Experimental signal-noise ratio enhancing of magnetotelluric time series with machine learning

Author: Ing. José R. Campos.,
Supervisor: Prof. PhD. Alex Marcuello.

Universitat de Barcelona

Universitat Autònoma de Barcelona

Presentation Outline

Introduction	Methodology	Results	Discussion	Conclusions
• The problem • the solution • hypothesis & Objectives • Fundamentals	• Get the Data • Feature Engineering • Train the Model • Validate the Model • Make Predictions	• First Simulation • Second simulation • Real data denoising	What does these results mean and the implications on the subject.	What is the main contribution to the field.

Problem statement line of thinking

Electromagnetic Methods (EM)

- Characterize and monitor reservoirs → resistivity data

Magnetotelluric (MT)

- Difficulty to obtain quality data in noisy environments

Previous research

- Robust methods that hardcode specific denoising.

Machine Learning (ML)

- Independent of the type of noise; relationship between the EM components $E = Z * B$.

Hypotheses

Machine Learning provides an ideal solution to find the electromagnetic fields relationship in noisy data.

The relationship between the electric and magnetic fields can be useful for ML, even more if working in the frequency domain.

Stockwell (S) transform can provide a good conversion from time to frequency representation to achieve better results.

Objectives

To established and evaluate a workflow as a new methodology for treating Magnetotelluric noise with Machine Learning.

To use up-to-date open source Python libraries for the noise-to-signal ratio enhancing .

To implement the S-transform as the time-frequency representation (TFR) to achieve better results from the ML mode.

MT Fundamentals

1. Assume noise type and hardcode the filters
2. Implement inversion method
3. Input EM components

$$\|\boldsymbol{E} - \boldsymbol{Z} \cdot \boldsymbol{B}\| = \boldsymbol{min}$$

Classical Approach

→ Obtain $\boldsymbol{Z}$ to calculate resistivities

1. Architect the ML model
2. Adjust the Hyperparameters
3. Input EM components

$$\boldsymbol{mf} : \boldsymbol{B} \cdot \boldsymbol{W} \rightarrow \boldsymbol{E}$$

ML Vision

→ Use $\boldsymbol{mf}$ to deduce resistivities

For $\boldsymbol{E}$ & $\boldsymbol{B}$ **complex** **electric** and **magnetic** fields in the **frequency** Domain

