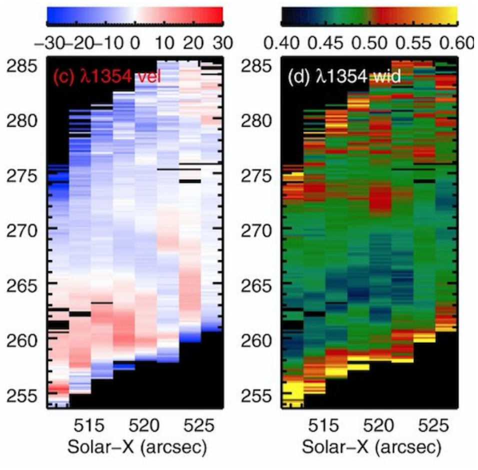
Differentiable programming for spectra modeling and inference

Carlos José Díaz Baso

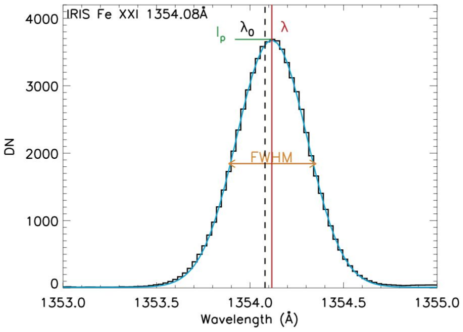
Collaborators: Luc Rouppe van der Voort, Andrés Asensio Ramos, Jaime de la Cruz Rodríguez

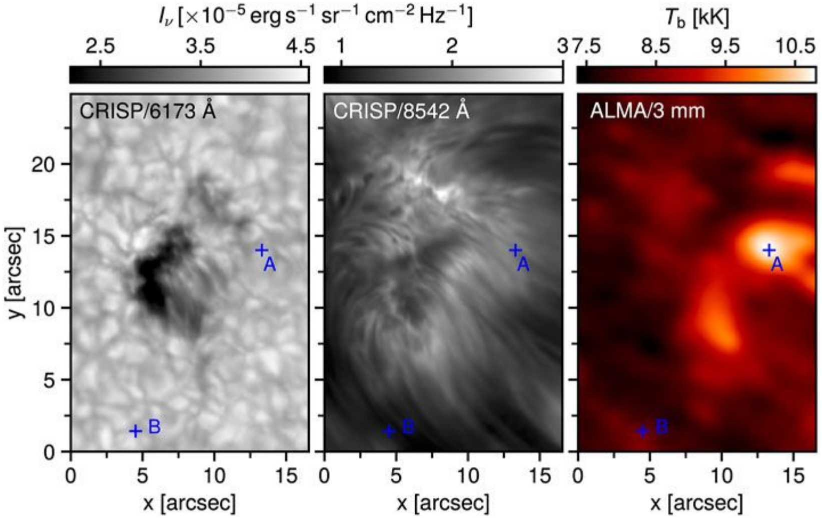
Rosseland Centre for Solar Physics

Modeling solar observations



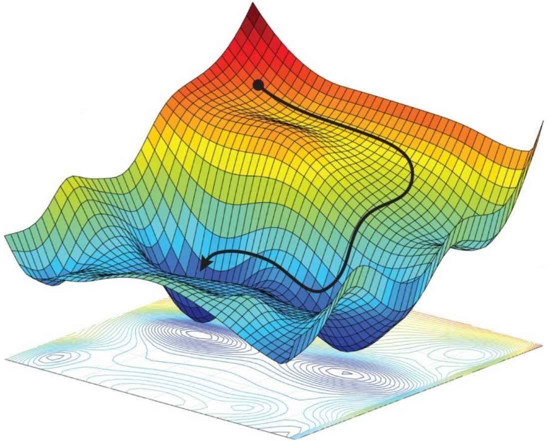
Young et al. (2015)





da Silva Santos et al. (2022)

Gradient-based optimization



source: https://imanahmadianfar.com/codes/

$$\boldsymbol{x}^* = \arg\ \min_{\boldsymbol{x}} \left[||f(\boldsymbol{x}) - \boldsymbol{y}||_2^2 + \lambda\ g(\boldsymbol{x}) \right]$$

Deep Learning frameworks

TensorFlow | Caffe | Microsoft CNTK | Chainer

Keras | theano | DL4J | fast.ai

PyTorch | APACHE mxnet | Caffe2 | ...

