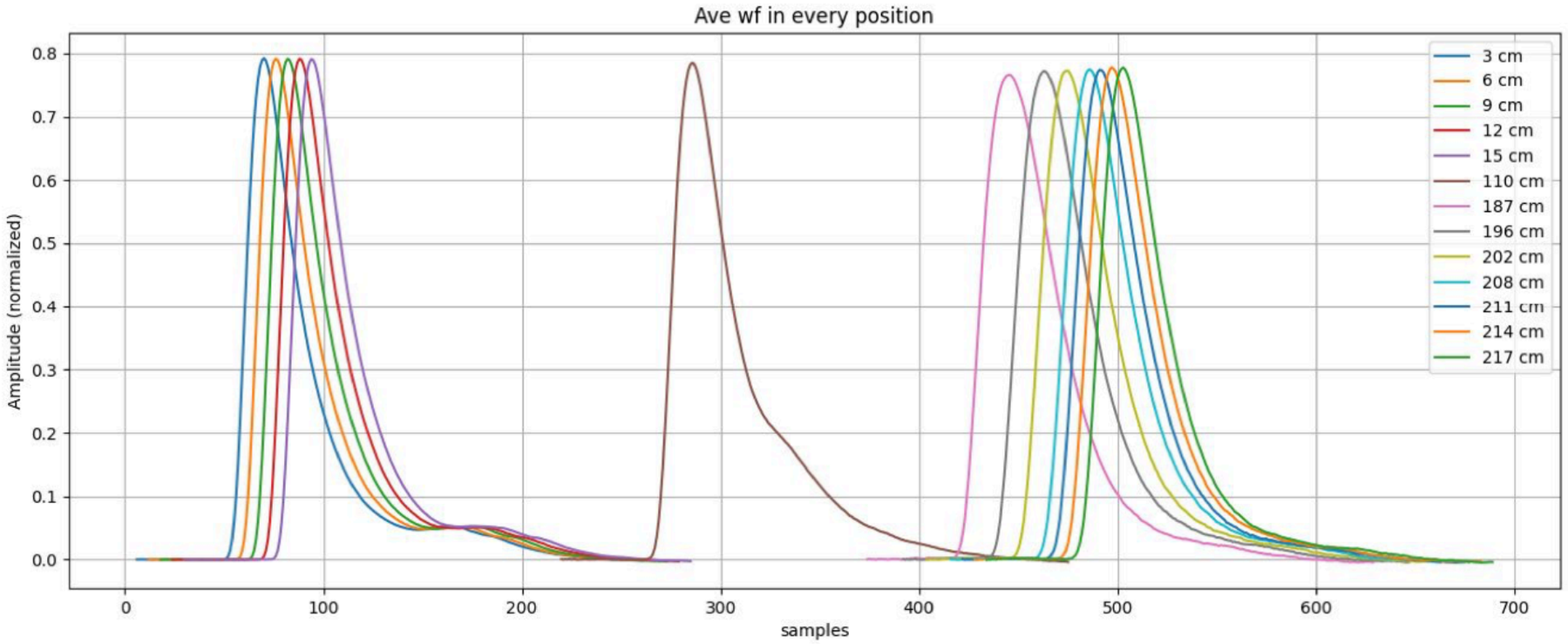


# Introduction



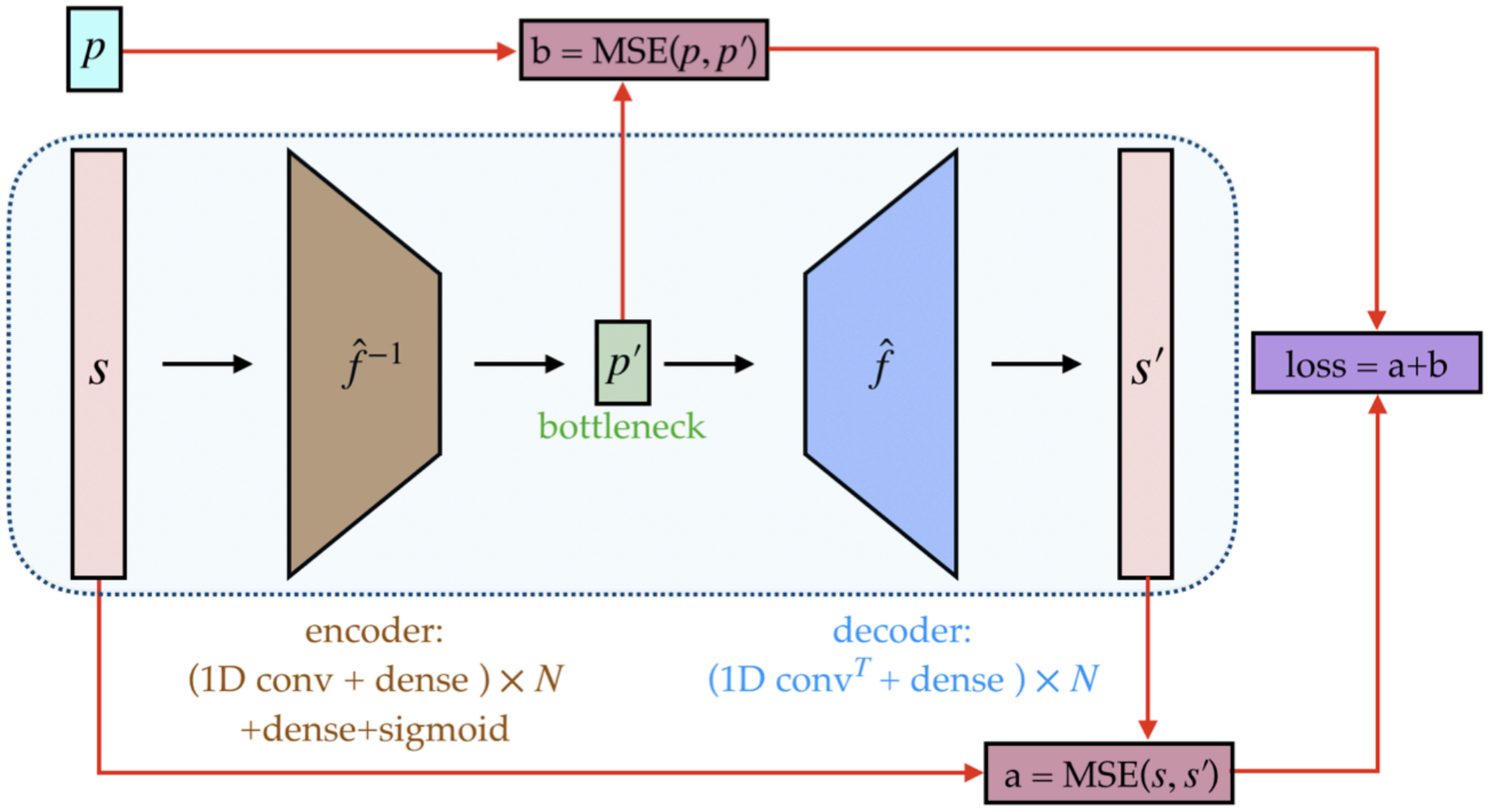


# Introduction