Universitat de Barcelona

Theoretical Background: Overview

Universitat Autònoma de Barcelona

Supervised Machine Learning Models

Machine Learning "Learn" With Data, Without Being Explicitly Programmed

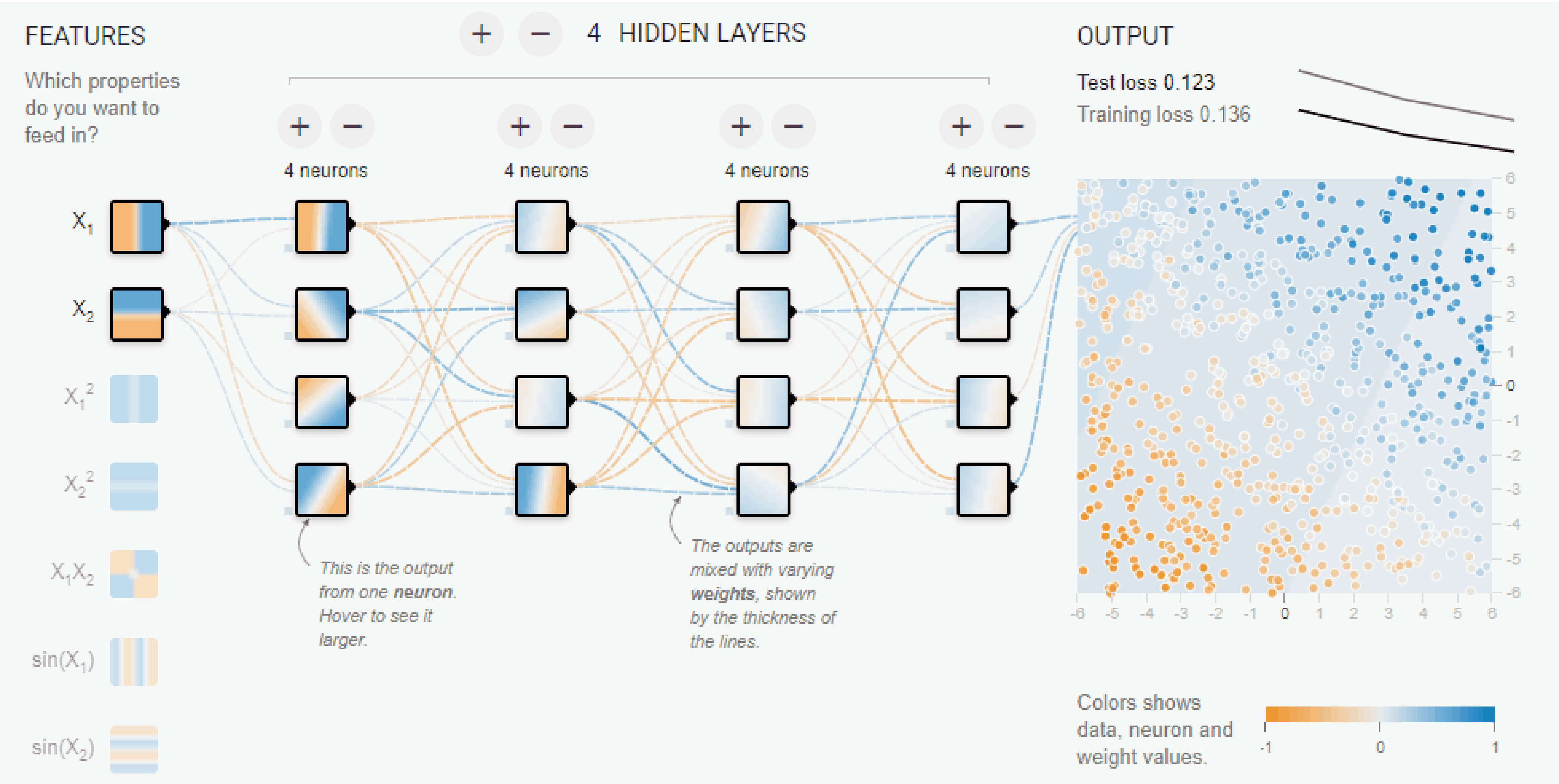
Training Data set

l: **Input** (B) → Learned Target (Ep) - **Expected Target** (E) = Loss Score

Validation Data set

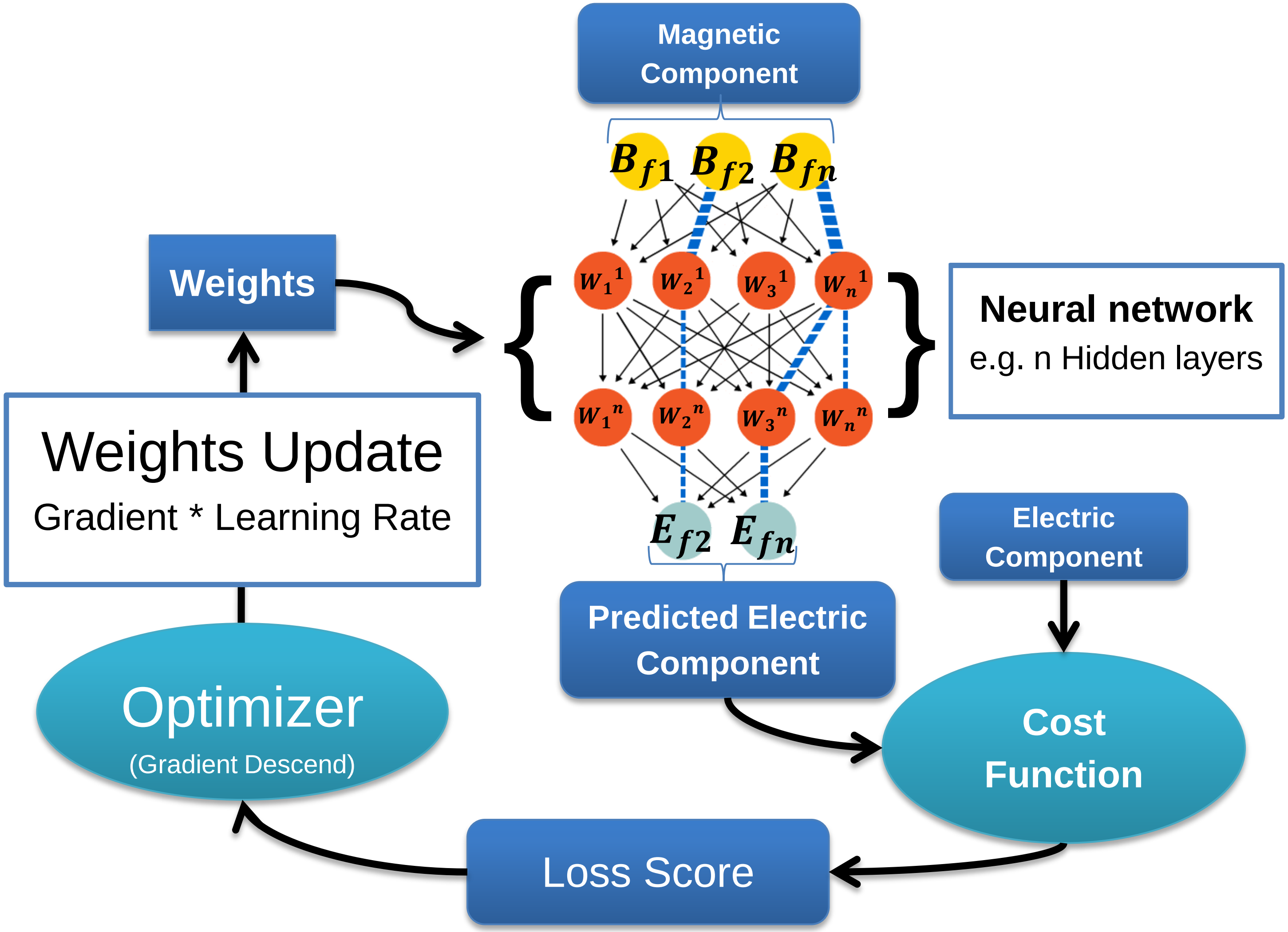
p: **Input** (B) → Predicted Target (Ep) - **Expected Target** (E) = Accuracy Score

