

# NUMERICAL INFORMATION FIELD THEORY - TUTORIAL

## BAYESIAN IMAGING USING IFT

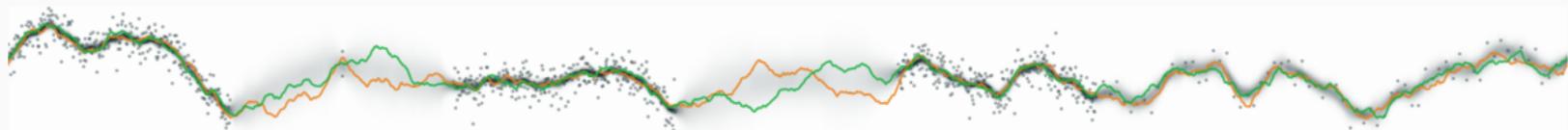
---

Philipp Frank<sup>1</sup> and Torsten Enßlin<sup>1,2</sup>

Black Hole Initiative - Harvard University, November 8, 2022

(1) Max-Planck Institute for Astrophysics, Garching, Germany

(2) Ludwig-Maximilians Universität LMU, Munich, Germany



# OVERVIEW

Coding tutorial: [https://gitlab.mpcdf.mpg.de/ift/tutorial\\_nifty\\_resolve](https://gitlab.mpcdf.mpg.de/ift/tutorial_nifty_resolve)

NIFTy [ABE<sup>+</sup>19]:

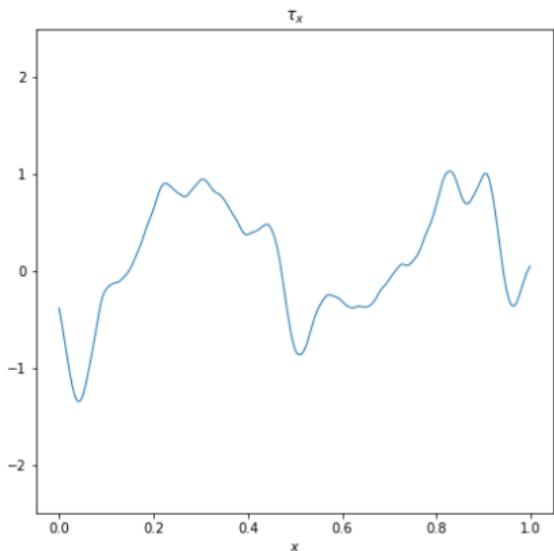
- python library for statistical inference
- differentiable generative models
- flexible gaussian processes (correlated field model)
- variety of observational likelihoods
- variational inference



## SPACES

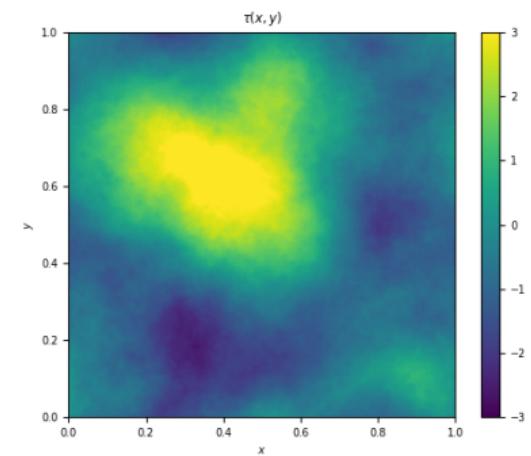
---

# NIFTY - SPACES



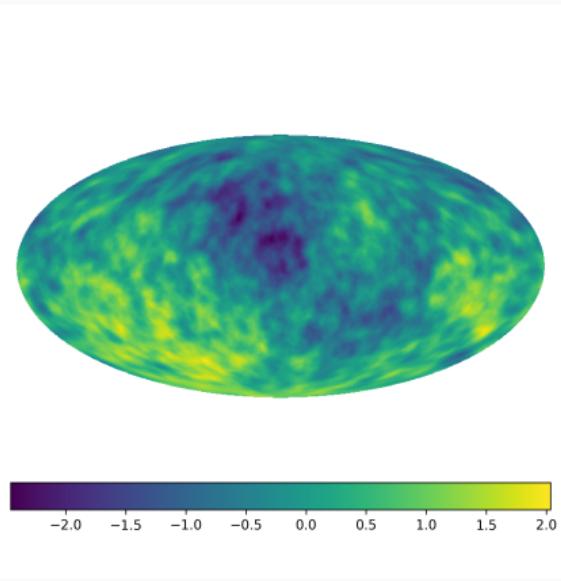
```
1 import nifty8 as ift
2
3 # 1-dimensional regular grid space
4 # with 128 pixels and pixelsize 1/128
5 space = ift.RGSpace(128, 1/128)
6
7 ...
```

# NIFTY - SPACES



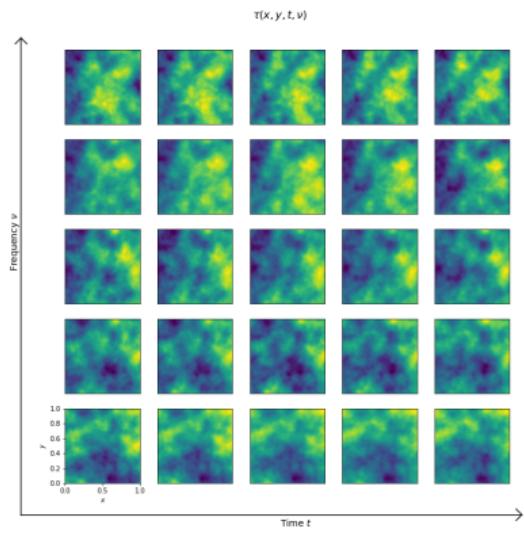
```
1 import nifty8 as ift  
2  
3 # 2-dimensional regular grid space  
4 # with 128x128 pixels and pixelsizes 1/128  
5 space = ift.RGSpace((128, 128), (1/128, 1/128))  
6  
7 * * *
```

# NIFTY - SPACES



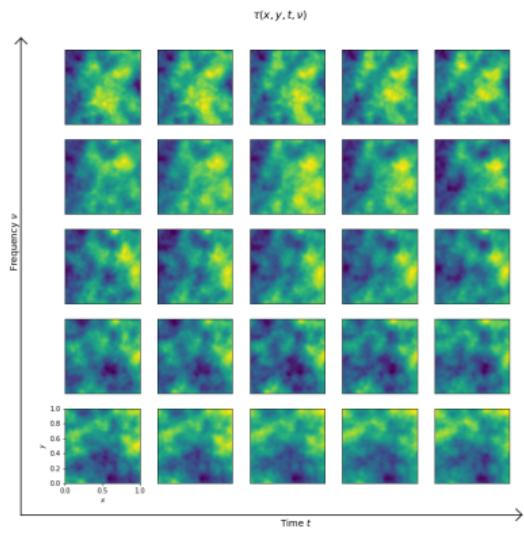
```
1 import nifty8 as ift  
2  
3 # 2-dimensional spherical (HEALPiX) space  
4 # with nside 128  
5 space = ift.HPSpace(128)  
6  
7 ...
```

# NIFTY - SPACES



```
1 import nifty8 as ift
2
3 # 2-dimensional regular grid space
4 # with 128x128 pixels
5 image_dom = ift.RGSpace((128, 128))
6
7 # frequency & time domain with
8 # 5 regularly spaced pixels
9 time = ift.RGSpace(5)
10 freq = ift.RGSpace(5)
11
12
13
14
15 ...
```

# NIFTY - SPACES

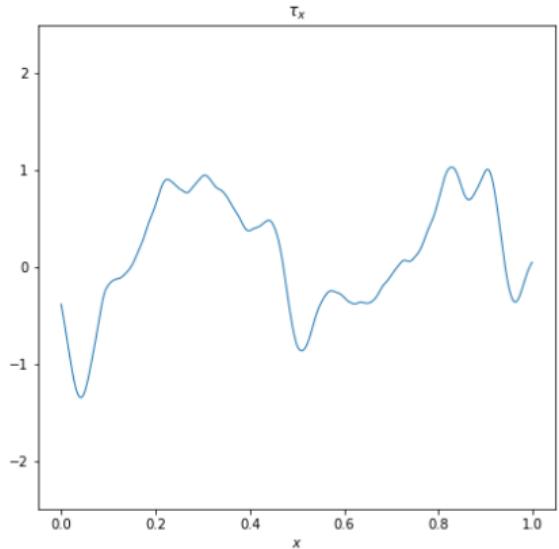


```
1 import nifty8 as ift
2
3 # 2-dimensional regular grid space
4 # with 128x128 pixels
5 image_dom = ift.RGSpace((128, 128))
6
7 # frequency & time domain with
8 # 5 regularly spaced pixels
9 time = ift.RGSpace(5)
10 freq = ift.RGSpace(5)
11
12 # Set up joint space
13 space = ift.makeDomain((freq, time, image_dom))
14
15 ...
```