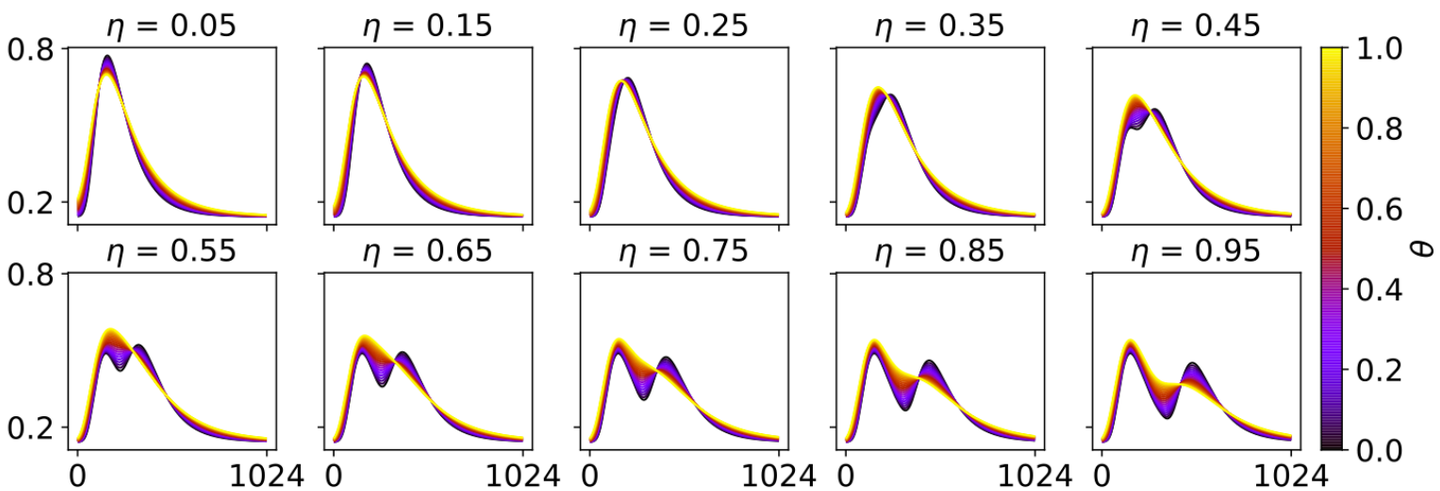
- We can measure the typical signals for all distances. We can learn this relation directly from data?