Neural Network



http://playground.tensorflow.org

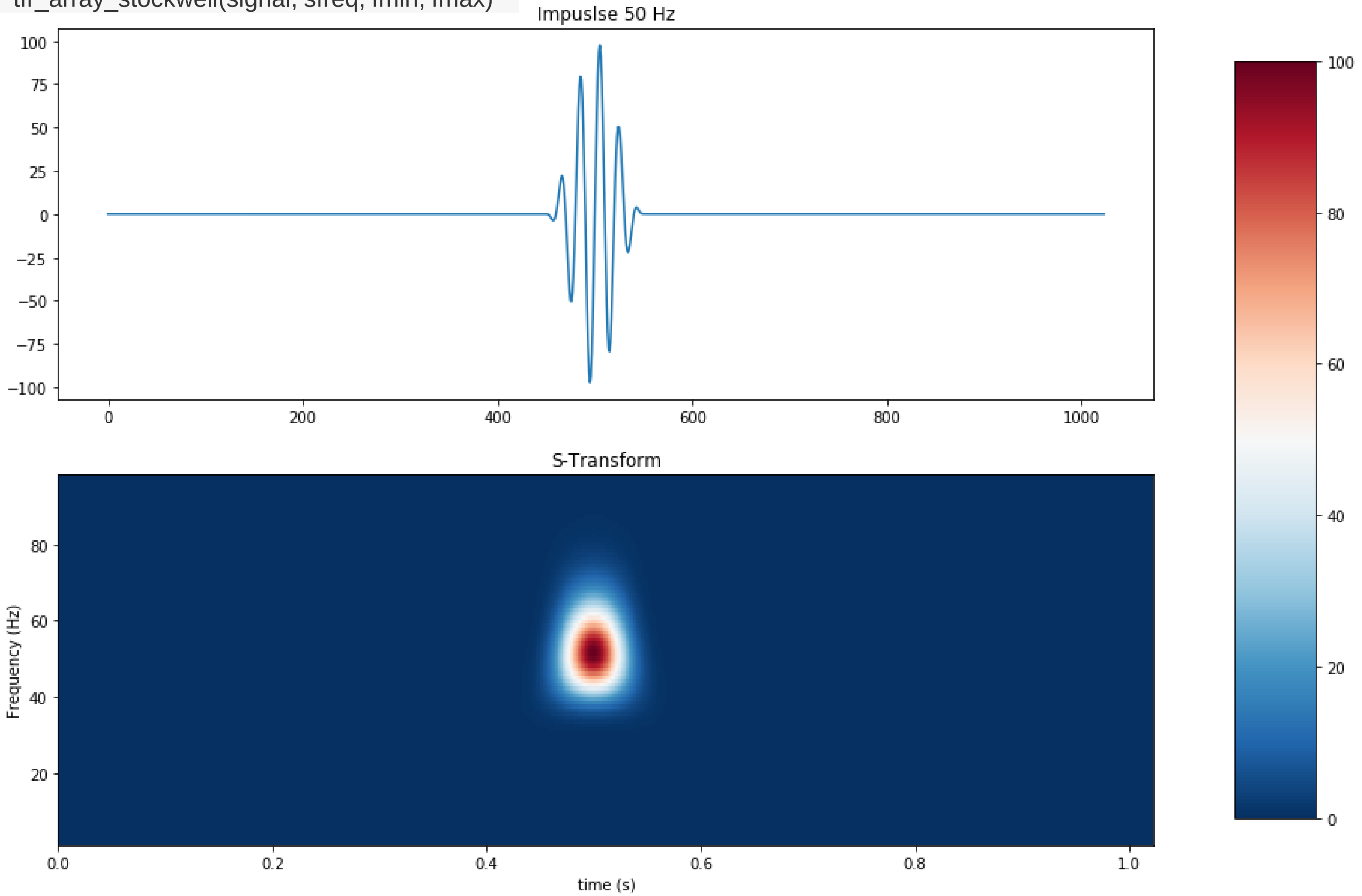
Neural Network per iteration



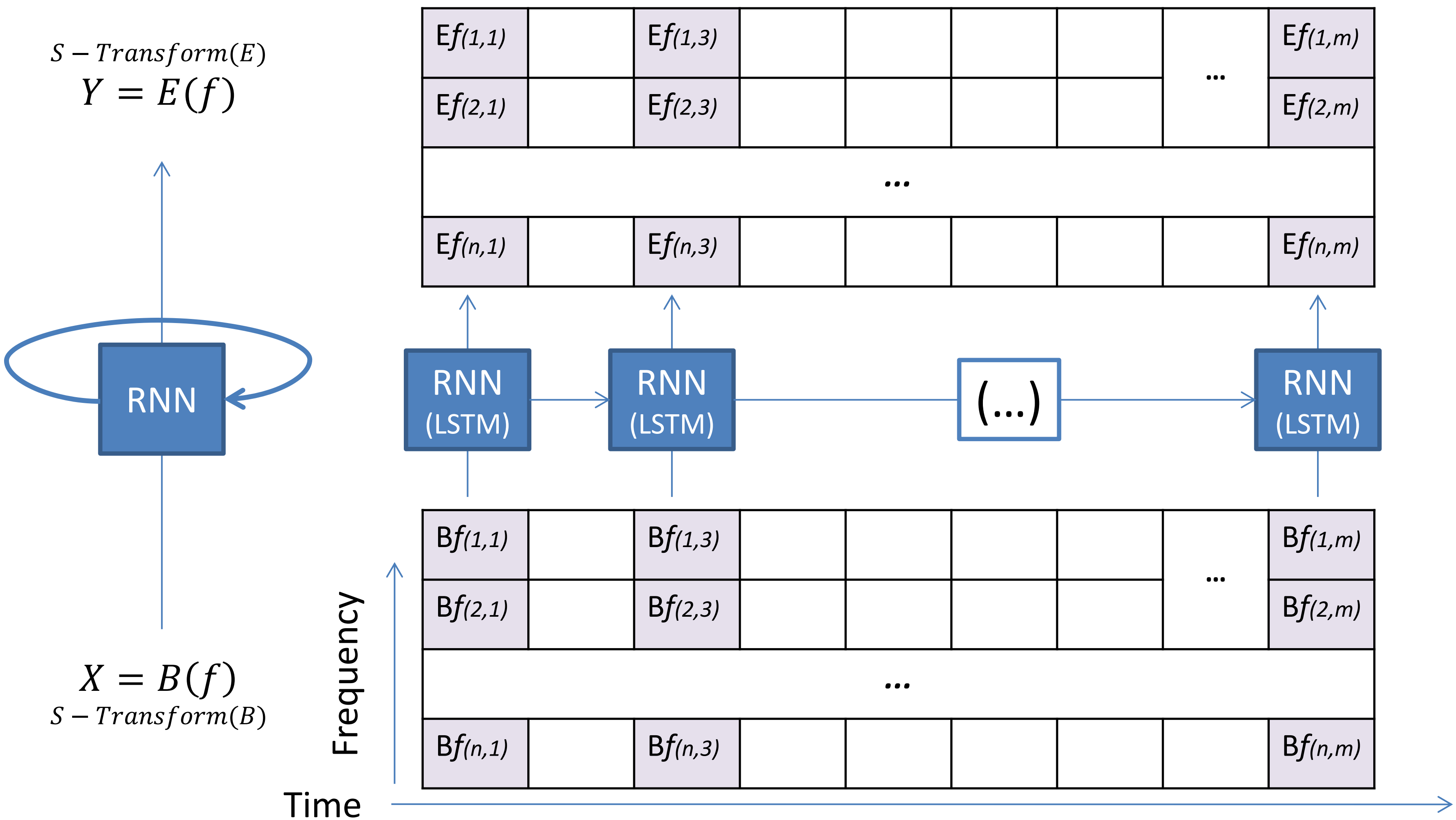
Feature engineering

```
# Calculate the S-Transform / MNE array function
from mne.time_frequency import tfr_array_stockwell
st_power =  tfr_array_stockwell(signal, sfreq, fmin, fmax)
```

Apretiation of the TFR



RNN Procedure



Methodology Summary

Get the Data → Feature Engineering → Train the Model → Validate the Model → Make Predictions

