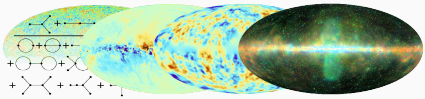
NIFTy



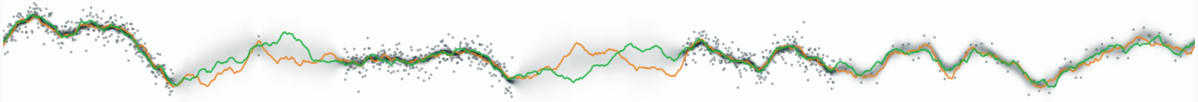
Numerical information field theory - Tutorial

VLBI imaging using IFT & RESOLVE

Philipp Frank[1], Jakob Knollmüller[2]

June 23, 2022; ngEHT Meeting, Granada, Spain

(1) Max-Planck Institute for Astrophysics, Garching, Germany
(2) Technical University TUM, Munich, Germany



Overview

NIFTy [ABE+19]:

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

Overview

NIFTy [ABE+19]:

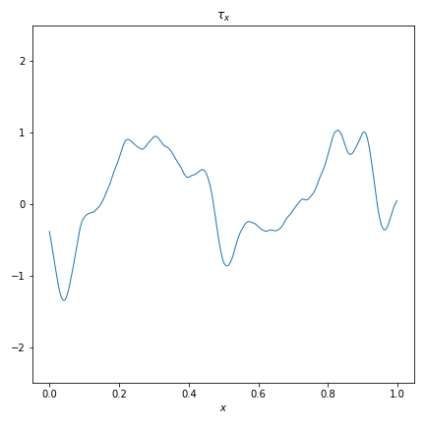
- python library for statistical inference
- differentiable generative models
- flexible gaussian processes (correlated field model)
- variety of observational likelihoods
- variational inference

Resolve:

- radio aperture syntesis algorithm
- single dish and interferometric imaging
- radio interferometric response via wgridder (ducc)
- nifty models for diffuse emission and point sources
- radio specific functionality (data handling, visualization, ...)

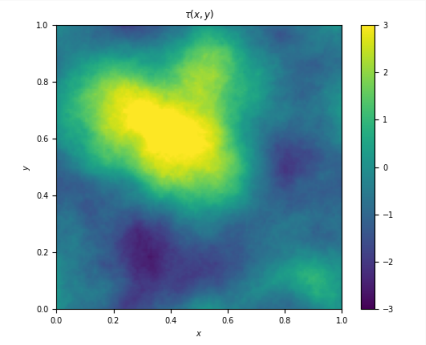
Spaces

NIFTy - Spaces



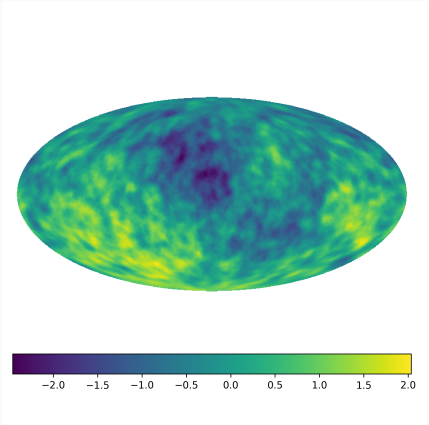

```python
import nifty8 as ift

# 1-dimensional regular grid space
# with 128 pixels and pixelsize 1/128
space = ift.RGSpace(128, 0.1)

...
```

NIFTy - Spaces



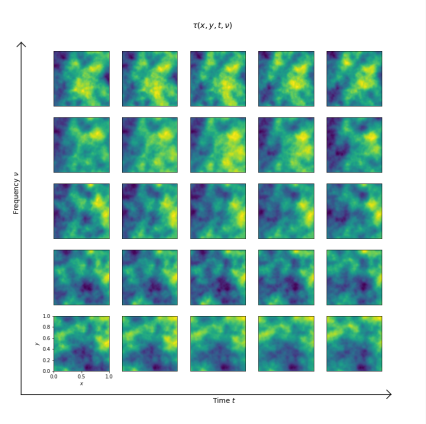

```python
import nifty8 as ift

# 2-dimensional regular grid space
# with 128x128 pixels and pixelsizes 1/128
space = ift.RGSpace((128, 128), (1/128, 1/128))

...
```