**The idea:**

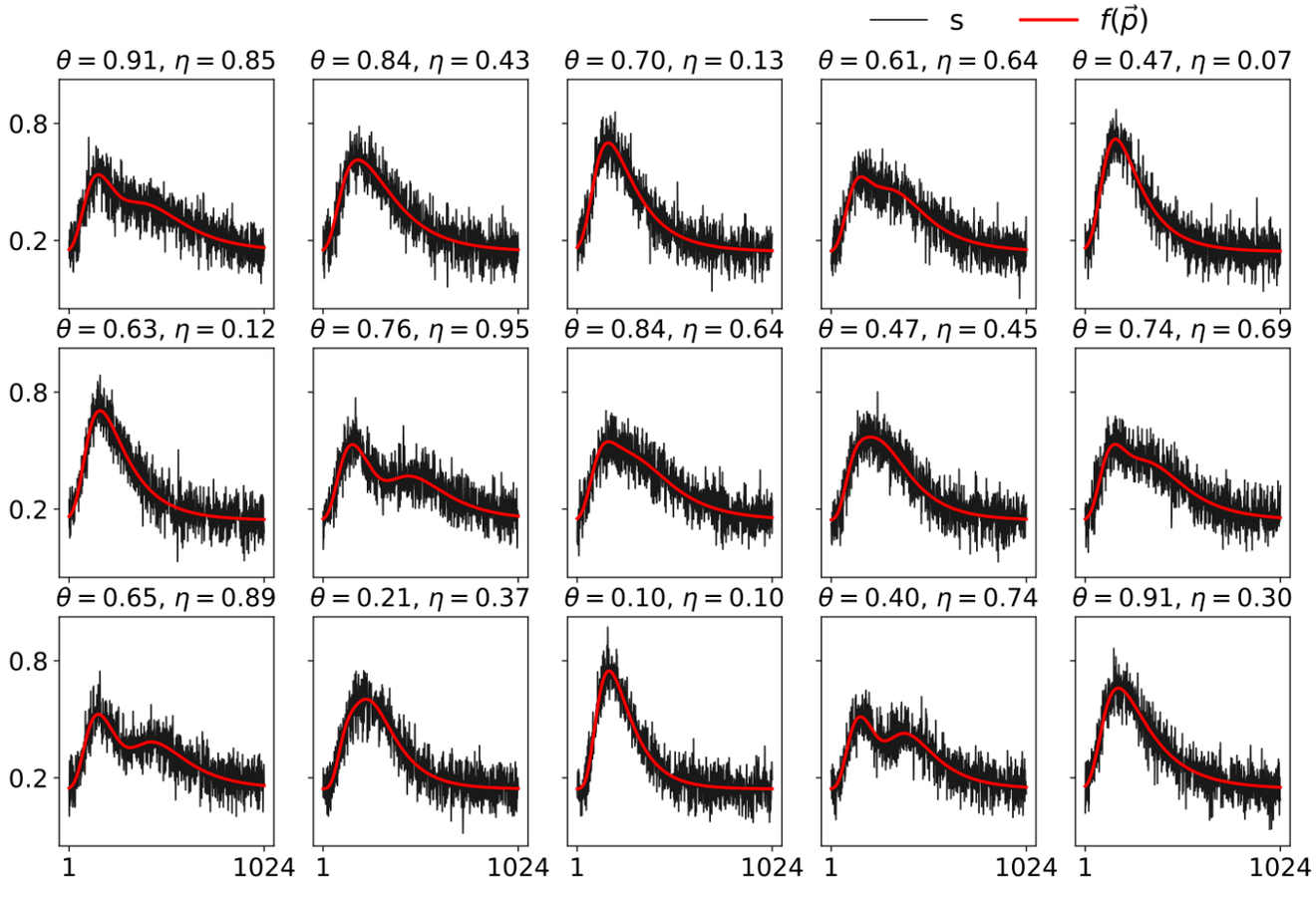


# Methodology

- Create a function (toy model) to generate synthetic data:

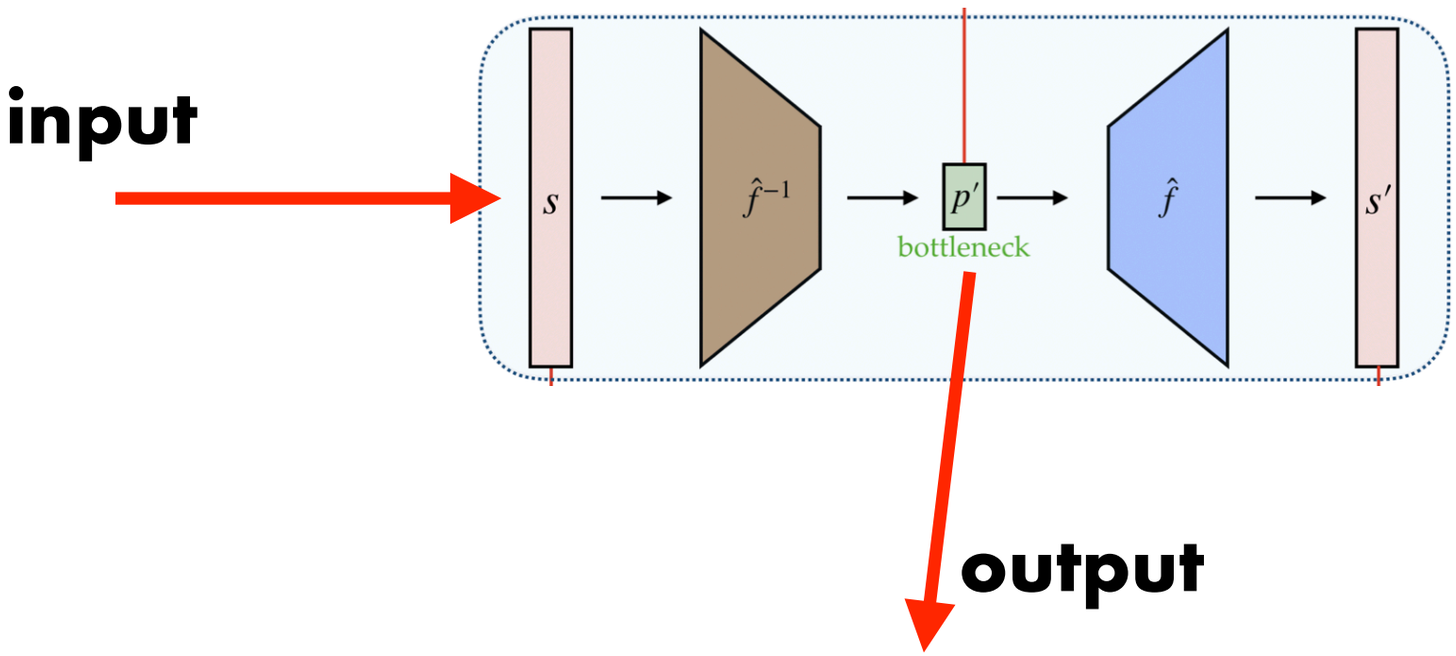


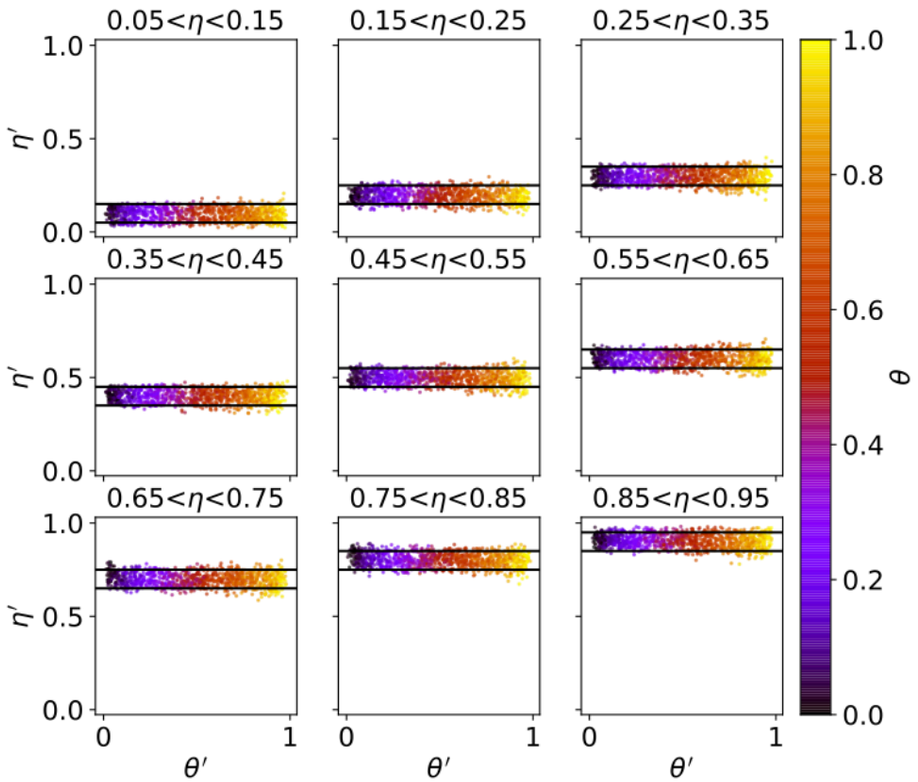

- Add noise:

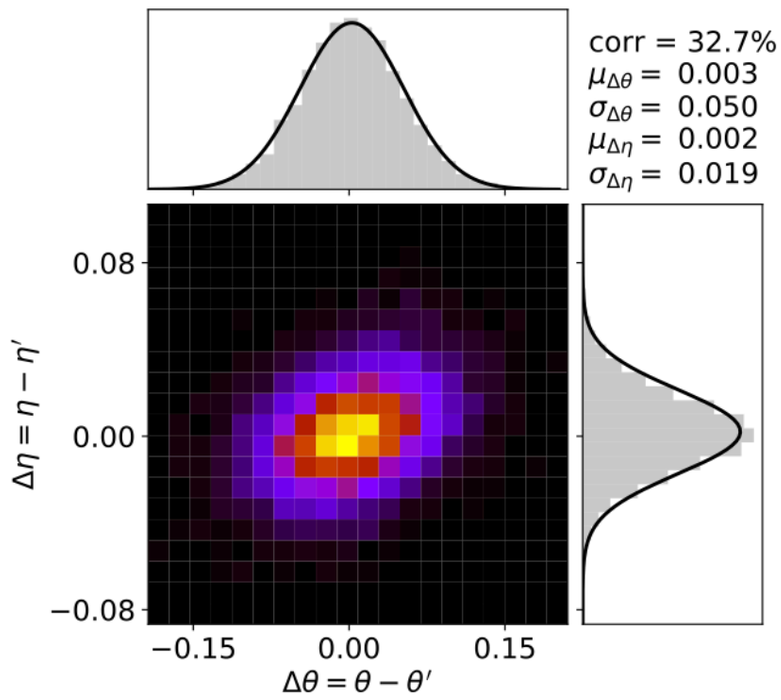



**and train!**

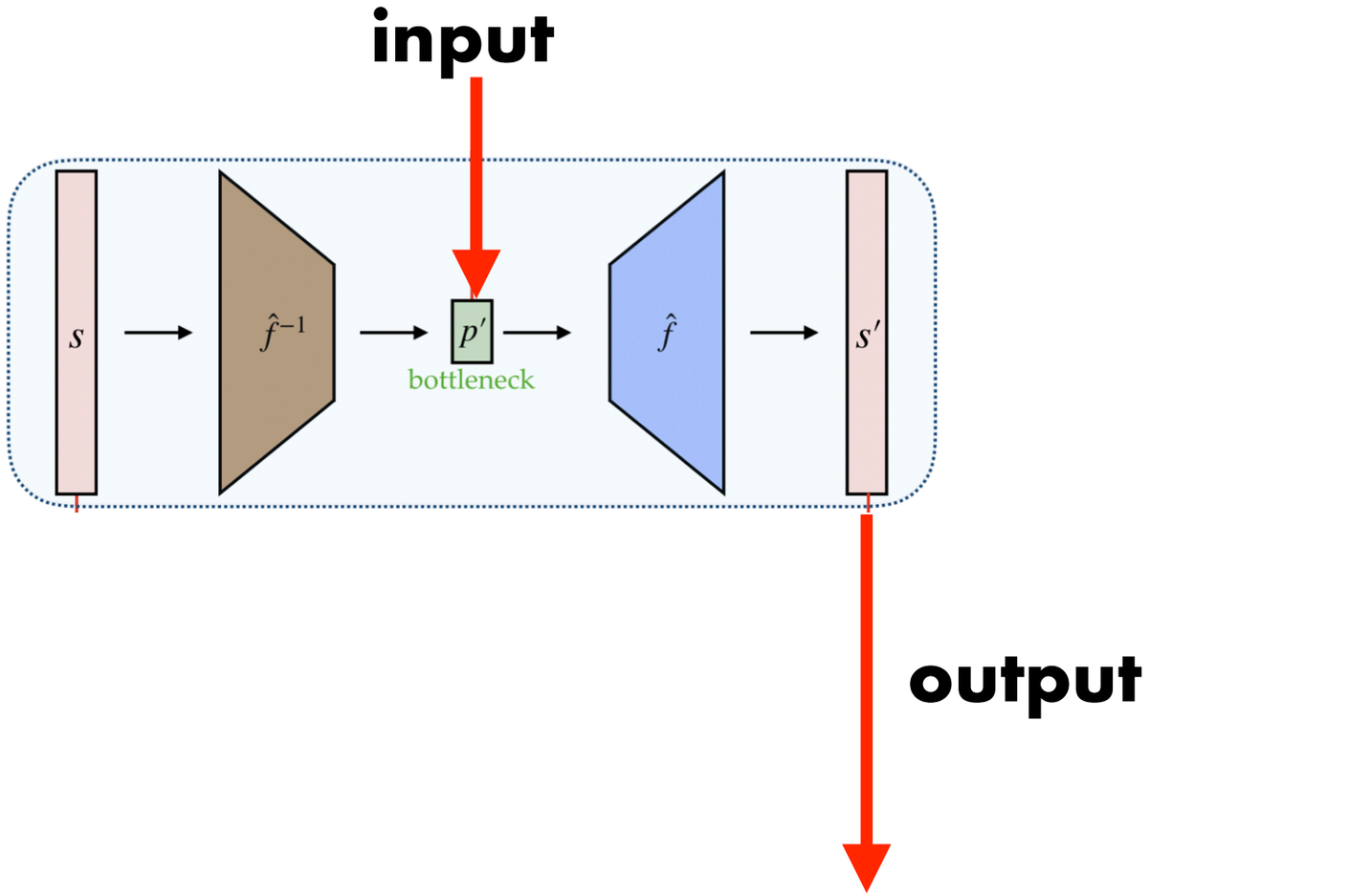
# Results

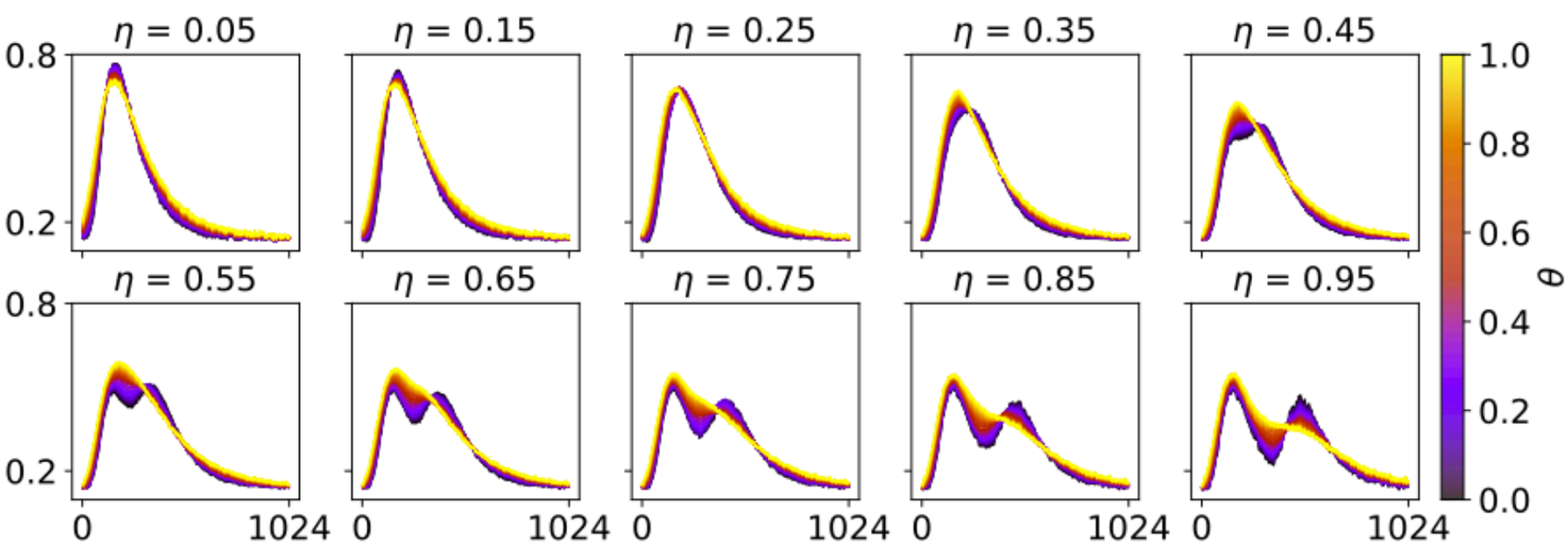

input
$s$
$\hat{f}^{-1}$
$p'$
bottleneck
$\hat{f}$
$s'$
output


0.05<η<0.15
0.15<η<0.25
0.25<η<0.35
0.35<η<0.45
0.45<η<0.55
0.55<η<0.65
0.65<η<0.75
0.75<η<0.85
0.85<η<0.95
η′
θ′
θ