Assuming Parameters

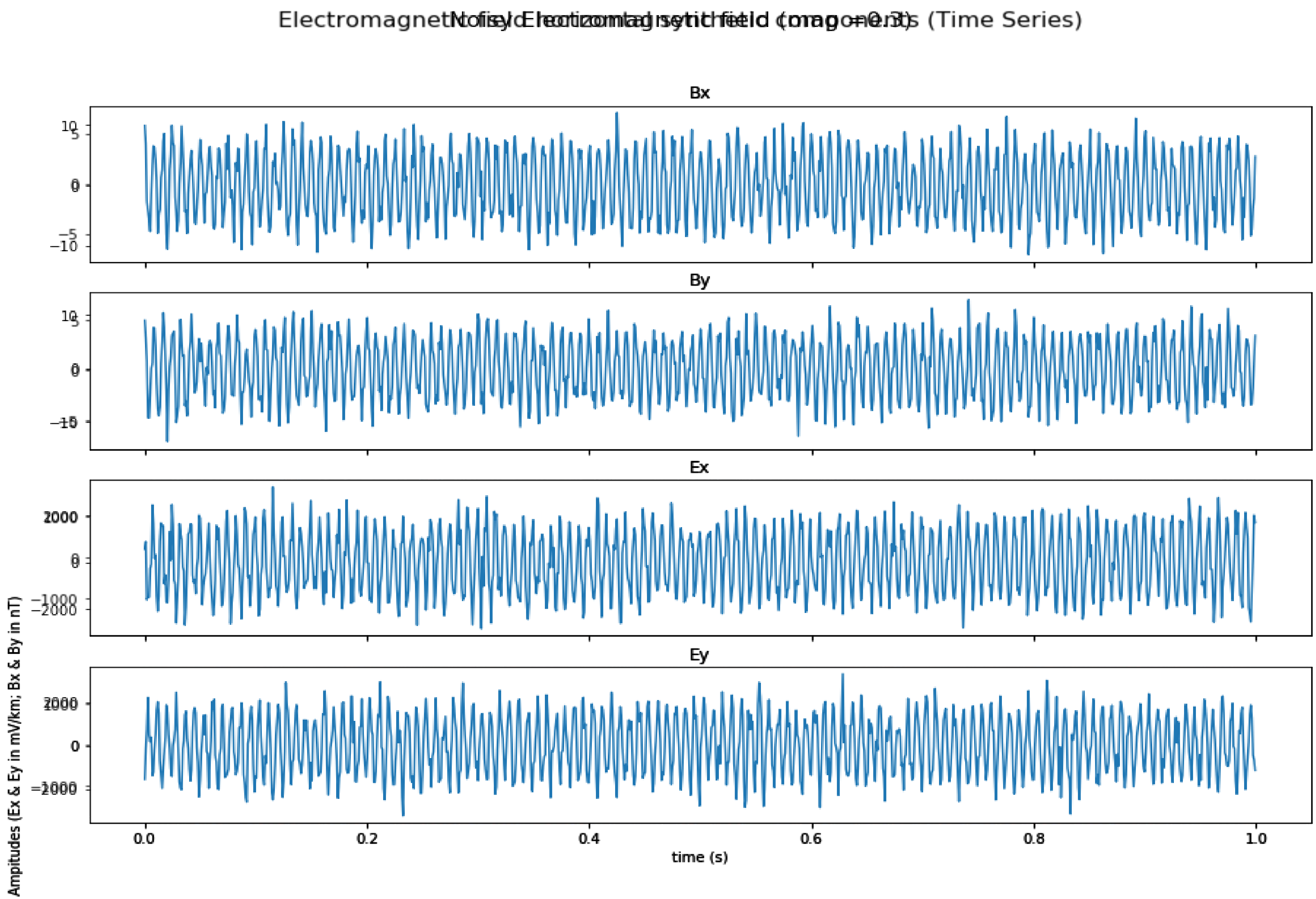
→ Generating the magnetic field

→ Generating the impedance matrix

→ Generating the electric field

→ Generating the random noise → Add the noise to the EM fields

Methodology Summary



Methodology Summary

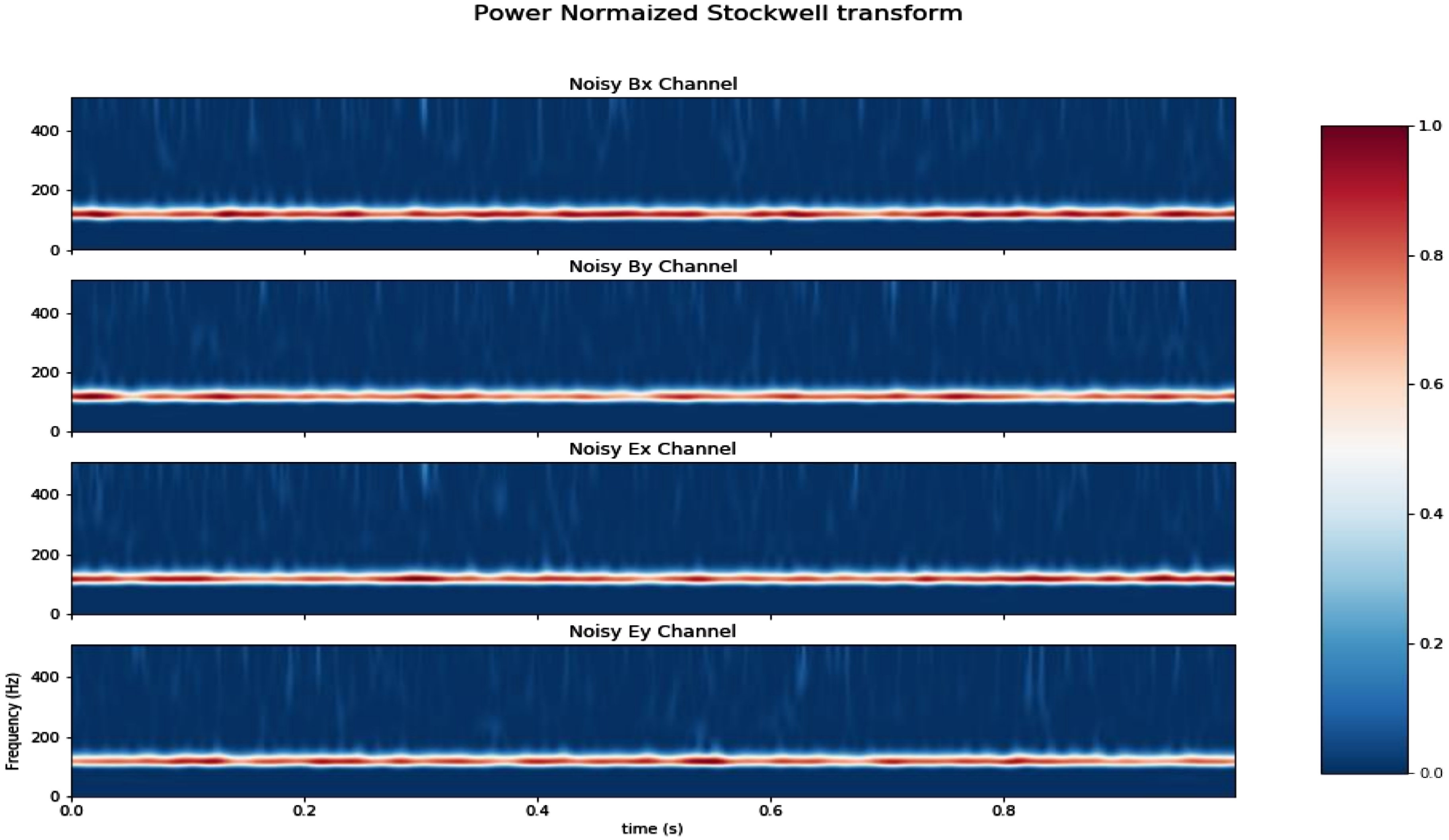
Get the Data | **Feature Engineering** | Train the Model | Validate the Model | Make Predictions

Add the noise to the EM fields

→ Apply the S-Transform to the signals (field components)

→ Obtain the power matrices

Methodology Summary



Get the Data → Feature Engineering → **Train the Model** → Validate the Model → Make Predictions

B_x X_Train

E_y Y_Train

B_y X_Val

E_x Y_Val

Obtain the power matrices

→ Set Training (Bx maps to Ey) and Validation (By to Ex) Data

→ Define the architecture of the model

→ Configure the loss function and the optimizer

→ Train (fit) the model

Universitat de Barcelona

Alternative ML Model for real data

UAB Universitat Autònoma de Barcelona

Get the Data → Feature Engineering → **Train the Model** → Validate the Model → Make Predictions

B_x X_Train

$B_x{}^s$ Y_Train

B_x X_Val

$B_x{}^s$ Y_Val

→ Split Training and Validation Data as subsets of the same component at different time segments

→ Time shift the same subset to use it as labels E.g.