NIFTy - Spaces



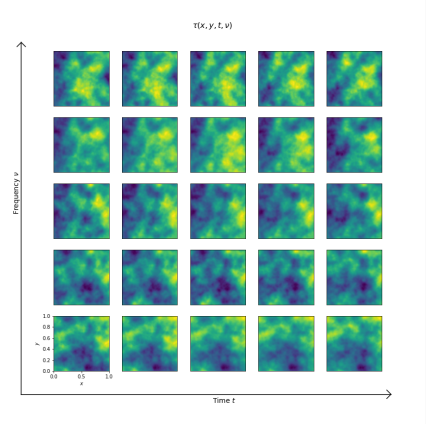

```python
import nifty8 as ift

# 2-dimensional spherical (HEALPiX) space
# with nside 128
space = ift.HPSpace(128)

...
```

NIFTy - Spaces




```python
import nifty8 as ift

# 2-dimensional regular grid space
# with 128x128 pixels
image_dom = ift.RGSpace((128, 128))

# frequency & time domain with
# 5 regularly spaced pixels
time = ift.RGSpace(5)
freq = ift.RGSpace(5)



...
```

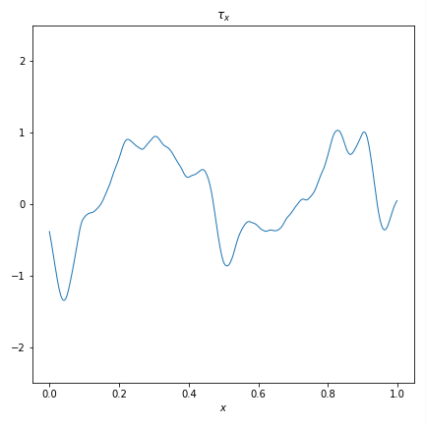
NIFTy - Spaces




```python
import nifty8 as ift

# 2-dimensional regular grid space
# with 128x128 pixels
image_dom = ift.RGSpace((128, 128))

# frequency & time domain with
# 5 regularly spaced pixels
time = ift.RGSpace(5)
freq = ift.RGSpace(5)

# Set up joint space
space = ift.makeDomain((freq, time, image_dom))

...
```

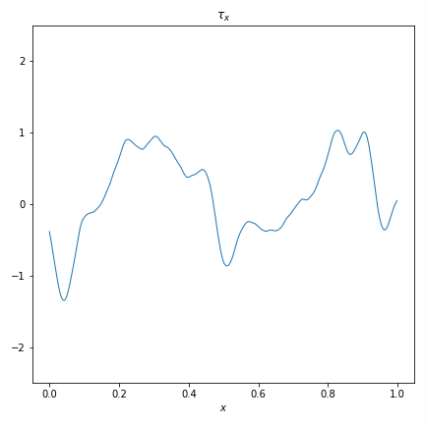
CorrelatedFields

NIFTy - CorrelatedFields [AFH+22]



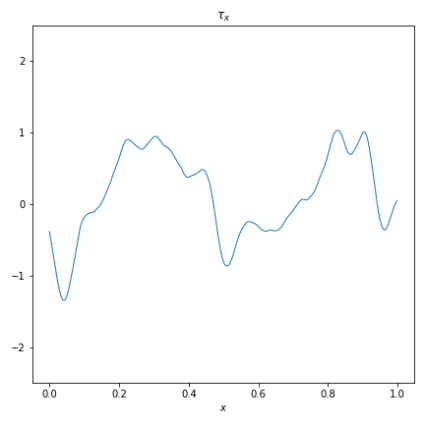

```python
import nifty8 as ift

# 1-dimensional regular grid space
# with 128 pixels and pixelsize 1/128
space = ift.RGSpace(128, 1/128)









... # Plot 'tau'
```