## CORRELATEDFIELDS

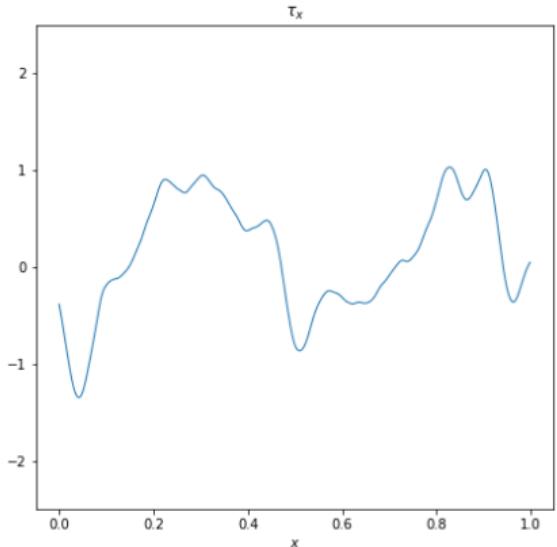
---

# NIFTY - CORRELATEDFIELDS [AFH<sup>+</sup>20]



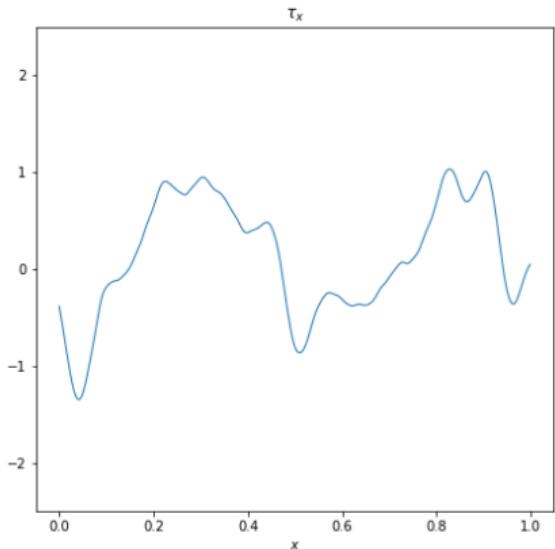
```
1 import nifty8 as ift
2
3 # 1-dimensional regular grid space
4 # with 128 pixels and pixelsize 1/128
5 space = ift.RGSpace(128, 1/128)
6
7
8
9
10
11
12
13
14
15
16
17 ... # Plot 'tau'
```

# NIFTY - CORRELATEDFIELDS [AFH<sup>+</sup>20]



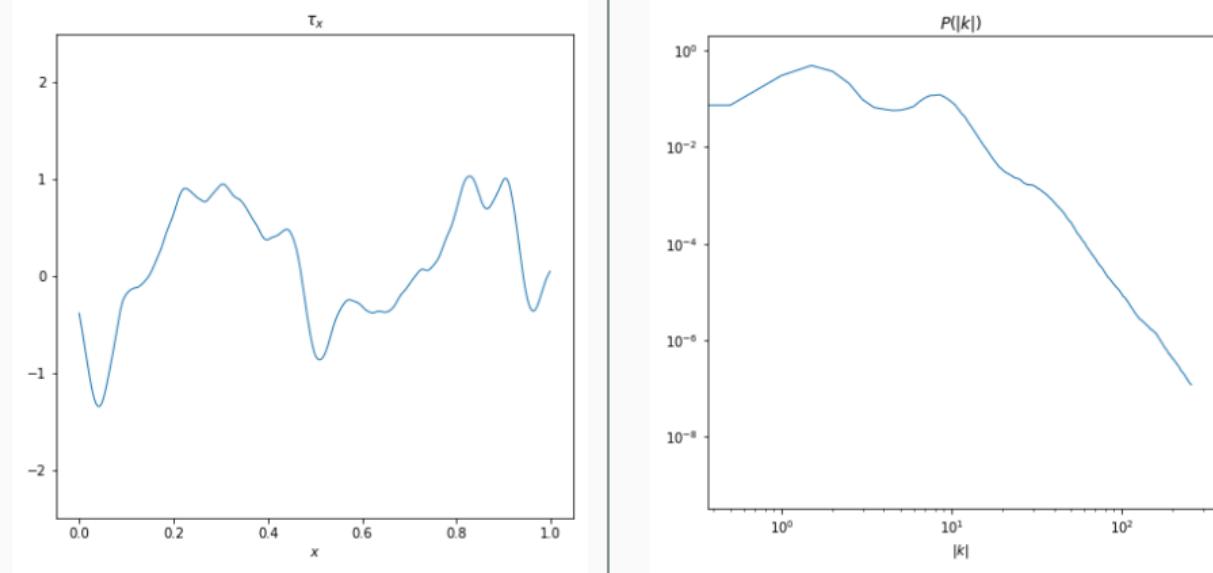
```
1 import nifty8 as ift
2
3 # 1-dimensional regular grid space
4 # with 128 pixels and pixelsize 1/128
5 space = ift.RGSpace(128, 1/128)
6
7 # Define a Gaussian random processes on 'space'
8 args = {...} # Hyperparameters for GP model
9 model = ift.SimpleCorrelatedField(space, **args)
10
11
12
13
14
15
16
17 ... # Plot 'tau'
```

# NIFTY - CORRELATEDFIELDS [AFH<sup>+</sup>20]

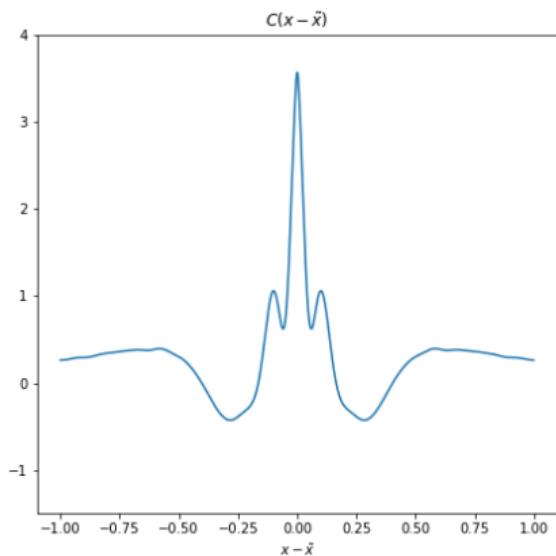
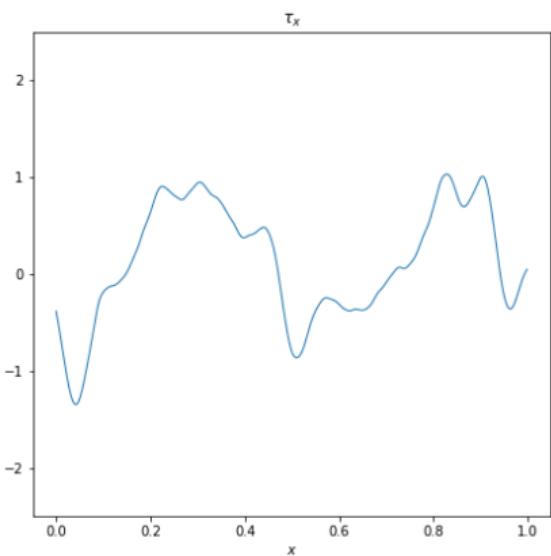


```
1 import nifty8 as ift
2
3 # 1-dimensional regular grid space
4 # with 128 pixels and pixelsize 1/128
5 space = ift.RGSpace(128, 1/128)
6
7 # Define a Gaussian random processes on 'space'
8 args = {...} # Hyperparameters for GP model
9 model = ift.SimpleCorrelatedField(space, **args)
10
11 # Draw a random realization of standard normal
12 # distributed variables
13 realization = ift.from_random(model.domain)
14 # Apply model to get a realization
15 tau = model(realization)
16
17 ... # Plot 'tau'
```

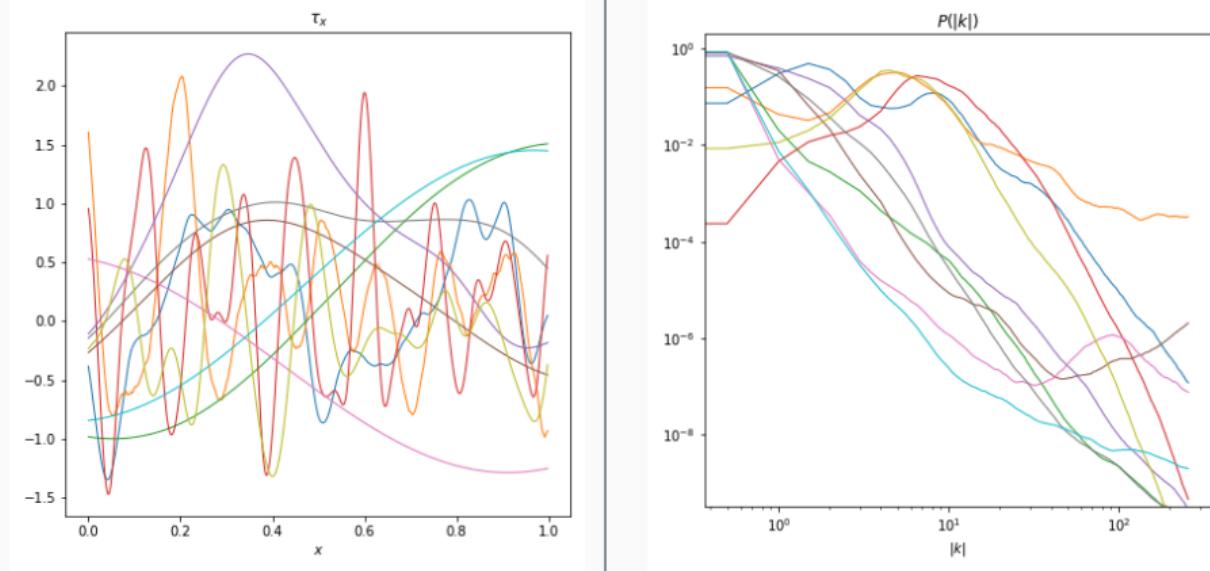
# NIFTY - CORRELATEDFIELDS [AFH<sup>+</sup>20]