$$\mathrm{T}[1{:}10] \;\rightarrow\; X^{train} = B_x[1{:}5] \;\&\; Y^{train} = B_x{}^{shift}[1{:}5]$$

$$\rightarrow\; X^{val} = B_x[5{:}10] \;\&\; Y^{val} = B_x{}^{shift}[5{:}10]$$

Methodology Summary

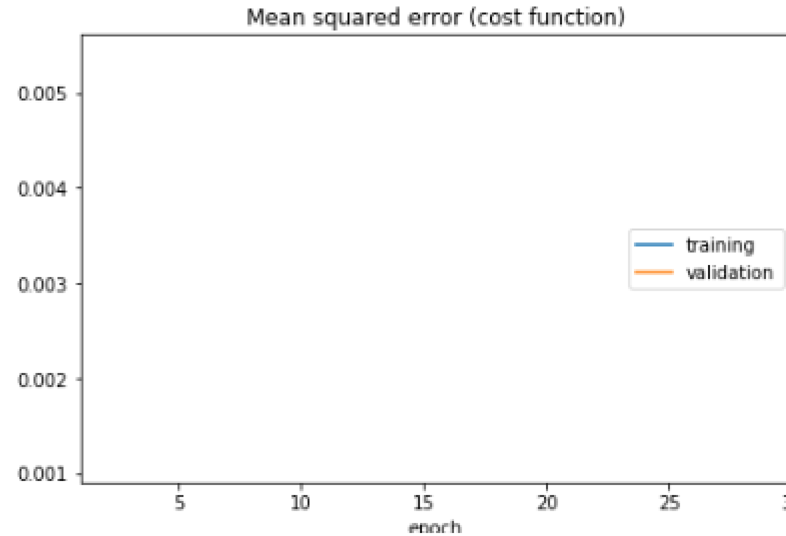
Get the Data → Feature Engineering → Train the Model → **Validate the Model** → Make Predictions

```
layers = [LSTM(…),
          LSTM(…),
          LSTM(…),
          Dense(…)]

model = Sequential(layers)

model.compile(loss='mean_squared_error',
              optimizer='rmsprop',
              metrics=['accuracy'])

history = model.fit(x_train, y_train,
                    epochs=30,
                    validation_data=(x_val, y_val))
```



Methodology Summary

Get the Data → Feature Engineering → Train the Model → Validate the Model → Make Predictions

Train (fit) the model

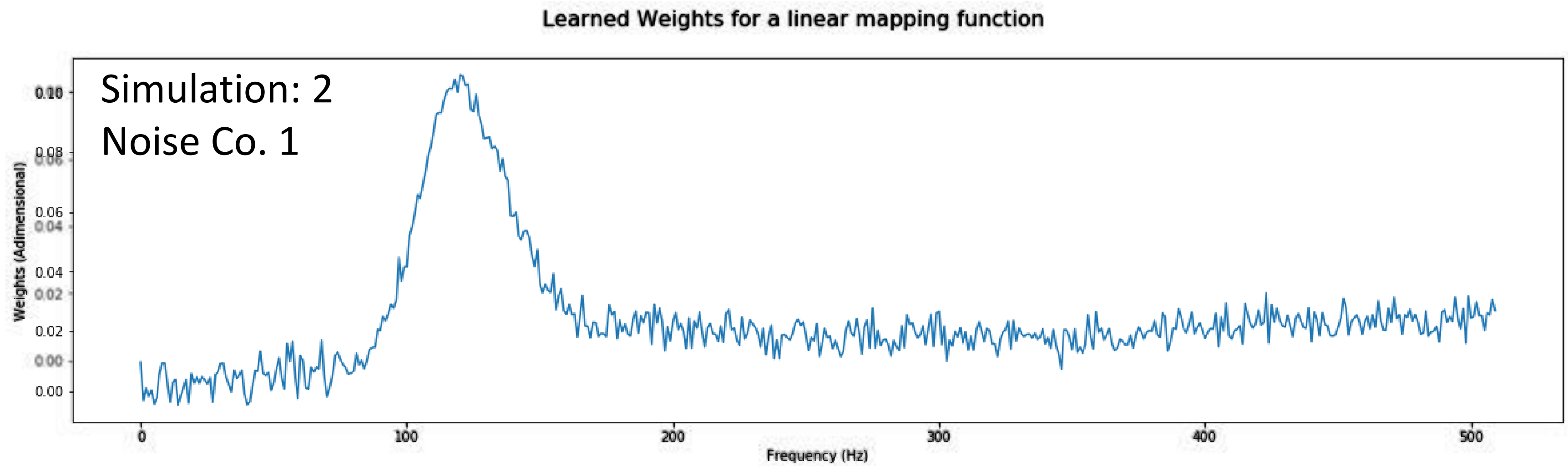
→ Obtain the Mapping Function (MP)

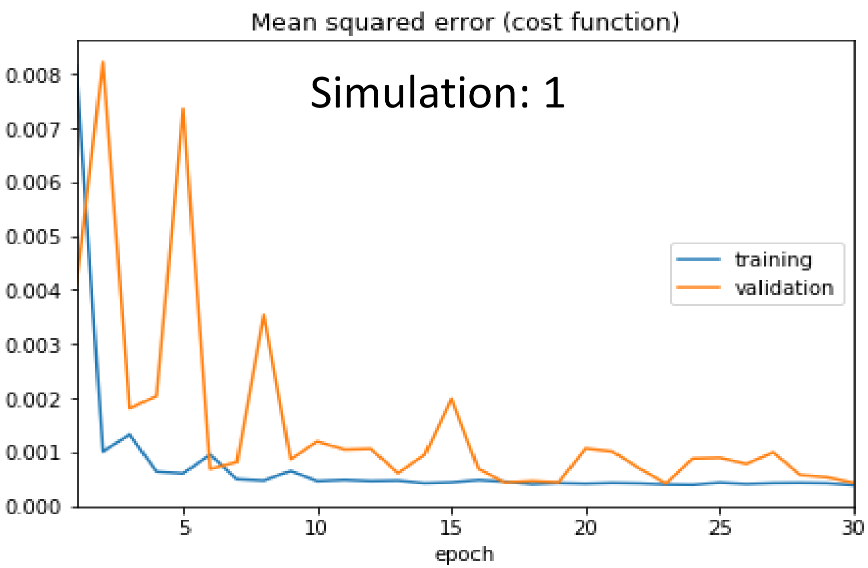
→ Use the ML to predict denoised E components