NIFTy - CorrelatedFields [AFH+22]



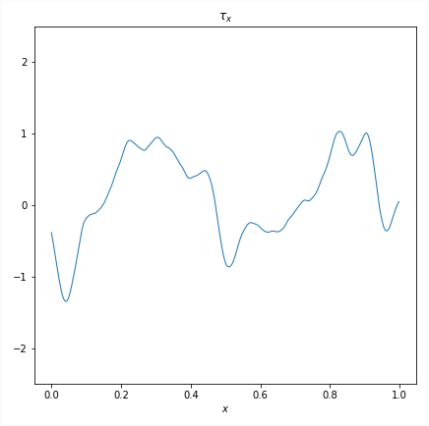

```python
import nifty8 as ift

# 1-dimensional regular grid space
# with 128 pixels and pixelsize 1/128
space = ift.RGSpace(128, 1/128)

# Define a Gaussian random processes on 'space'
args = {...} # Hyperparameters for GP model
model = ift.SimpleCorrelatedField(space, **args)









... # Plot 'tau'
```

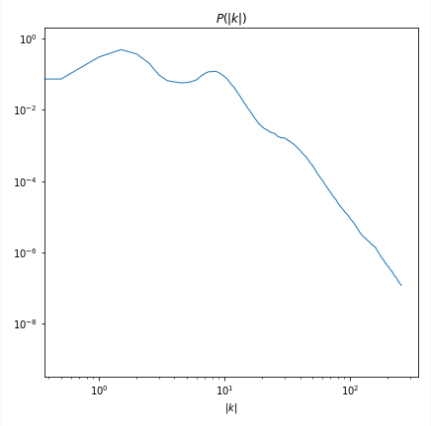
NIFTy - CorrelatedFields [AFH+22]



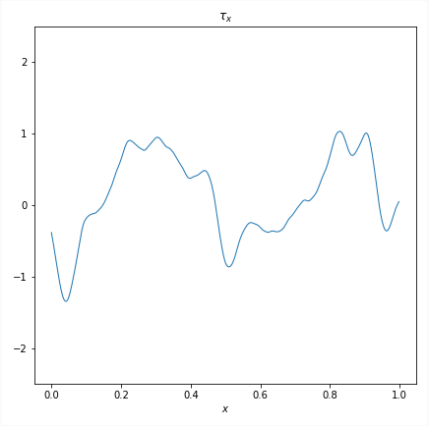

```python
import nifty8 as ift

# 1-dimensional regular grid space
# with 128 pixels and pixelsize 1/128
space = ift.RGSpace(128, 1/128)

# Define a Gaussian random processes on 'space'
args = {...} # Hyperparameters for GP model
model = ift.SimpleCorrelatedField(space, **args)

# Draw a random realization of standard normal
# distributed variables
realization = ift.from_random(model.domain)
# Apply model to get a realization
tau = model(realization)

... # Plot 'tau'
```

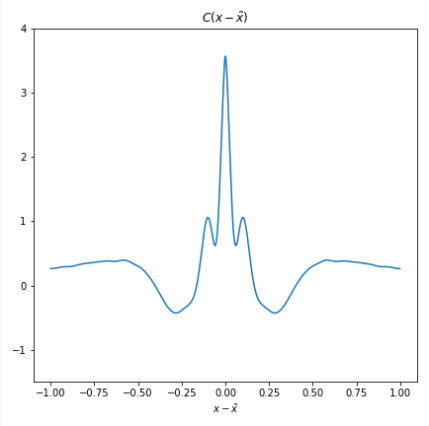
NIFTy - CorrelatedFields [AFH+22]