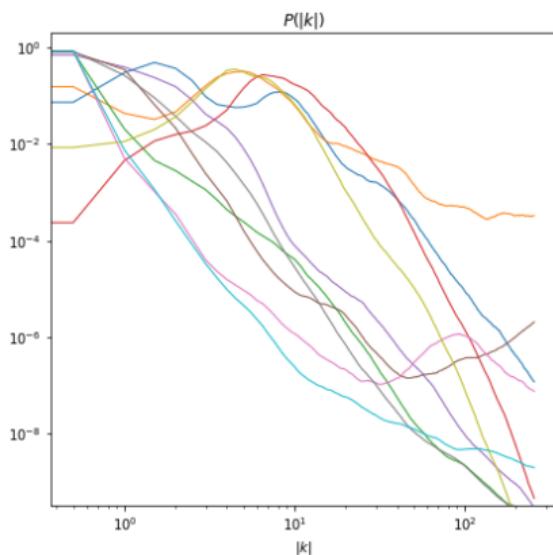
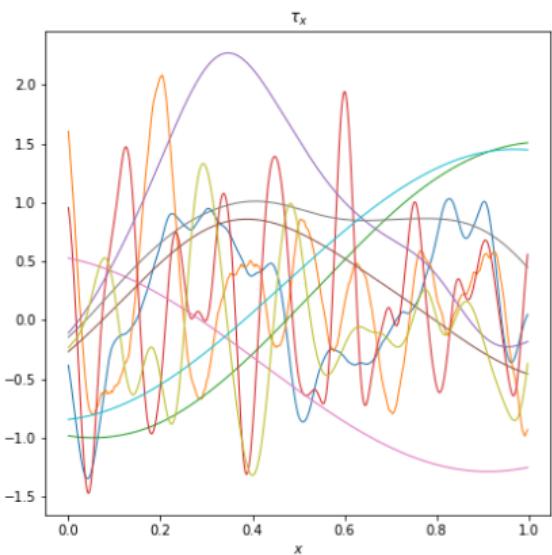
# NIFTY - CORRELATEDFIELDS [AFH<sup>+</sup>20]



# NIFTY - CORRELATEDFIELDS [AFH<sup>+</sup>20]

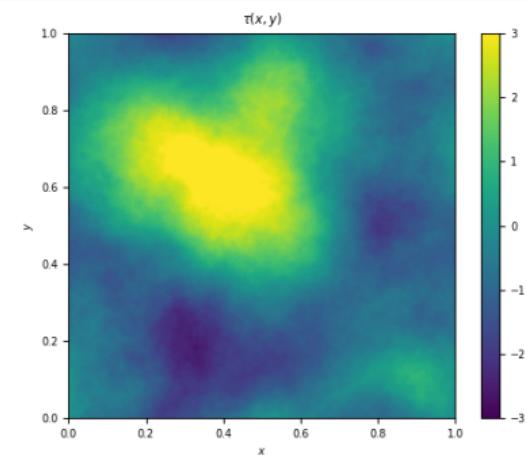


# NIFTY - CORRELATEDFIELDS [AFH<sup>+</sup>20]



⇒ See the `demo_CorrelatedFields.ipynb` notebook for further information!

# NIFTY - CORRELATEDFIELDS [AFH<sup>+</sup>20]

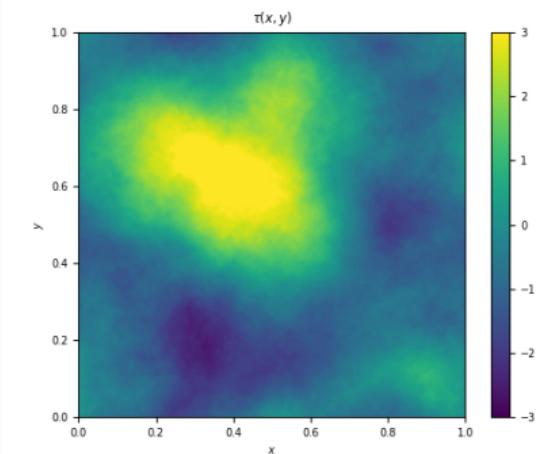


```
1 import nifty8 as ift
2
3 # 2-dimensional regular grid space
4 # with 128x128 pixels and pixelsizes 1/128
5 space = ift.RGSpace((128, 128), (1/128, 1/128))
6
7 # Define a Gaussian random processes on 'space'
8 args = {...} # Hyperparameters for GP model
9 model = ift.SimpleCorrelatedField(space, **args)
10
11 # Draw a random realization of standard normal
12 # distributed variables
13 realization = ift.from_random(model.domain)
14 # Apply model to get a model realization
15 tau = model(realization)
16
17 ... # Plot 'tau'
```

## DATA MODEL & VARIATIONAL INFERENCE

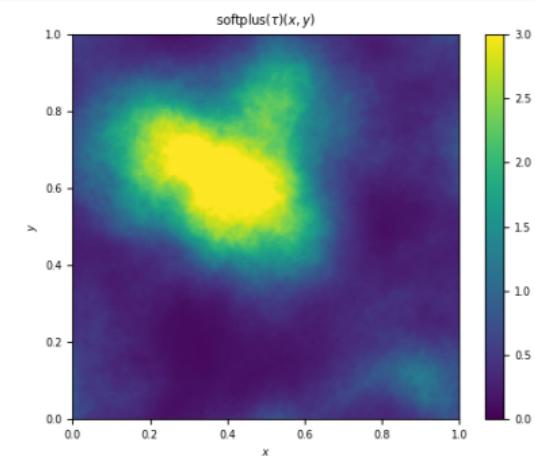
---

# NIFTY - VARIATIONAL INFERENCE [FLE21]



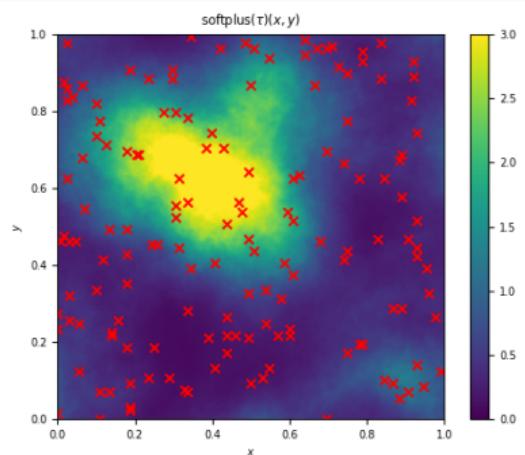
```
1 model = ... # model for tau  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17 * * *
```

# NIFTY - VARIATIONAL INFERENCE [FLE21]



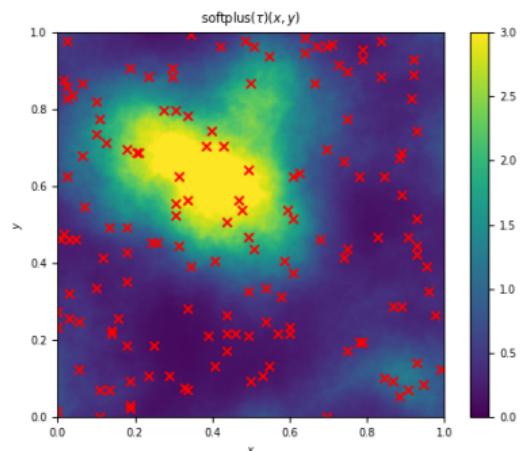
```
1 model = ift.softplus(model) # apply nonlinearity
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17 . . .
```