Automatic differentiation

AD exploits the chain rule to obtain an fast accurate derivative:

1.- <u>AD vs SD:</u> As accurate as symbolic/manual differentiation but AD can handle complex control flow: conditionals, loops, recursion, etc. (not prone to human errors).

2.- <u>AD vs ND:</u> AD is faster than ND (with a higher memory cost), which could be prone to rounding/truncation errors.

What do they offer?

⇒ You can impose any constraint that you want very easily (e.g. *B>0*):

HARD:

$$B_{ext} = \exp(B_{in})$$

SOFT:

$$\ldots + \lambda\left[\max(-B_{ext}, 0)\right]$$

⇒ We can easily choose which parameters should be free:

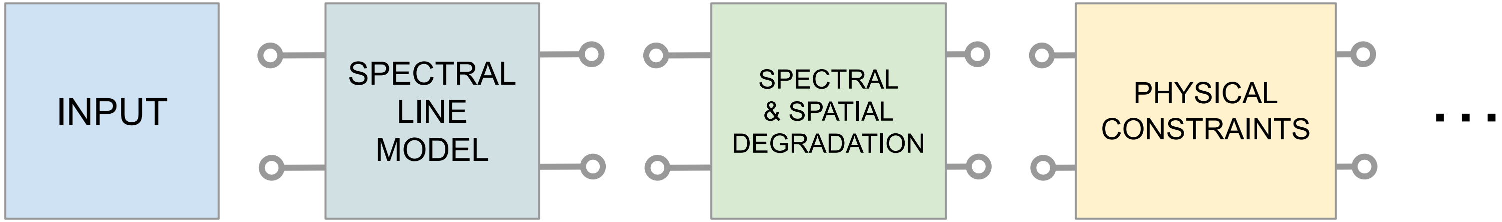
```
A.requires_grad = True
```
**

⇒ Computations can be accelerated on the GPU with minimal changes:

```
A.cuda()
```
**