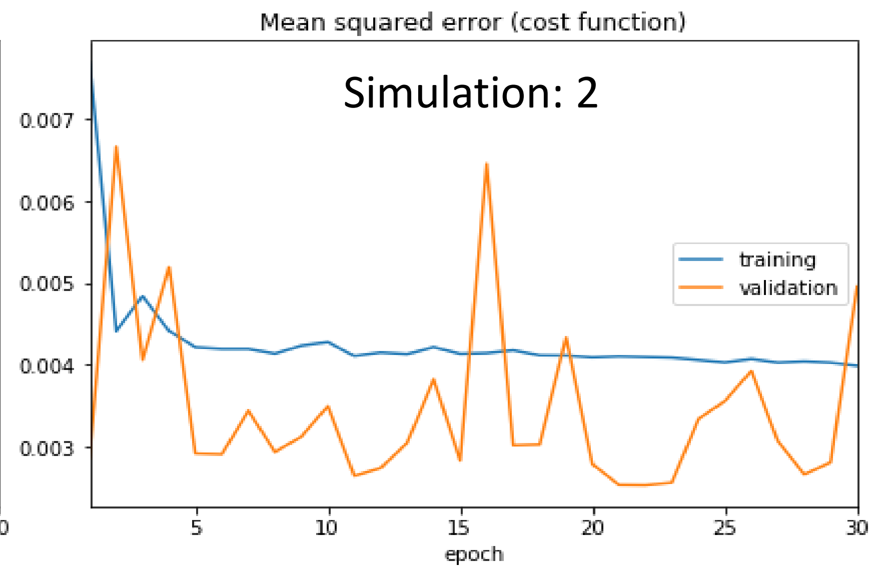
Universitat de Barcelona

Universitat Autònoma de Barcelona

Metrics 1^{rst} & 2^{nd} Simulations





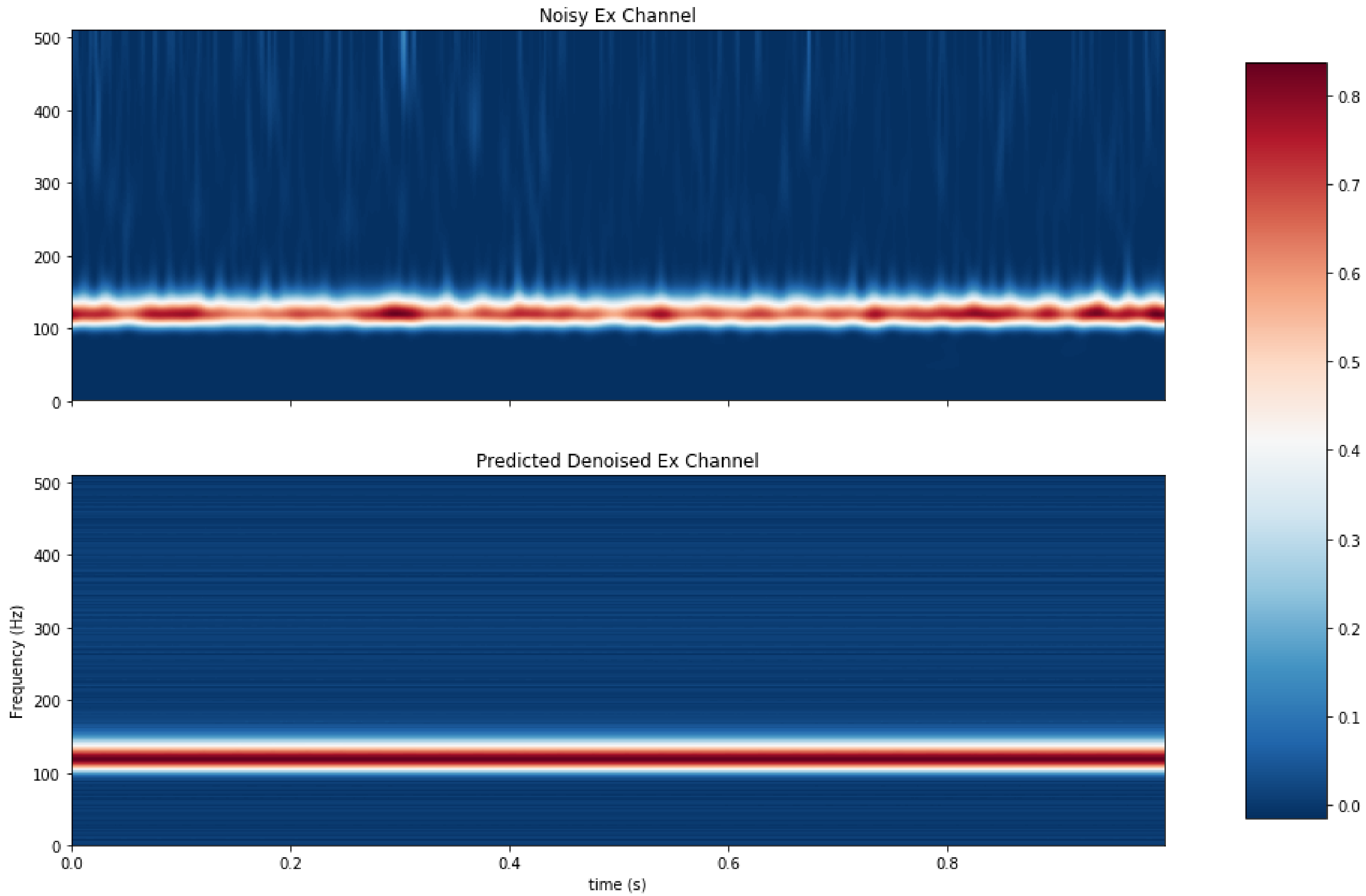


Universitat de Barcelona

Results 1^{rst} Simulation

Universitat Autònoma de Barcelona

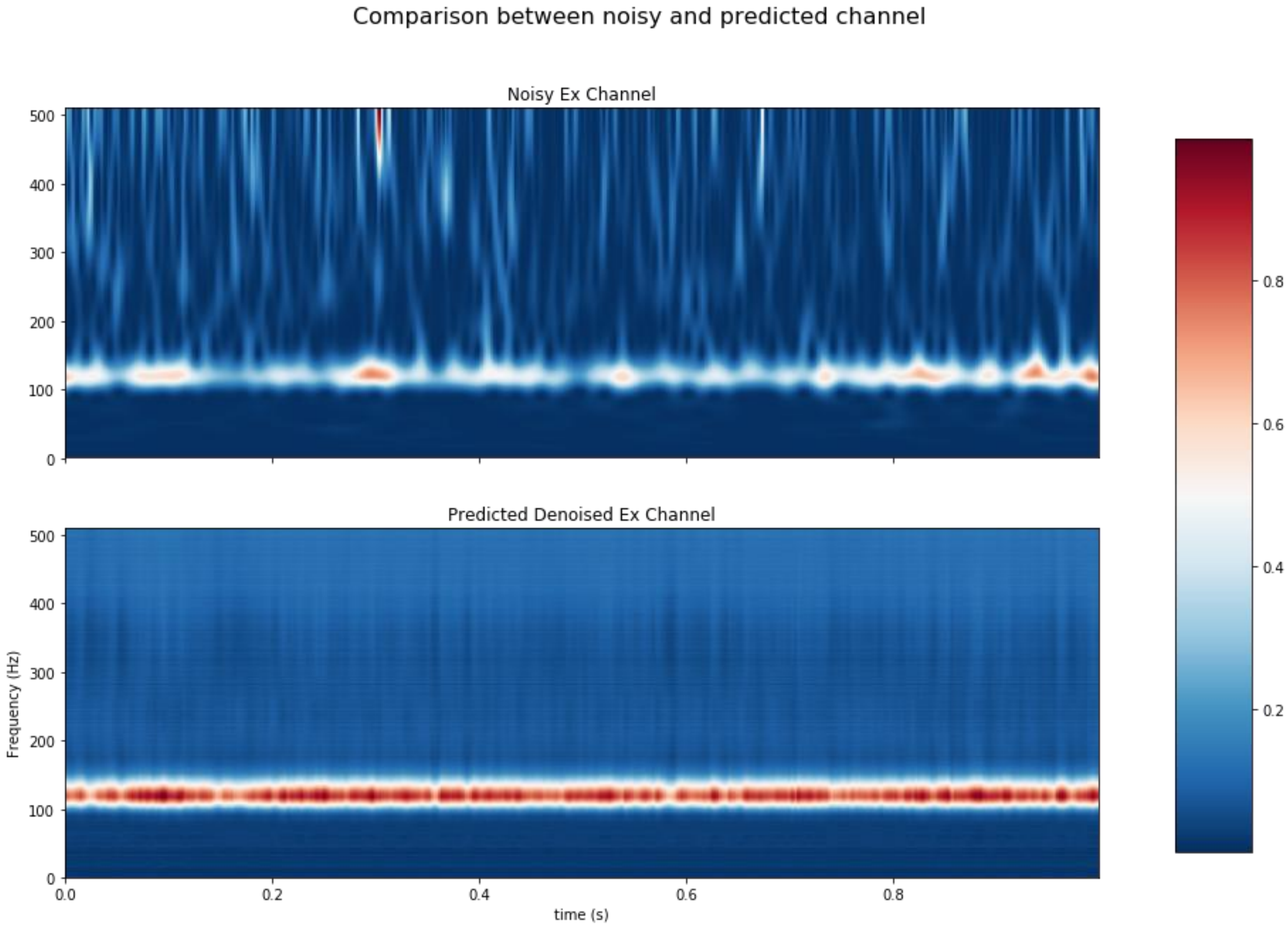
Comparison between noisy and predicted channel



Universitat de Barcelona

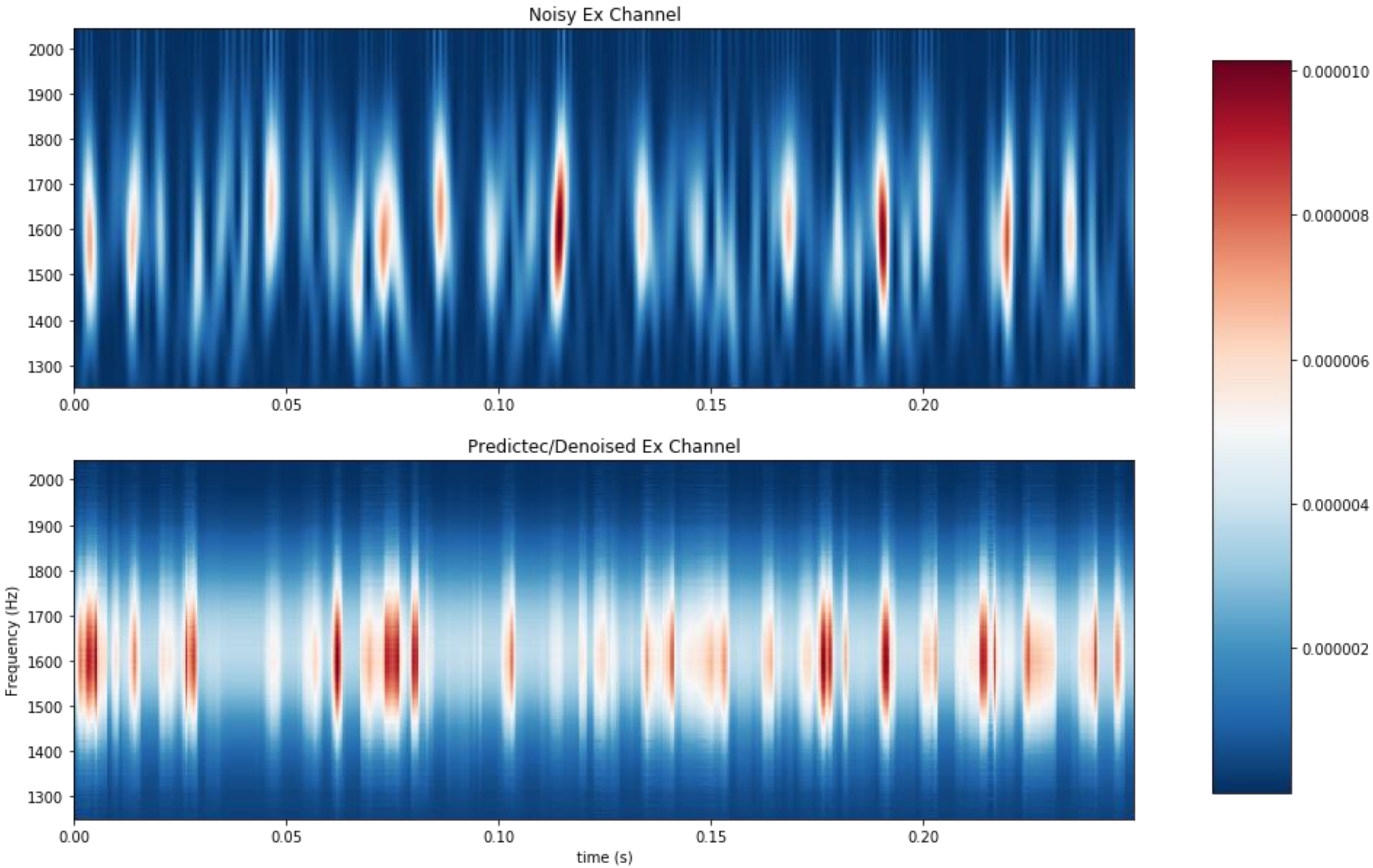
Results 2^{nd} Simulation

UAB Universitat Autònoma de Barcelona



Results Real Data

Comparison between noisy and predicted channel



What have we seen?

How do these results relate to the original objectives?

- The ML approach to denoise is accurate and an effective workflow is presented.
- Python provides an excellent programing environment for the task.
- The S-transform provides an effective tool; it success in depicting the declared fundamental frequency.

Do the data support the hypothesis?

Weaknesses of the method?

Is there another way to interpret results?

What further research would be necessary?

What is the contribution?

What have we seen?

How do these results relate to the original objectives?

Do the results support the hypothesis?

- The relationship between EM components can be used to train effective ML models.
- Results suggest that TFR as input data provides an effective feature engineering.
- S-transform provides good resolution for the input data; similar resolution is obtain from the predictions.