# NIFTY - VARIATIONAL INFERENCE [FLE21]



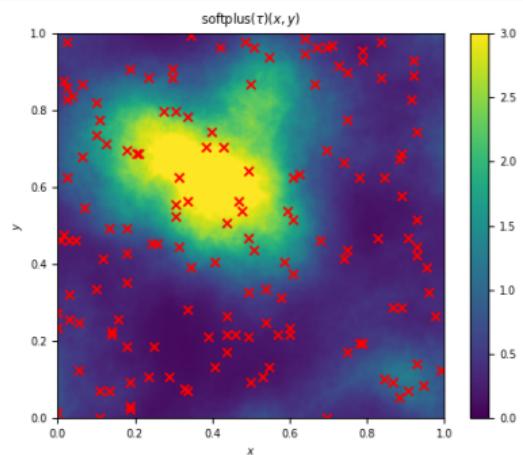
```
1 model = ift.softplus(model) # apply nonlinearity
2
3 # Set up a random response
4 flags = np.random.binomial(1, 0.99, size = 128**2)
5 flags = ift.makeField(space, flags)
6 Response = ift.MaskOperator(flags)
7
8
9
10
11
12
13
14
15
16
17 ...
```

# NIFTY - VARIATIONAL INFERENCE [FLE21]



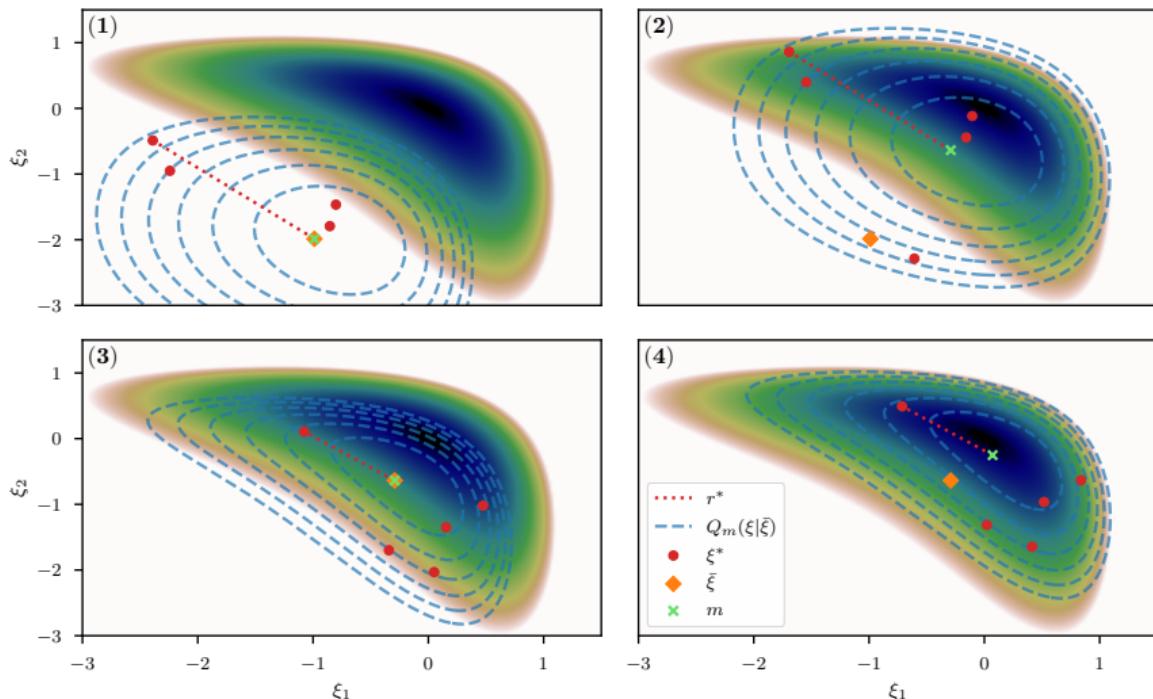
```
1 model = ift.softplus(model) # apply nonlinearity
2
3 # Set up a random response
4 flags = np.random.binomial(1, 0.99, size = 128**2)
5 flags = ift.makeField(space, flags)
6 Response = ift.MaskOperator(flags)
7
8 # Define observational model and likelihood
9 data, noise_icov = # load data and noise
10 lh = ift.GaussianEnergy(data = data,
11                         inverse_covariance = noise_icov)
12 likelihood = lh @ Response(model)
13
14
15
16
17 ...
```

# NIFTY - VARIATIONAL INFERENCE [FLE21]

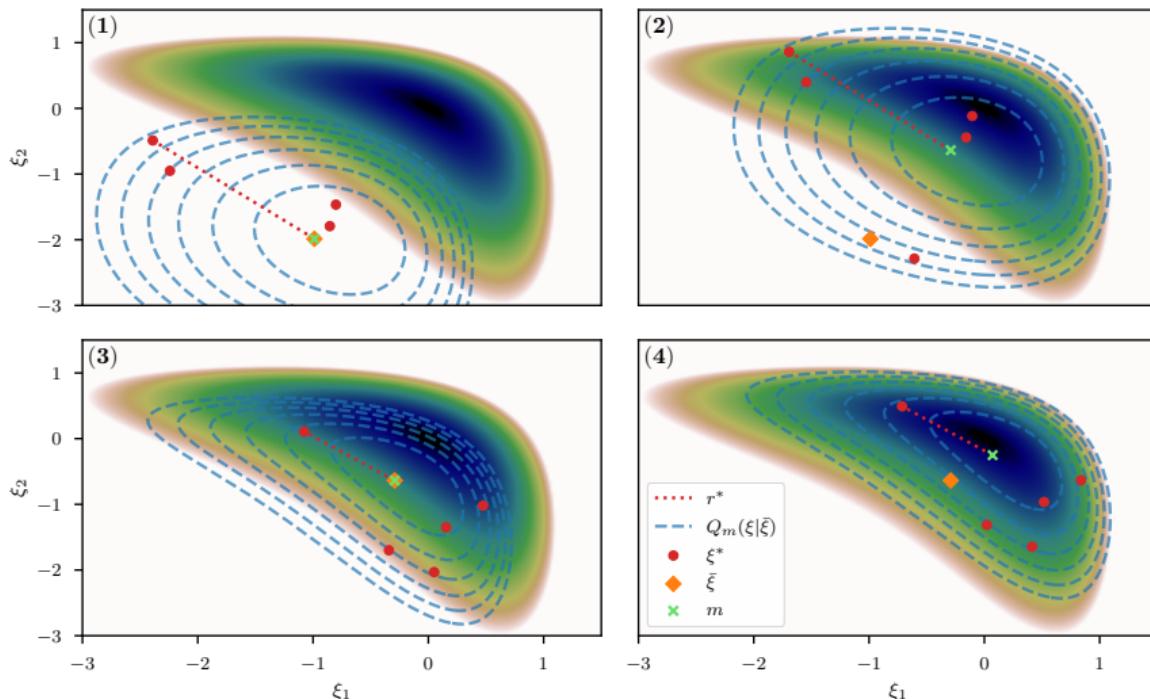


```
1 model = ift.softplus(model) # apply nonlinearity
2
3 # Set up a random response
4 flags = np.random.binomial(1, 0.99, size = 128**2)
5 flags = ift.makeField(space, flags)
6 Response = ift.MaskOperator(flags)
7
8 # Define observational model and likelihood
9 data, noise_icov = # load data and noise
10 lh = ift.GaussianEnergy(data = data,
11                         inverse_covariance = noise_icov)
12 likelihood = lh @ Response(model)
13
14 # Generate approximate posterior samples
15 # using variational inference (geoVI)
16 samples = ift.optimize_kl(lh, **params)
17 ...
```

# NIFTY - VARIATIONAL INFERENCE [FLE21]

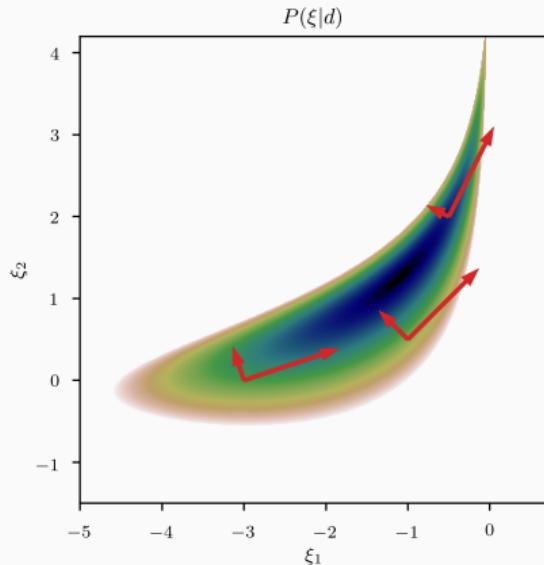


# NIFTY - VARIATIONAL INFERENCE [FLE21]



→ not 2- but (# of pixels + # spectrum parameters)-dimensional probability distributions!