** PyTorch syntax

Modularity

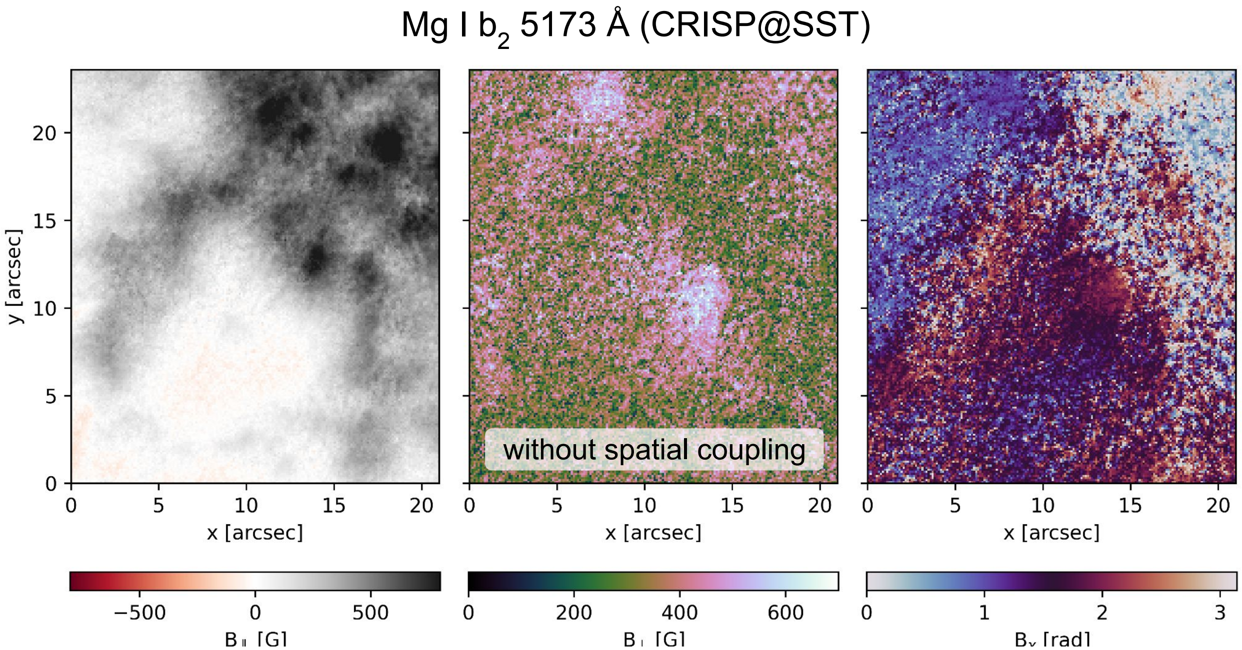


Experiments

- Python + Pytorch for the automatic differentiation (reverse mode by default)

  Result: quick prototyping and analysis

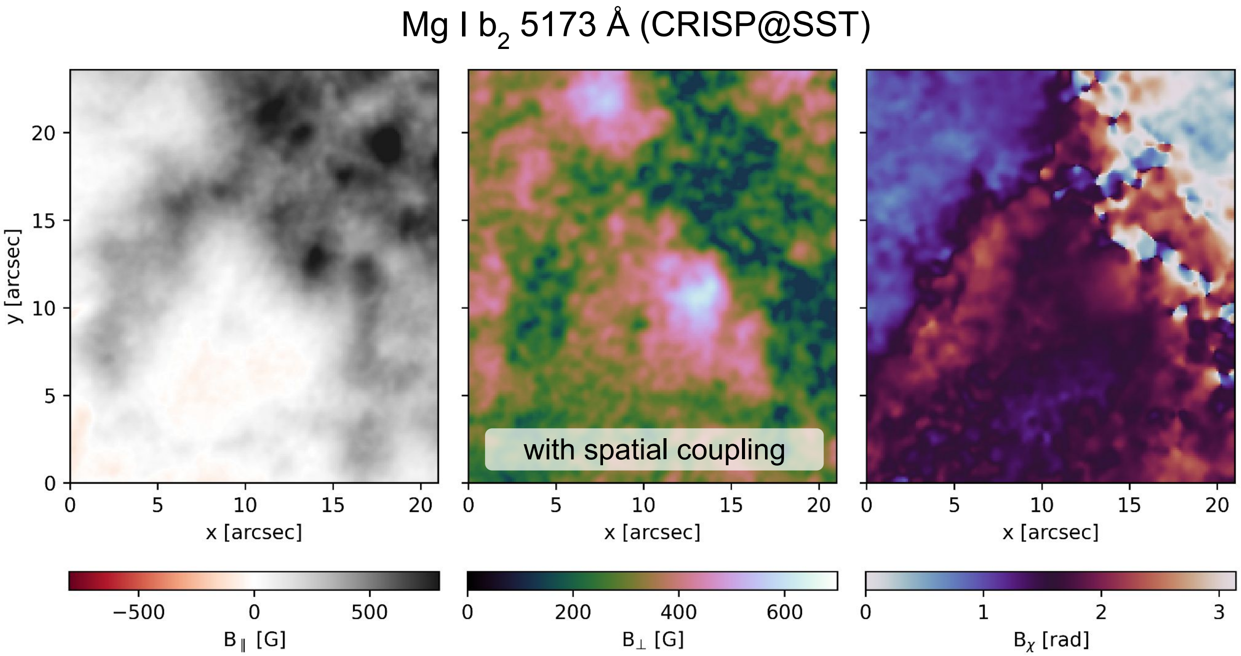
WFA model on spectropolarimetric data



⇒ We expect the solution of one pixel be consistent with the model in the surroundings.

$$\left[||f(x_i) - y_i||_2^2 \right] + \lambda \left[(x_i - x_{up})^2 + (x_i - x_{down})^2 + (x_i - x_{left})^2 + (x_i - x_{right})^2 \right]$$