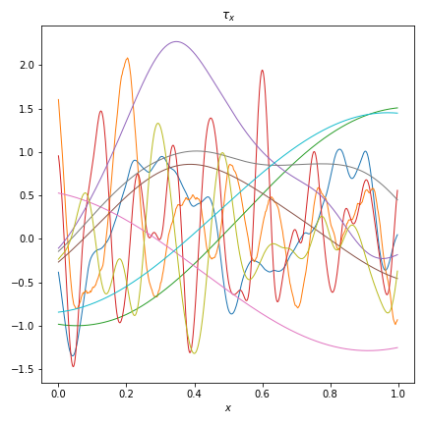


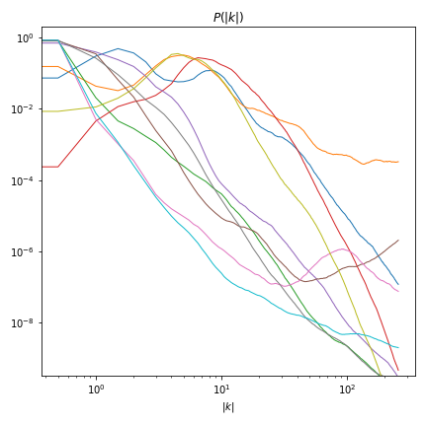
NIFTy - CorrelatedFields [AFH+22]



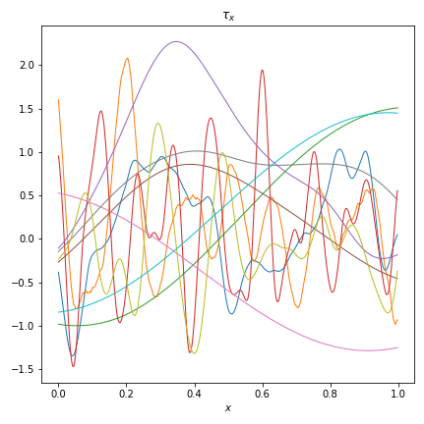


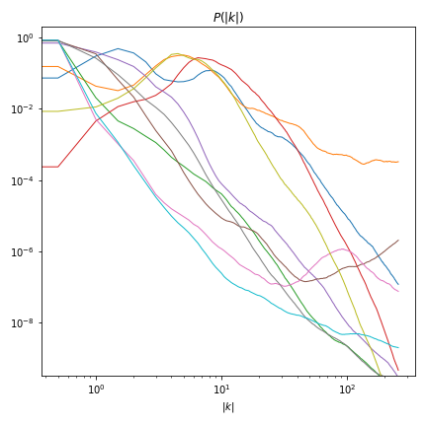
NIFTy - CorrelatedFields [AFH+22]





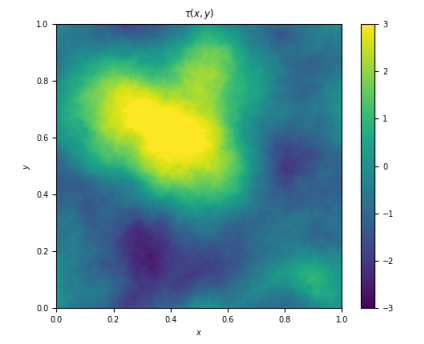
NIFTy - CorrelatedFields [AFH+22]





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

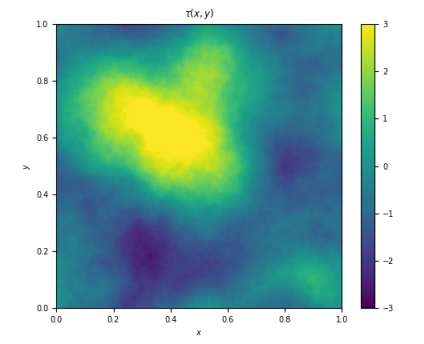
NIFTy - CorrelatedFields [AFH+22]




```python
import nifty8 as ift

# 2-dimensional regular grid space
# with 128x128 pixels and pixelsizes 1/128
space = ift.RGSpace((128, 128), (1/128, 1/128))

# Define a Gaussian random processes on 'space'
args = {...} # Hyperparameters for GP model
model = ift.SimpleCorrelatedField(space, **args)

# Draw a random realization of standard normal
# distributed variables
realization = ift.from_random(model.domain)
# Apply model to get a model realization
tau = model(realization)

... # Plot 'tau'
```