# GEOMETRIC VARIATIONAL INFERENCE (GEOVI) [?]

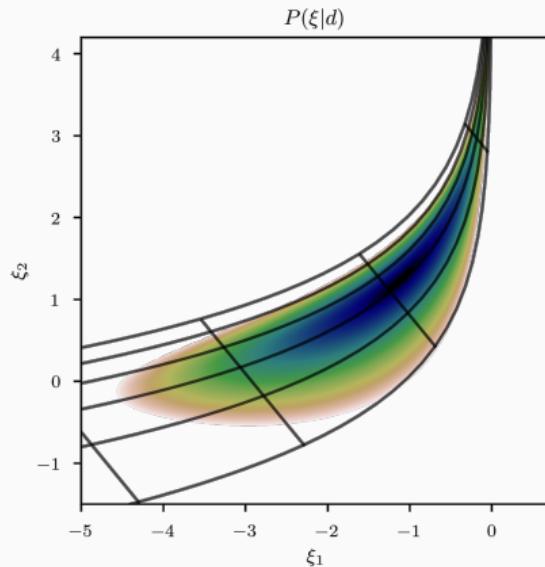


Information Hamiltonian  $\mathcal{H}(\xi|d)$ :  $-\log (\mathcal{P}(\xi|d))$

Posterior metric  $\mathcal{M}(\xi)$ :  $\mathcal{M}_{\text{lh}}(\xi) + \mathbb{1}$

Fisher information metric  $\mathcal{M}_{\text{lh}}(\xi)$ :  $\left\langle \frac{\partial^2 \mathcal{H}(d|\xi)}{\partial \xi \partial \xi'} \right\rangle_{\mathcal{P}(d|\xi)}$

# GEOMETRIC VARIATIONAL INFERENCE (GEOVI) [?]

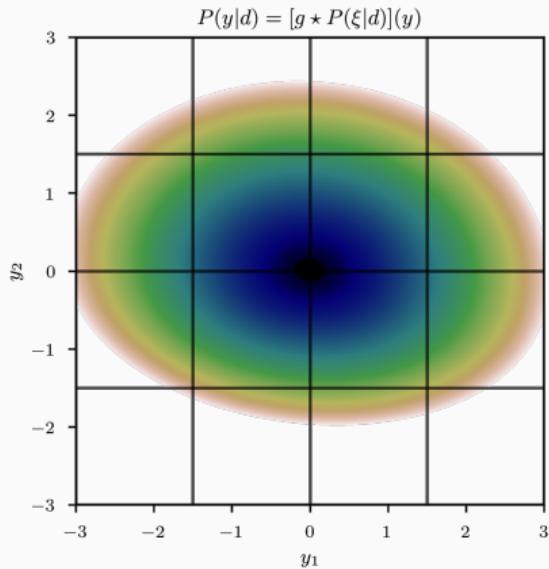
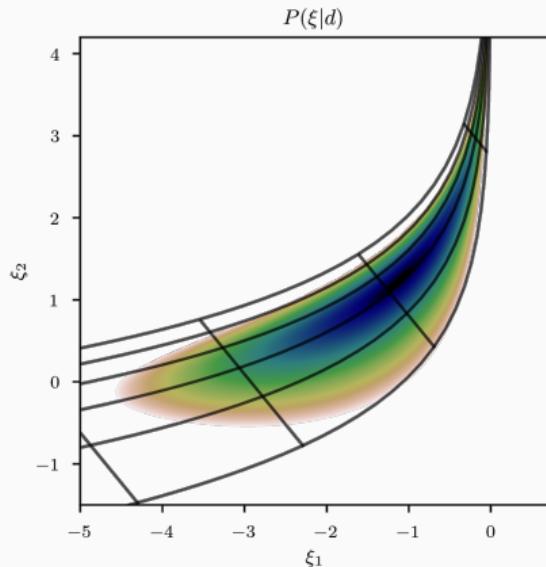


Information Hamiltonian  $\mathcal{H}(\xi|d)$ :  $-\log (\mathcal{P}(\xi|d))$

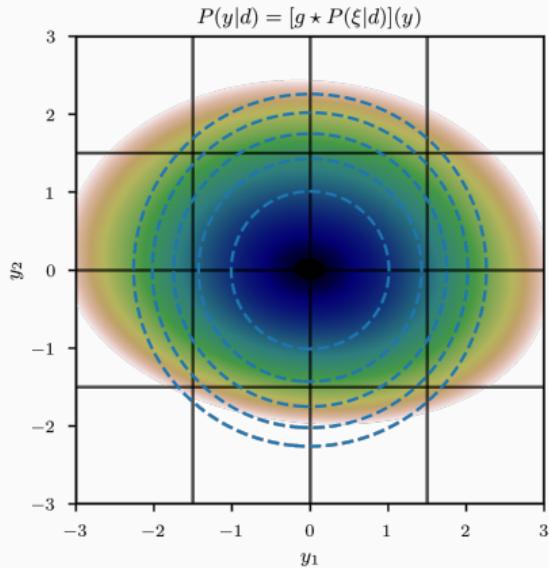
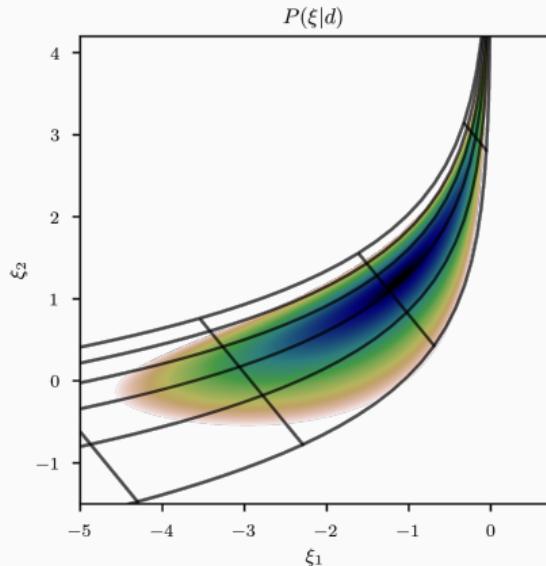
Posterior metric  $\mathcal{M}(\xi)$ :  $\mathcal{M}_{\text{lh}}(\xi) + \mathbb{1}$

Fisher information metric  $\mathcal{M}_{\text{lh}}(\xi)$ :  $\left\langle \frac{\partial^2 \mathcal{H}(d|\xi)}{\partial \xi \partial \xi'} \right\rangle_{\mathcal{P}(d|\xi)}$

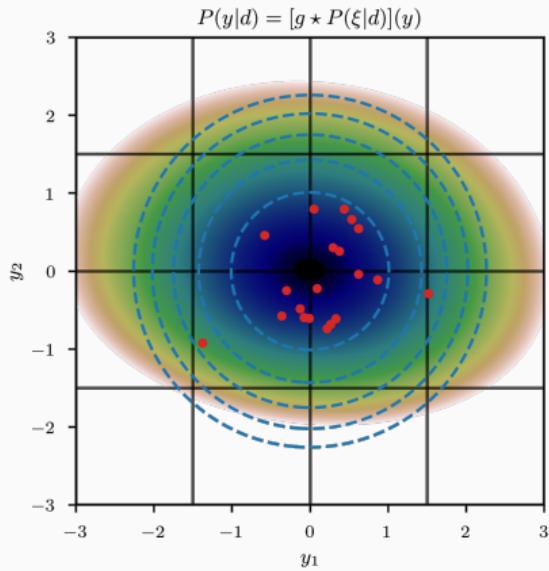
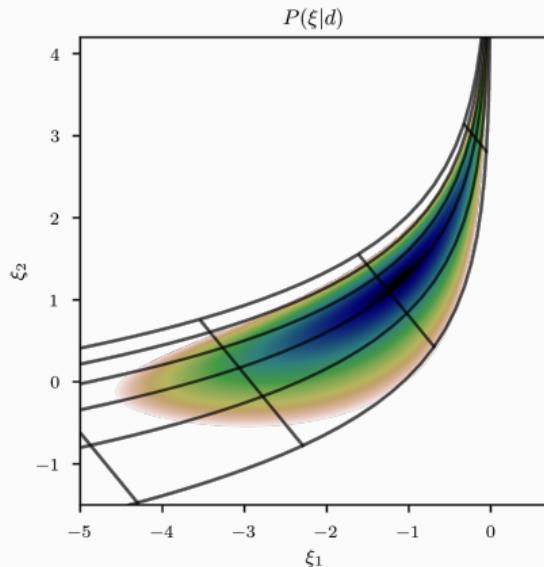
# GEOMETRIC VARIATIONAL INFERENCE (GEOVI) [?]