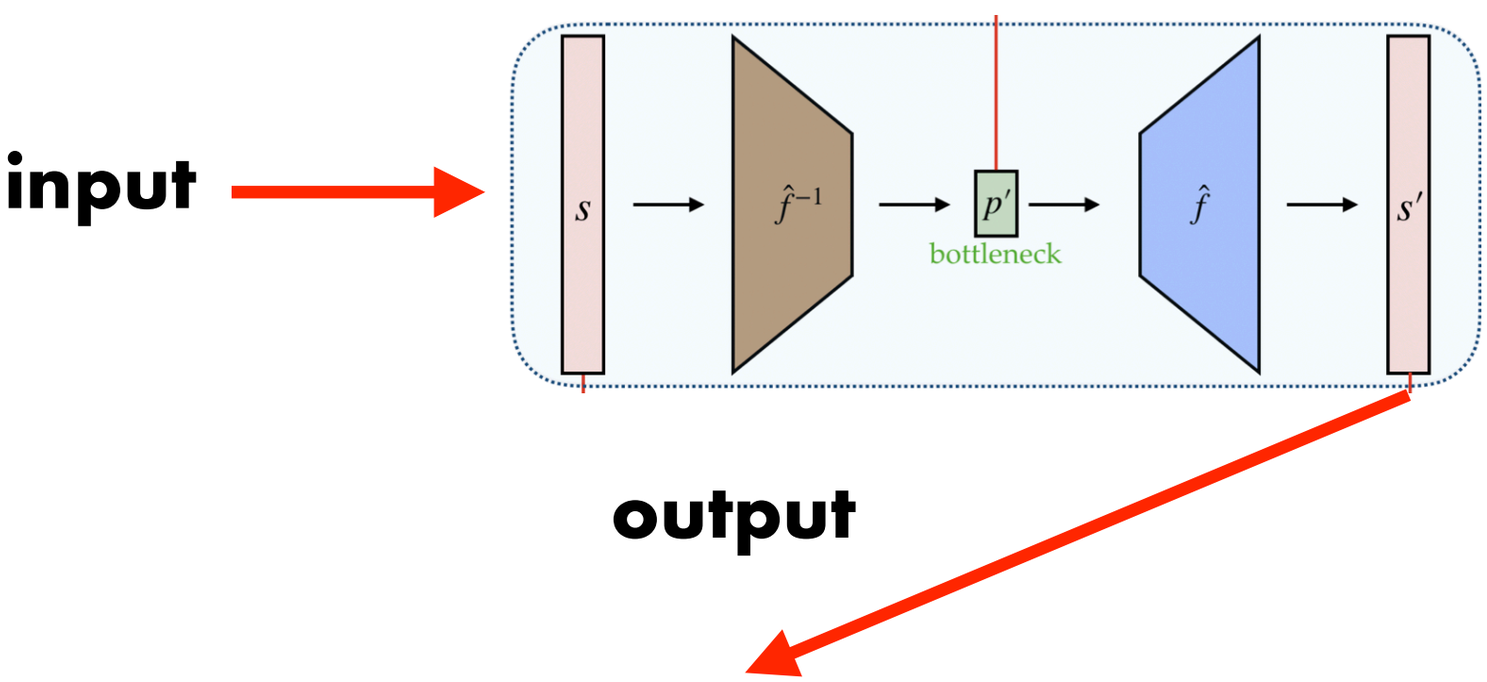

corr = 32.7%
$\mu_{\Delta\theta}$ = 0.003
$\sigma_{\Delta\theta}$ = 0.050
$\mu_{\Delta\eta}$ = 0.002
$\sigma_{\Delta\eta}$ = 0.019
$\Delta\eta = \eta - \eta'$
$\Delta\theta = \theta - \theta'$