WFA model on spectropolarimetric data



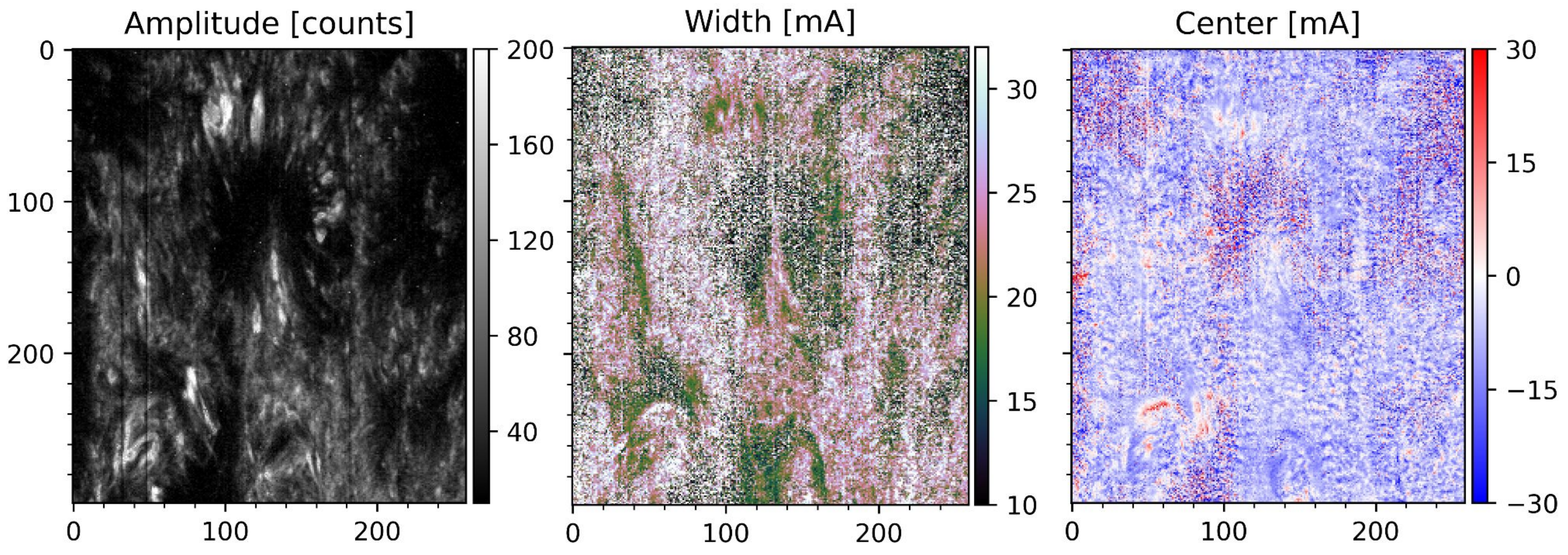
⇒ Penalizing strong spatial gradients if there is not information in the spectra that indicates that.

⇒ Here we do not couple all the pixels in a big matrix but every pixel in a independent way

de la Cruz Rodríguez et al. (2019), Morosin et al. (2020)

Gaussian model on spectroscopic data (IRIS)