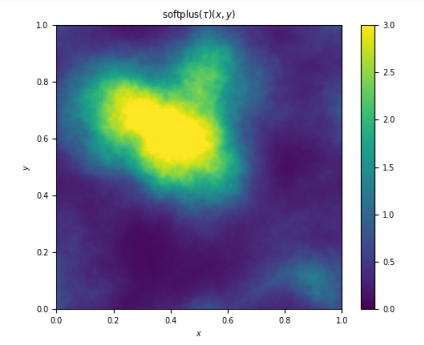
Data model & Variational inference

NIFTy - Variational Inference [FLE21]



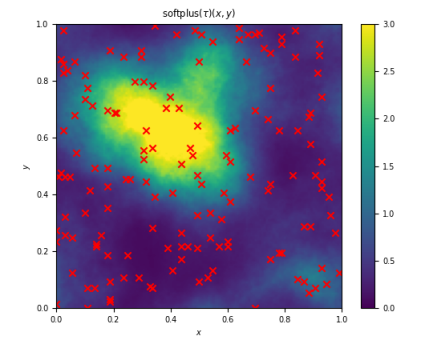

```
model = ... # model for tau


...
```

NIFTy - Variational Inference [FLE21]



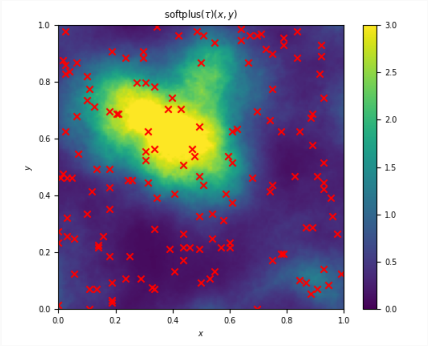

```
model = ift.softplus(model) # apply nonlinearity
...
```

NIFTy - Variational Inference [FLE21]



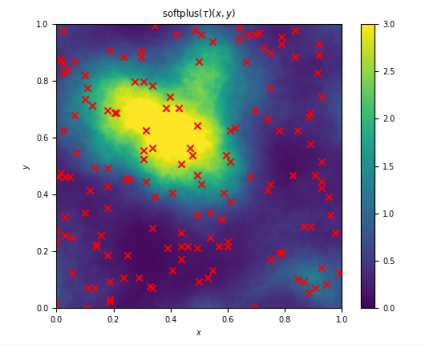

```python
model = ift.softplus(model) # apply nonlinearity

# Set up a random response
flags = np.random.binomial(1, 0.99, size = 128**2)
flags = ift.makeField(space, flags)
Response = ift.MaskOperator(flags)



...
```

NIFTy - Variational Inference [FLE21]



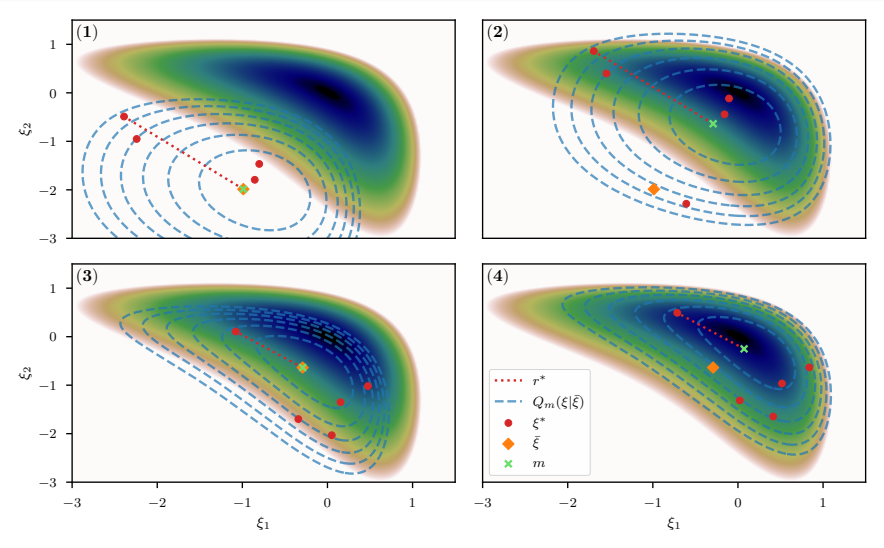

```python
model = ift.softplus(model) # apply nonlinearity

# Set up a random response
flags = np.random.binomial(1, 0.99, size = 128**2)
flags = ift.makeField(space, flags)
Response = ift.MaskOperator(flags)

# Define observational model and likelihood
data, noise_icov = # load data and noise
lh = ift.GaussianEnergy(data = data,
        inverse_covariance = noise_icov)
likelihood = lh @ Response(model)



...
```