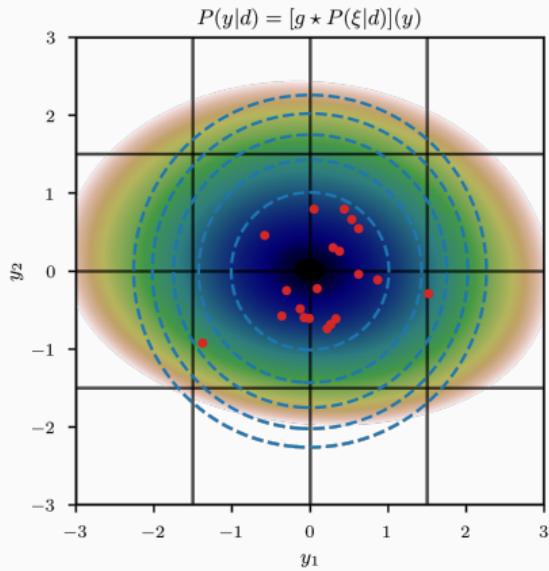
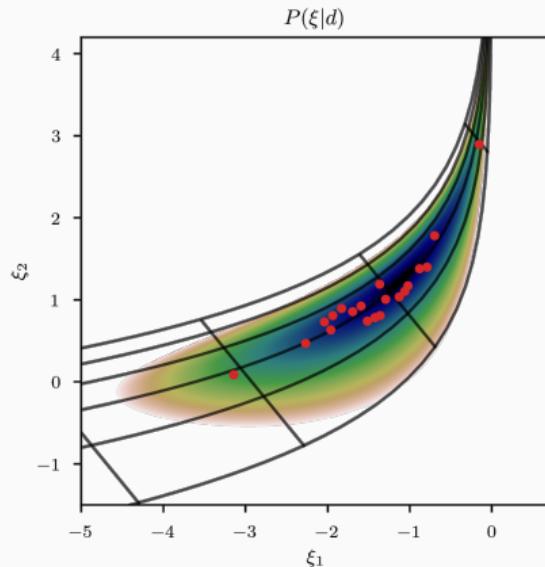
# GEOMETRIC VARIATIONAL INFERENCE (GEOVI) [?]



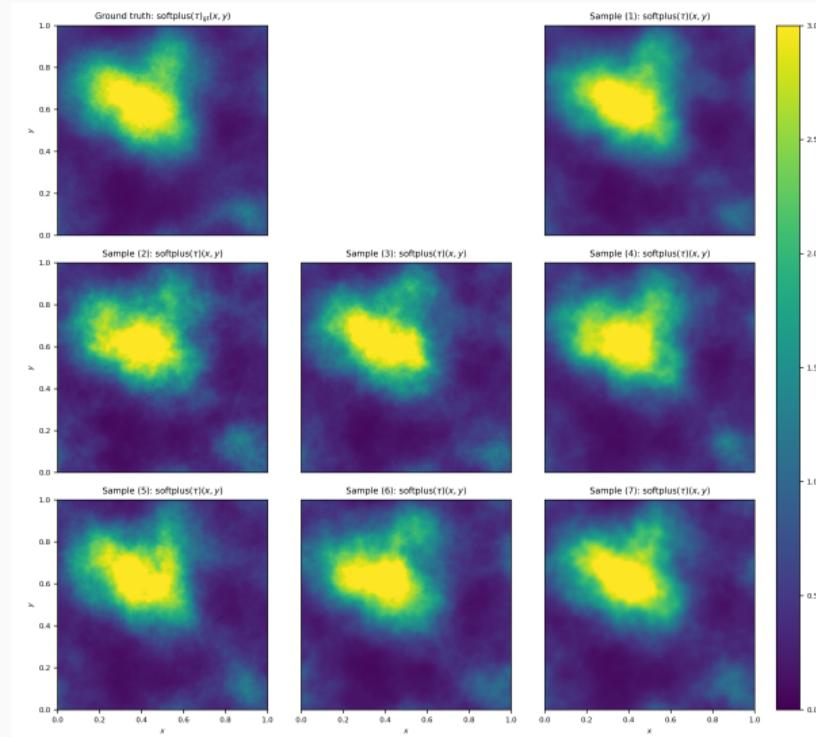
# GEOMETRIC VARIATIONAL INFERENCE (GEOVI) [?]



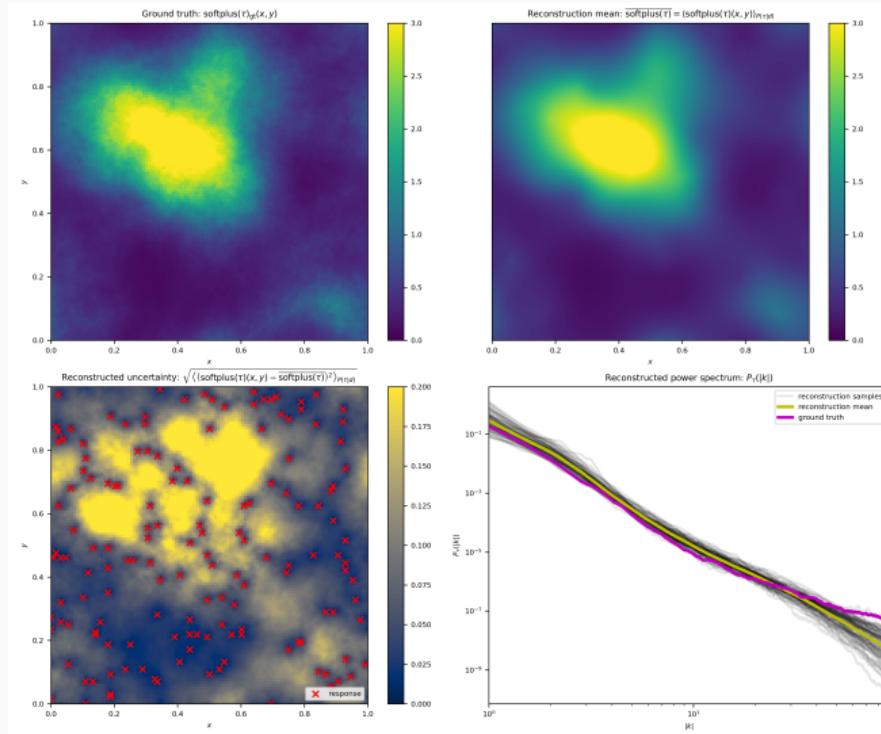
# GEOMETRIC VARIATIONAL INFERENCE (GEOVI) [?]



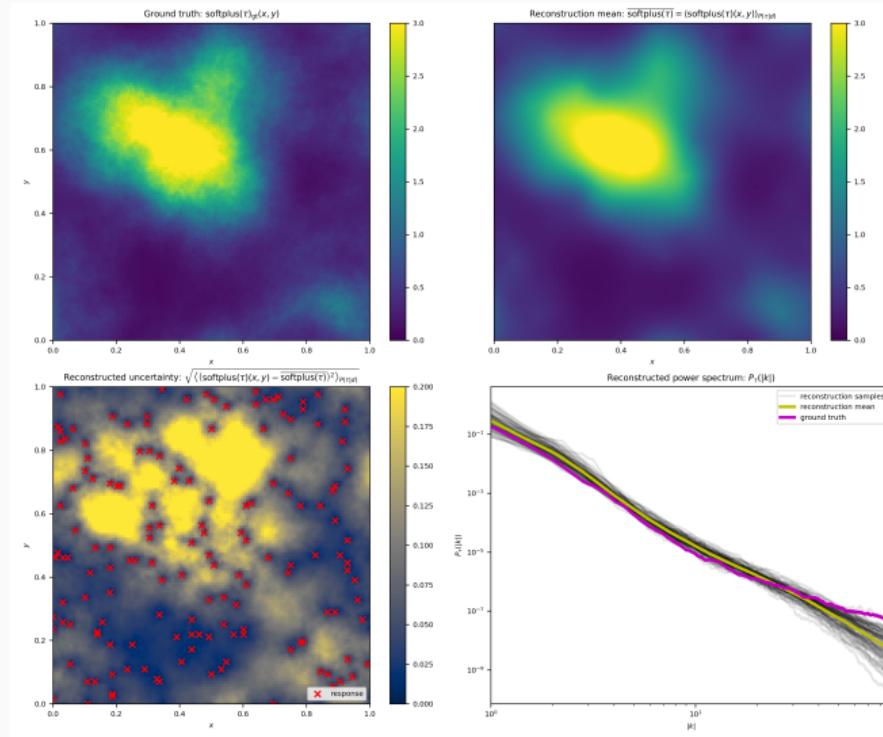
# NIFTY - VARIATIONAL INFERENCE [FLE21]



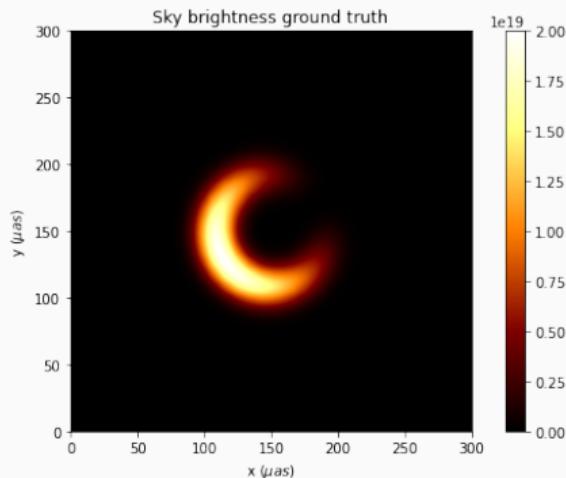
# NIFTY - VARIATIONAL INFERENCE [FLE21]