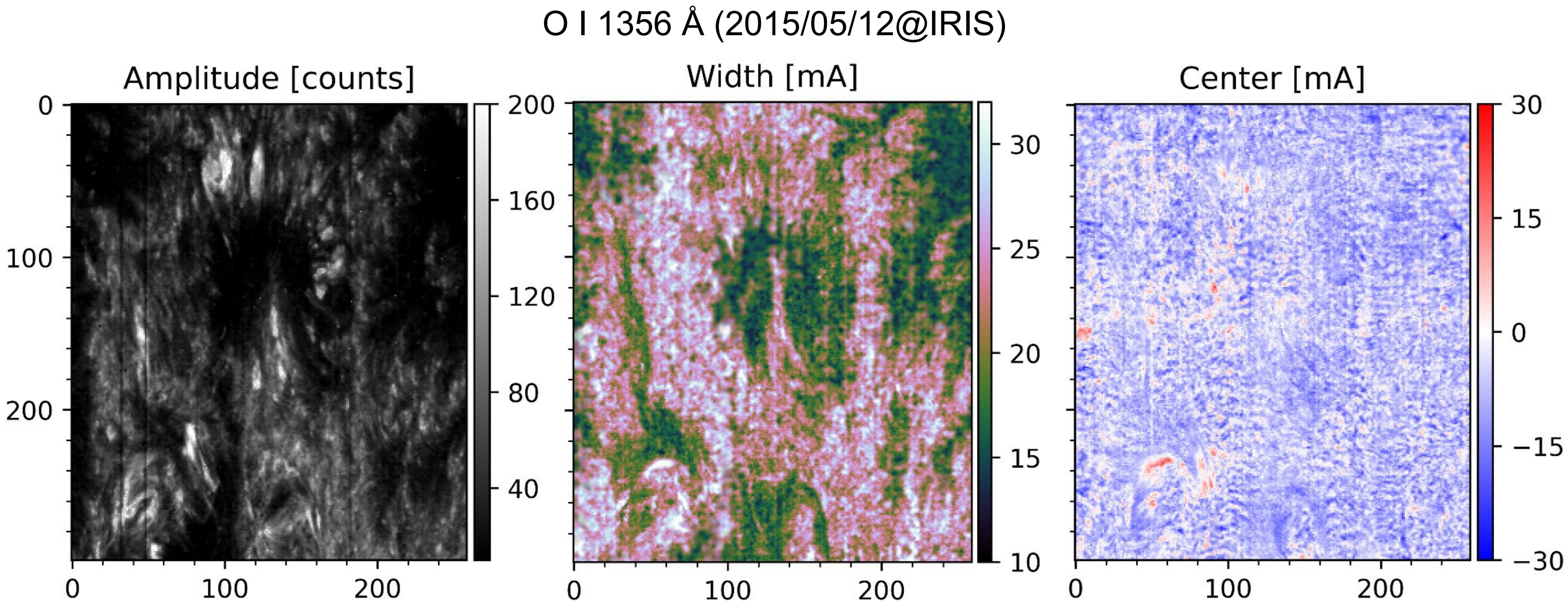
O I 1356 Å (2015/05/12@IRIS)



⇒ A non-linear problem: fitting many pixels. Again, weak lines are more affected by noise.

⇒ **Strengths**: simple parallelization by default as each pixel is treated independently.

Gaussian model on spectroscopic data (IRIS)