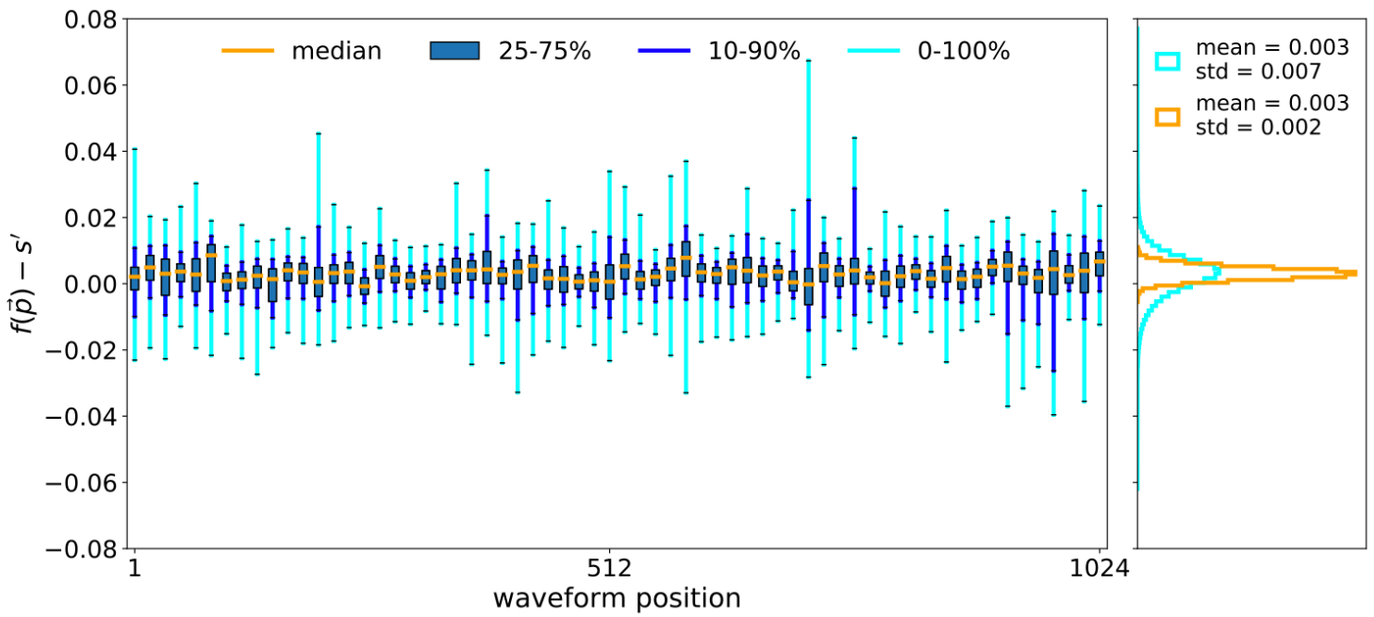
# Results

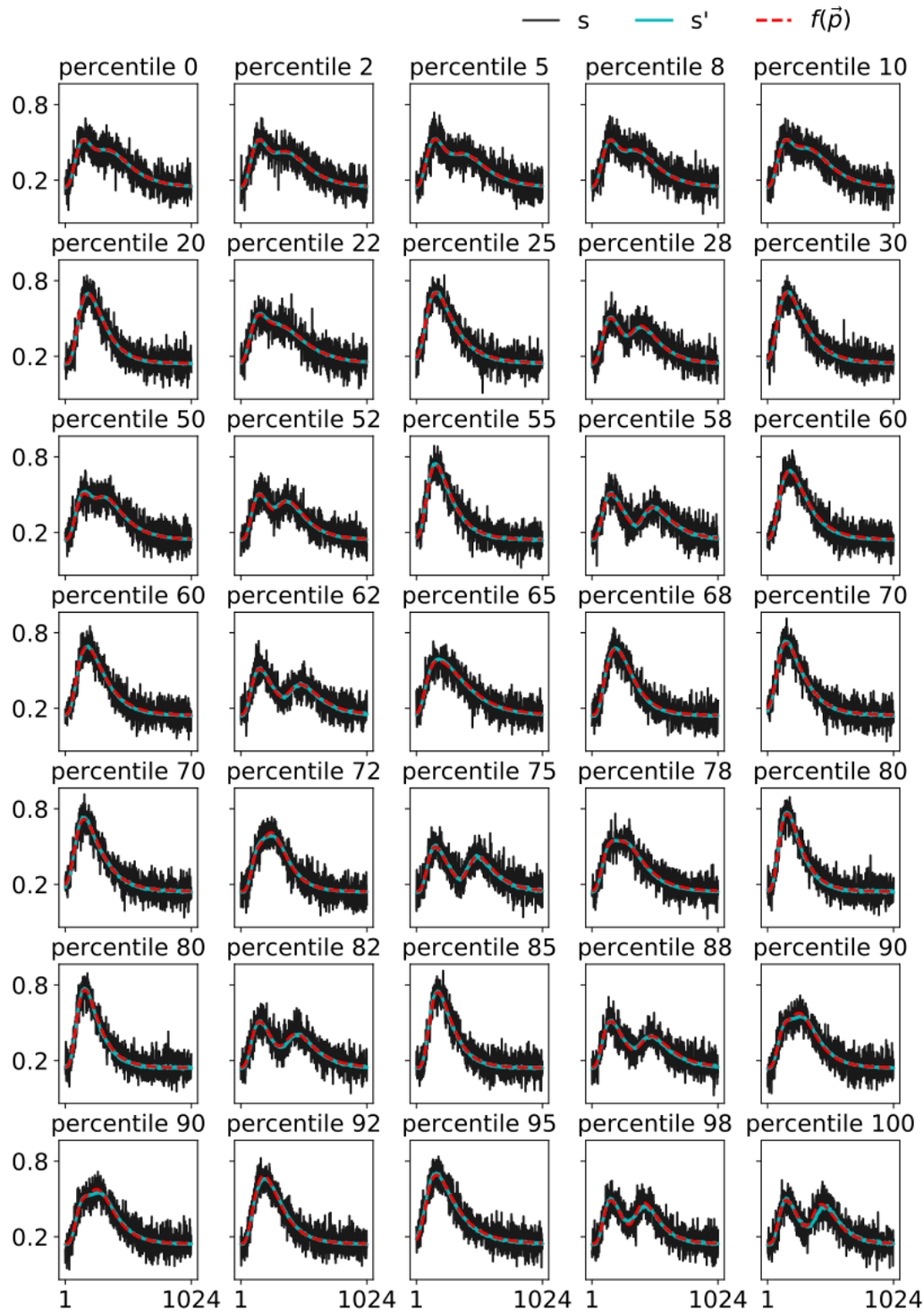

input
$s$
$\hat{f}^{-1}$
$p'$
bottleneck
$\hat{f}$
$s'$
output


$\eta$ = 0.05
$\eta$ = 0.15
$\eta$ = 0.25
$\eta$ = 0.35
$\eta$ = 0.45
$\eta$ = 0.55
$\eta$ = 0.65
$\eta$ = 0.75
$\eta$ = 0.85
$\eta$ = 0.95
0.8
0.2
0
1024
1.0
0.8
0.6
0.4
0.2
0.0
$\theta$

# Results


input
$s$
$\hat{f}^{-1}$
$p'$
bottleneck
$\hat{f}$
$s'$
output


median
25-75%
10-90%
0-100%
$f(\vec{p}) - s'$
waveform position
mean = 0.003
std = 0.007
mean = 0.003
std = 0.002


s
s'
$f(\vec{p})$
percentile 0
percentile 2
percentile 5
percentile 8
percentile 10
percentile 20
percentile 22
percentile 25
percentile 28
percentile 30
percentile 50
percentile 52
percentile 55
percentile 58
percentile 60
percentile 60
percentile 62
percentile 65
percentile 68
percentile 70
percentile 70
percentile 72
percentile 75
percentile 78
percentile 80
percentile 80
percentile 82
percentile 85
percentile 88
percentile 90
percentile 90
percentile 92
percentile 95
percentile 98
percentile 100

# Conclusions

- The algorithm shows very good performance.
- It can be used (potentially) for a very large number of problems.
- Got only minor comments from JINST, which I expect to address this week.