# NIFTY - VARIATIONAL INFERENCE [FLE21]

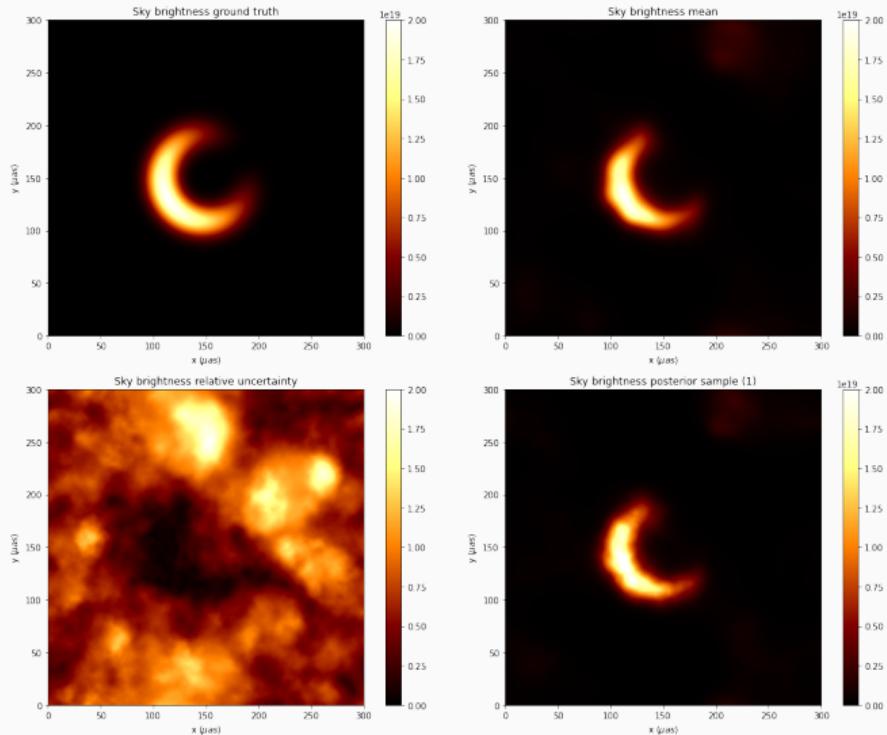


# INTERFEROMETRIC IMAGING

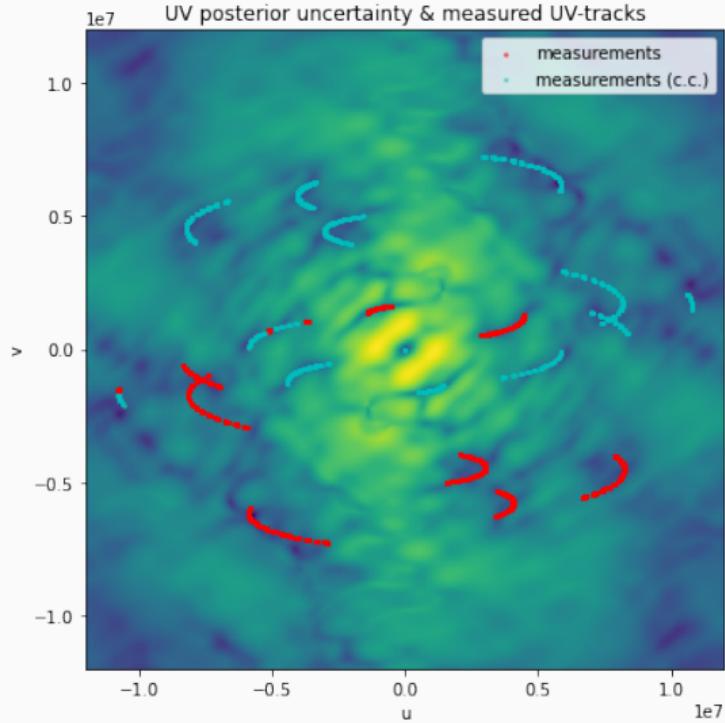
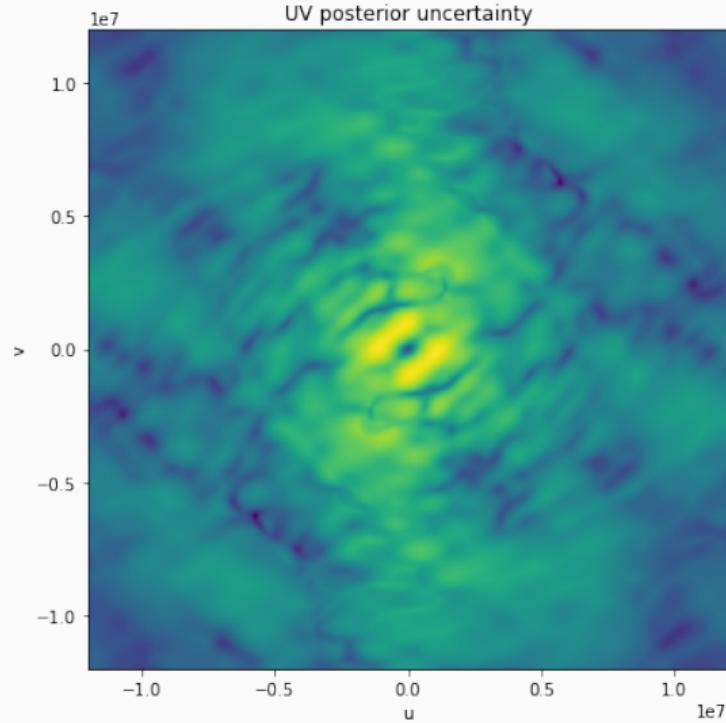


```
1 model = ift.exp(model) # apply nonlinearity
2
3 # Set up a vlbi response
4 import resolve as rve
5 observation = rve.Observation.load('data path...')
6 Response = rve.InterferometryResponse(
7                                     observation, model.target)
8
9 # Define observational model and likelihood
10 data, noise_icov = # load data and noise
11 lh = ift.GaussianEnergy(data = data,
12                           inverse_covariance = noise_icov)
13 likelihood = lh @ Response(model)
14
15
16
17 ...
```

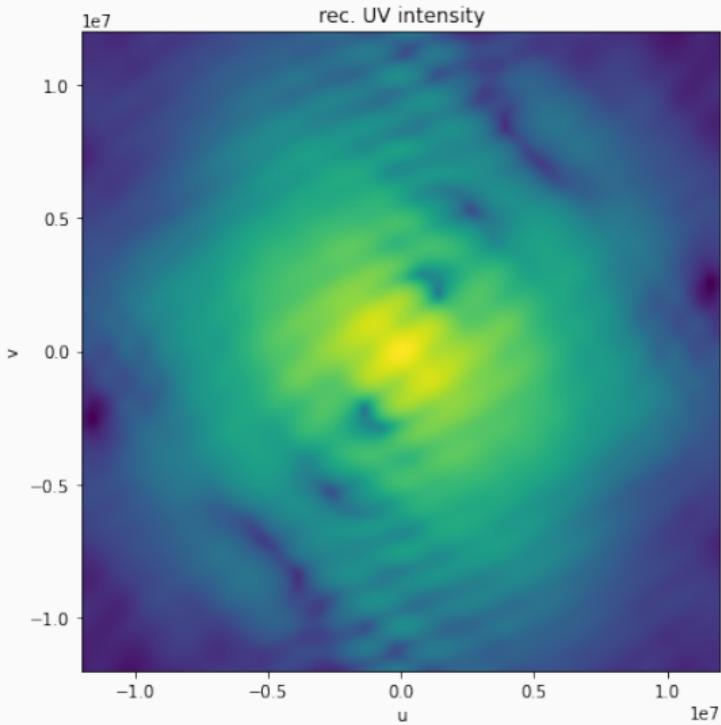
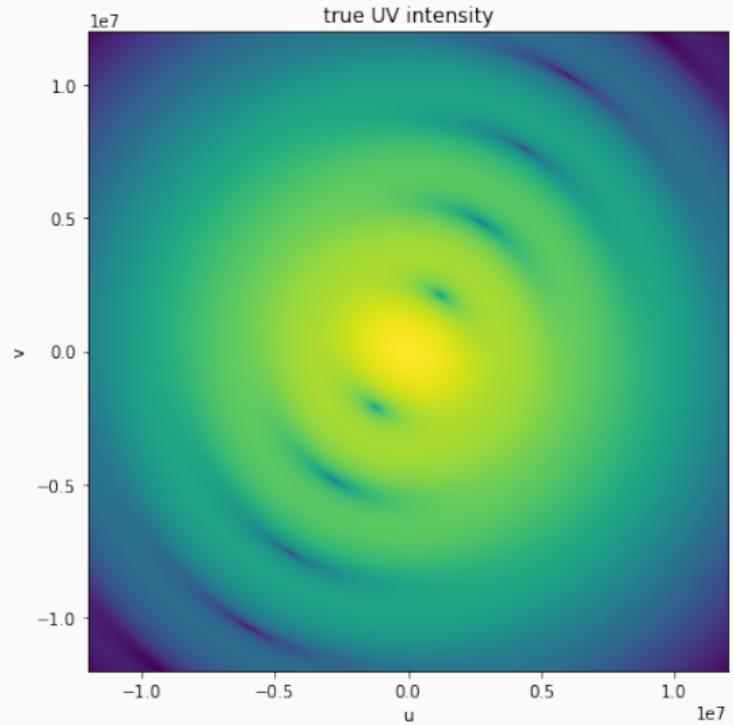
# INTERFEROMETRIC IMAGING