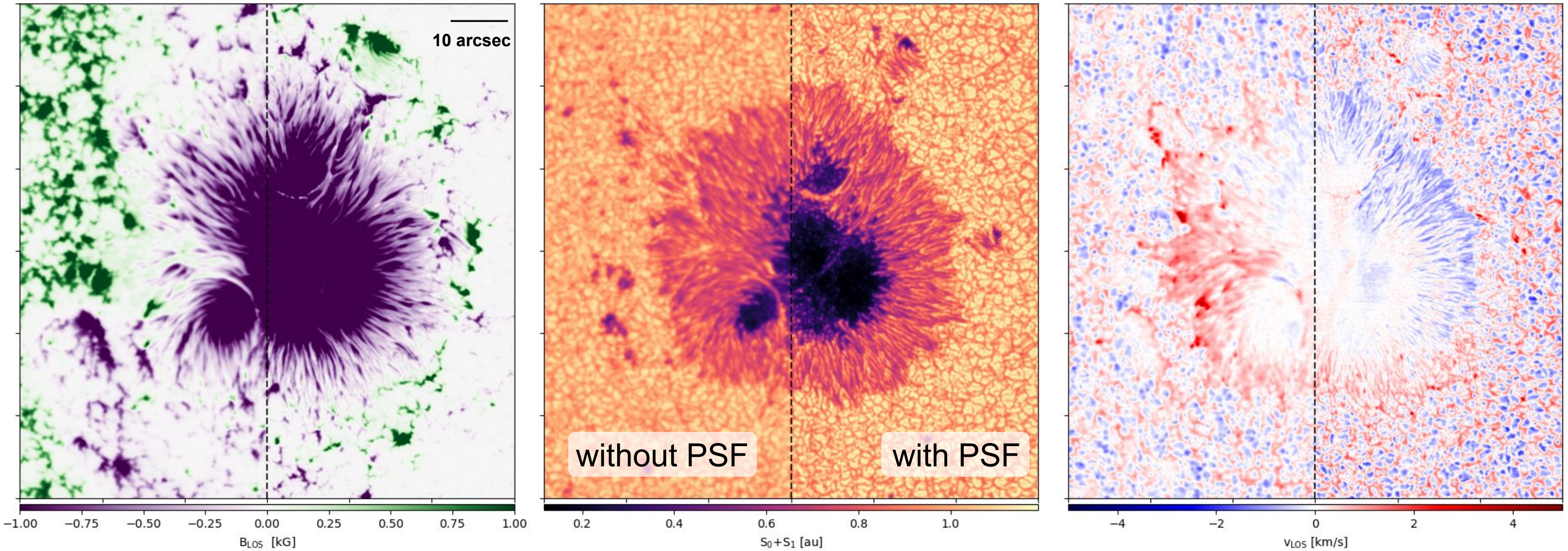


⇒ We added [spatial regularization] + [width > minimum_value]

⇒ It helps to provide a more coherent map but we should use uncertainty information to trust regions.

ME model + PSF in Hinode Data

Hinode/SP 2007/04/30 (NOAA 10953)

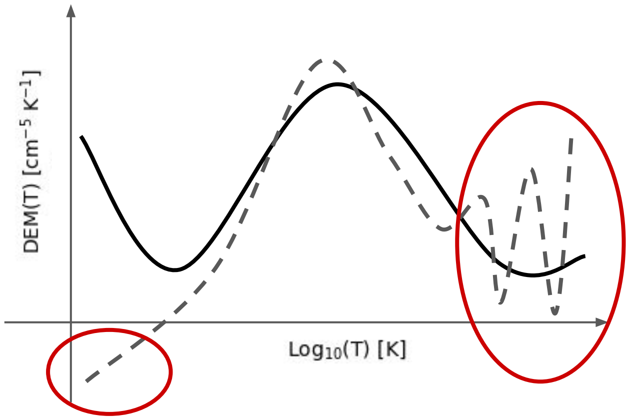


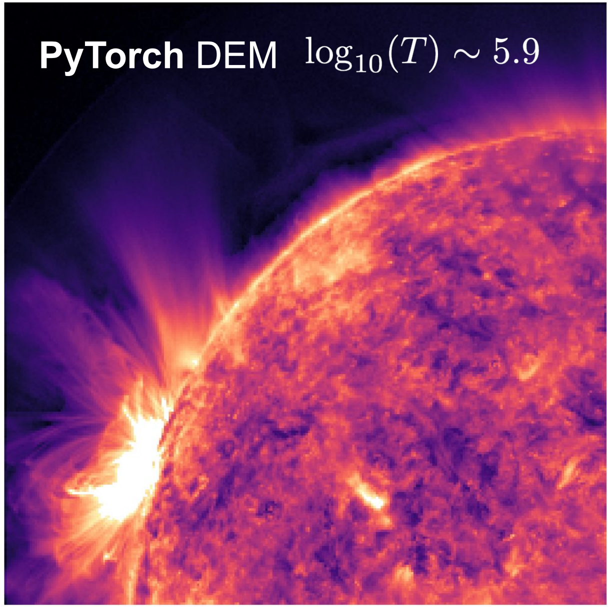
⇒ We can exchange the modules like lego pieces, now including the PSF of the telescope

⇒ The model is more computationally expensive, so the ME can be run on GPUs or written in C++

Differential Emission Measure inversions

$$I_\lambda = \int_0^\infty R_\lambda(T) \cdot \mathrm{DEM}(T)\, dT$$





⇒ **Strengths**: You can play with different penalty terms (DEM>0, **temporal coherence**, etc)

⇒ Note: current methods are **highly** optimized to perform much faster

Hannah & Kontar (2012), Plowman et al. (2012), Cheung et al. (2015), Warren et al. (2017), Massa et al. (2023)