Weaknesses of the method?

Is there another way to interpret results?

What further research would be necessary?

What is the contribution?

What have we seen?

How do these results relate to the original objectives?

Do the data support the hypothesis?

Weaknesses of the method?

- The models works on the assumption E=ZB.
- As noise increases; readjusting the hyperparameters is needed to compensate.
- Processing power delimits the amount of data and the speed of the method CPU<GPU<TPU
- The presented case is valid for a 1D medium.

Is there another way to interpret results?

What further research would be necessary?

What is the contribution?

What have we seen?

How do these results relate to the original objectives?

Do the data support the hypothesis?

Weaknesses of the method?

Is there another way to interpret results?

- The method can be regarded as a way to determine the impedance tensor Z (Mapping Function) and with it denoise the data.

What further research would be necessary?

What is the contribution?

What have we seen?

How do these results relate to the original objectives?

Do the data support the hypothesis?

Weaknesses of the method?

Is there another way to interpret results?

What further research would be necessary?

- To implement the model on 2D medium, complex tensors should be input. In this way the MF could be used to deduce the apparent resistivities .
- Further assessment on real data to provide more validation.

What is the contribution?

What have we seen?

How do these results relate to the original objectives?

Do the data support the hypothesis?

Weaknesses of the method?

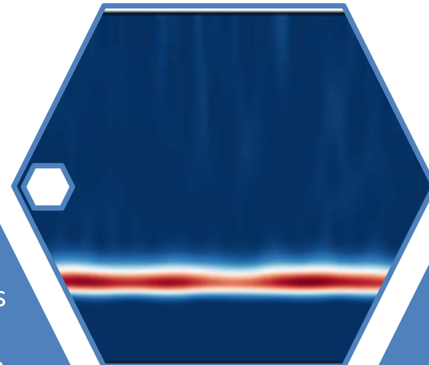
Is there another way to interpret results?

What further research would be necessary?

What is the contribution?

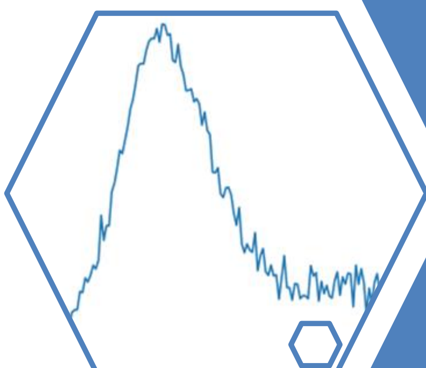
- The main contribution is the establishment of an open sourced code that can be used to denoise EM time series

What have we seen?



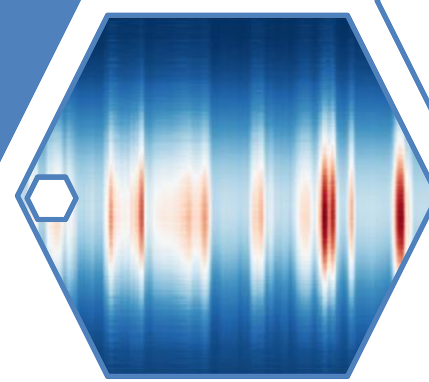
EM methods benefits from effective denoising and in turn provide better results for reservoir characterization.

The presented workflow efficiently enhances the signal to noise ratio.

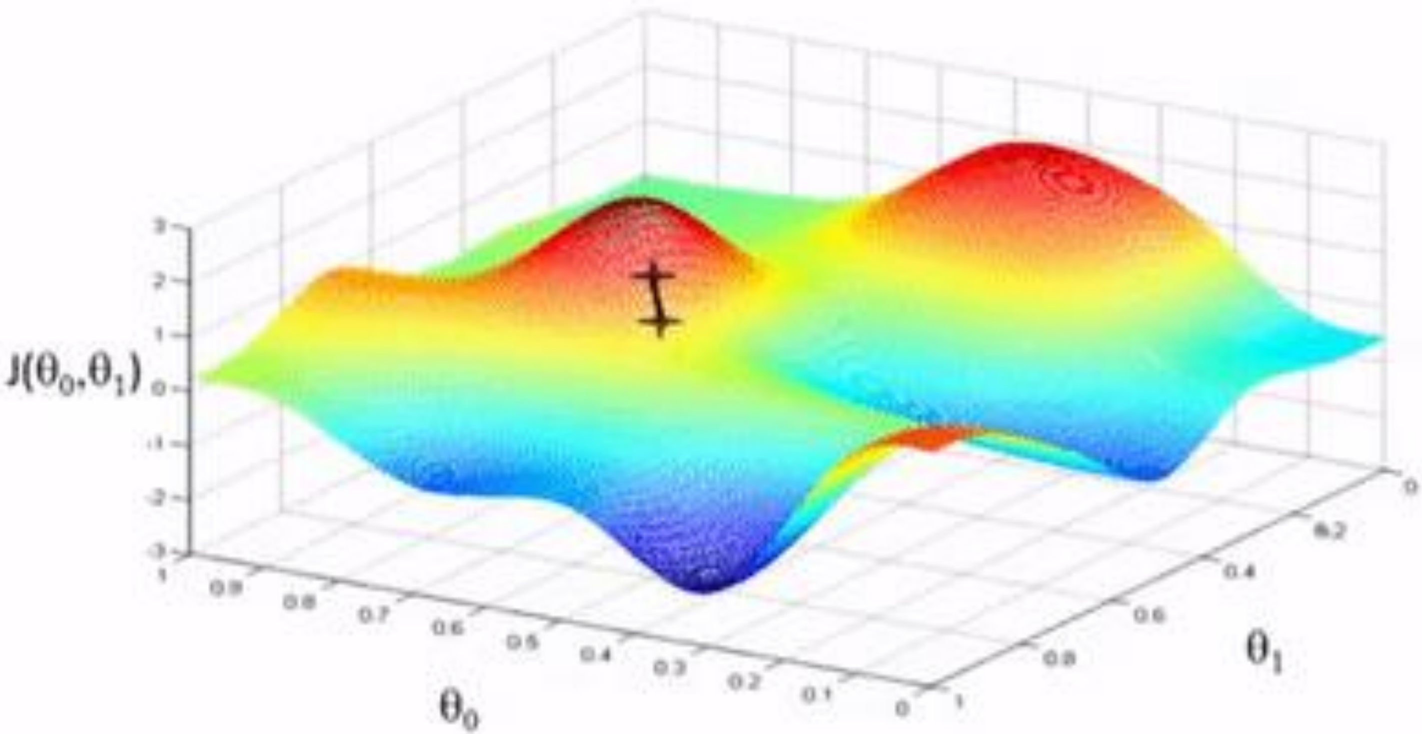


More elaborate model architectures can be design to directly find the medium resistivity values.

Future experimentation is wide open to hyperparameter adjustment and feature engineering.



Thanks for the attention



“Being able to go from idea to result with the least possible delay is key to doing good research.” Keras

Andrew Ng