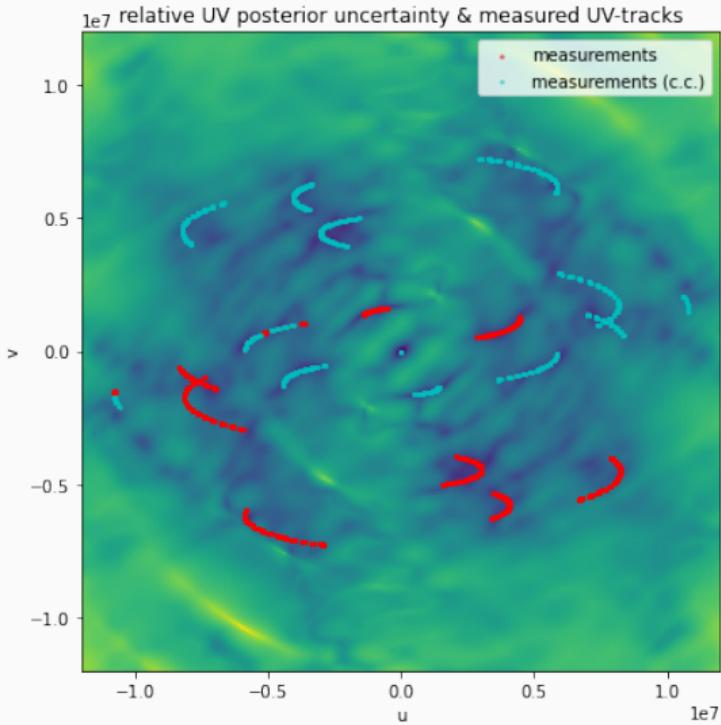
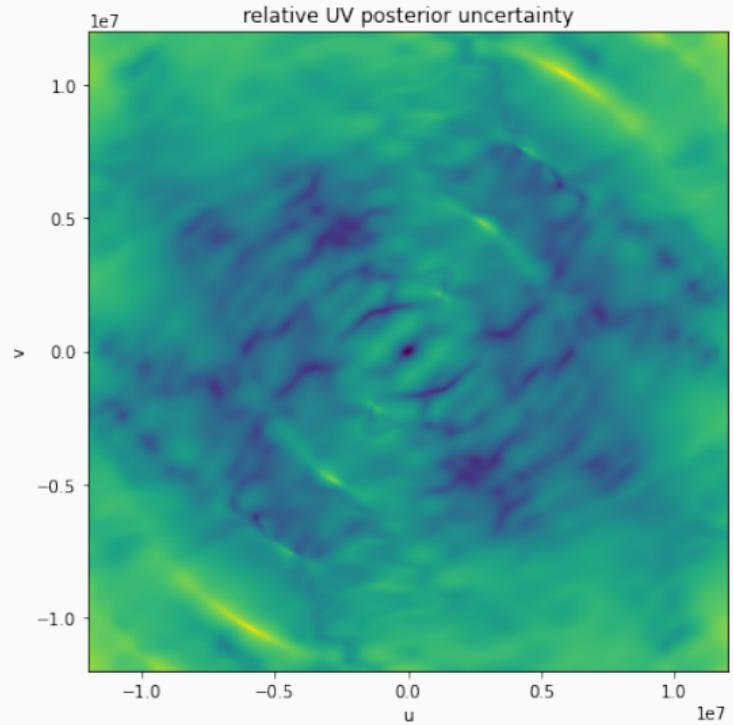
# INTERFEROMETRIC IMAGING



# INTERFEROMETRIC IMAGING



# INTERFEROMETRIC IMAGING



## CODING SESSION - RESOURCES

Coding tutorial: [https://gitlab.mpcdf.mpg.de/ift/tutorial\\_nifty\\_resolve](https://gitlab.mpcdf.mpg.de/ift/tutorial_nifty_resolve)

- Interactive CorrelatedFields tutorial: [demo\\_CorrelatedFields.ipynb](#)
- VLBI imaging demo: [demo\\_radio.ipynb](#)
- ...

nifty8: Code & Docs

- [https://gitlab.mpcdf.mpg.de/ift/nifty/-/tree/NIFTy\\_8](https://gitlab.mpcdf.mpg.de/ift/nifty/-/tree/NIFTy_8)
- <https://ift.pages.mpcdf.de/nifty/nifty8>

QUESTIONS? DISCUSSION!

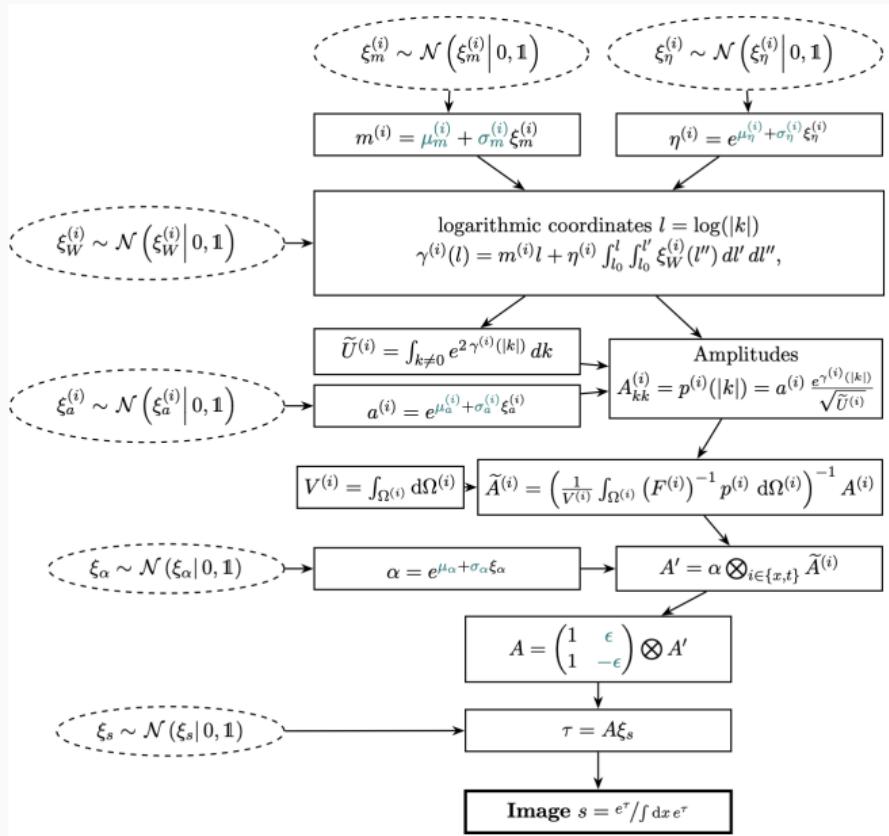
## REFERENCES

-  Philipp Arras, Mihai Baltac, Torsten A Ensslin, Philipp Frank, Sebastian Hutschenreuter, Jakob Knollmueller, Reimar Leike, Max-Niklas Newrzella, Lukas Platz, Martin Reinecke, et al.
- Nifty5: Numerical information field theory v5.**  
*Astrophysics Source Code Library*, pages ascl–1903, 2019.
-  Philipp Arras, Philipp Frank, Philipp Haim, Jakob Knollmüller, Reimar Leike, Martin Reinecke, and Torsten Enßlin.
- M87\* in space, time, and frequency.**  
*arXiv e-prints*, page arXiv:2002.05218, February 2020.
-  Philipp Frank, Reimar Leike, and Torsten A. Enßlin.
- Geometric variational inference.**  
*Entropy*, 23(7), 2021.

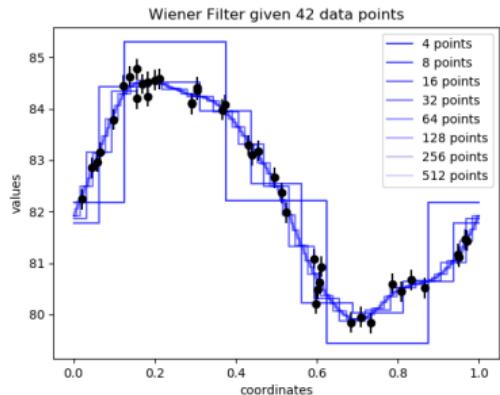
## BACKUP

---

# BACKUP



# BACKUP



```
1 import nifty8 as ift
2
3 for n in range(7):
4     # 1-dimensional regular grid space
5     # with  $2^{(n + 2)}$  pixels
6     space = ift.RGSpace(2**((n + 2)))
7
8     * * *
```