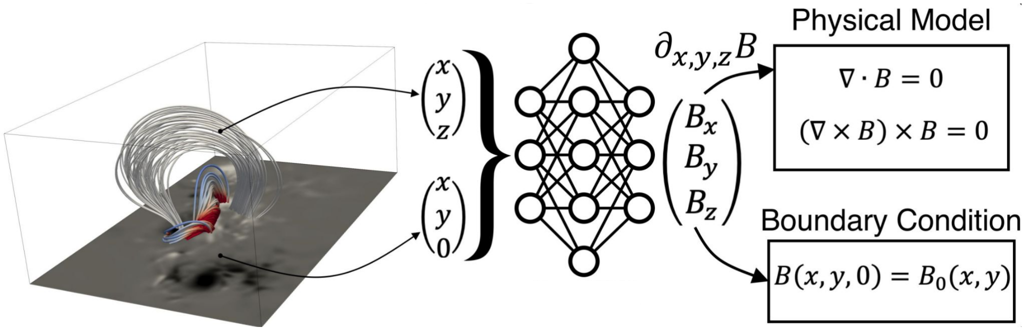
Implicit Neural Representation

⇒ Goal: Can we find a better way of parametrizing the data than using pixels?

$$I_{x,y} = f_\theta(x, y)$$

Asensio Ramos & de la Cruz Rodríguez (2015)

⇒ Goal: Can we use a neural network to describe our parameters?



Jarolim et al. (2022)

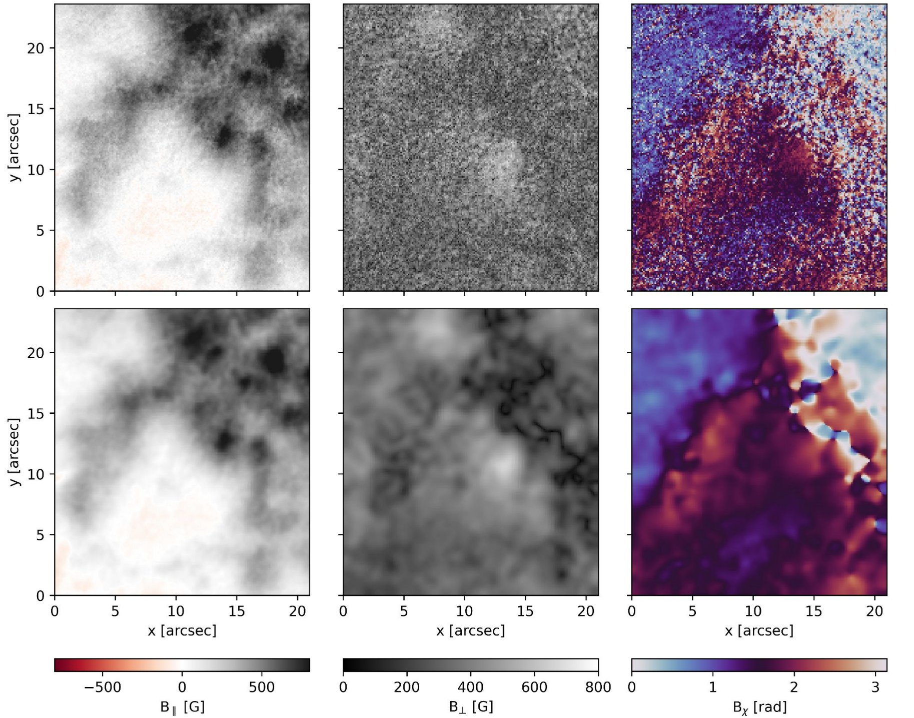
Mesh-free "Physics-informed Neural Networks"

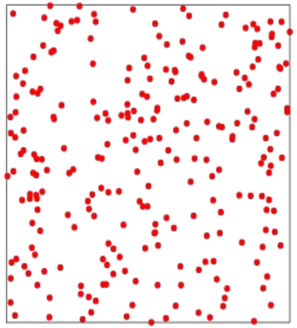
Mg I b_2 5173 Å (CRISP@SST)

Pixel-by-pixel

Neural network

Params (10% of $n_x n_y$)





Grid evaluation

⇒ **Strengths**: Continuous approximation of the parameters in the whole domain

Summary and conclusions

- **Versatility**: you can test different ideas and regularizations
- **Accuracy**: efficient gradient calculations with a simple interface
- **Modularity**: It can combine modules written in Numba/Fortran/C++
- **Speed**: run it in GPUs without almost modification
- **WIP and trade-off**: many things to explore but promising possibilities

Asensio Ramos & de la Cruz Rodríguez (2015), de la Cruz Rodríguez et al. (2019), Morosin et al. (2020), Jirí Štepán et al. (2022)