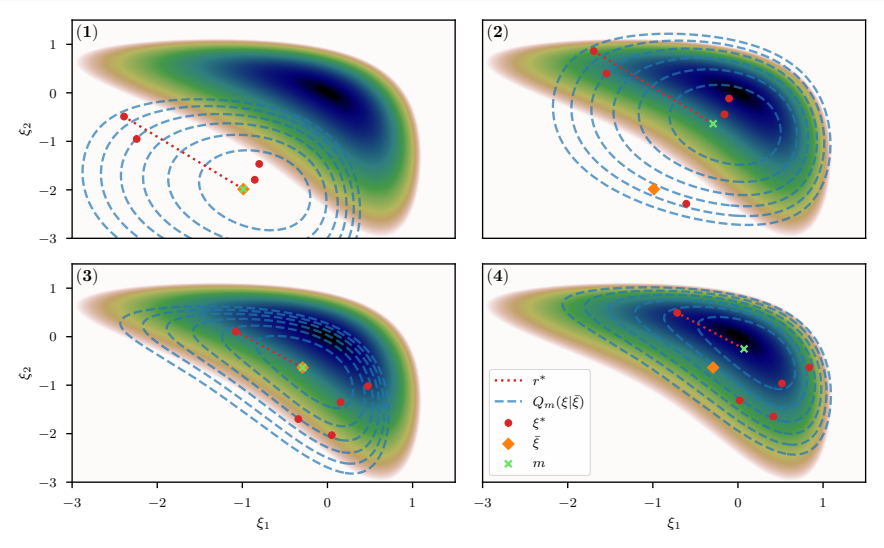
NIFTy - Variational Inference [FLE21]




```python
model = ift.softplus(model) # apply nonlinearity

# Set up a random response
flags = np.random.binomial(1, 0.99, size = 128**2)
flags = ift.makeField(space, flags)
Response = ift.MaskOperator(flags)

# Define observational model and likelihood
data, noise_icov = # load data and noise
lh = ift.GaussianEnergy(data = data,
        inverse_covariance = noise_icov)
likelihood = lh @ Response(model)

# Generate approximate posterior samples
# using variational inference (geoVI)
samples = ift.optimize_kl(lh, **params)
...
```

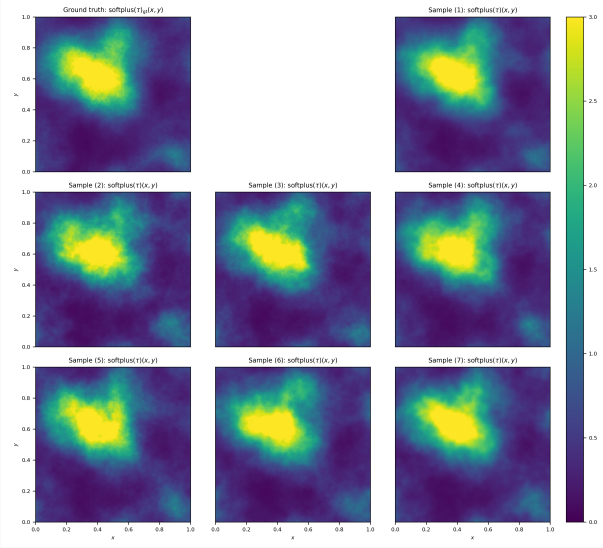
NIFTy - Variational Inference [FLE21]



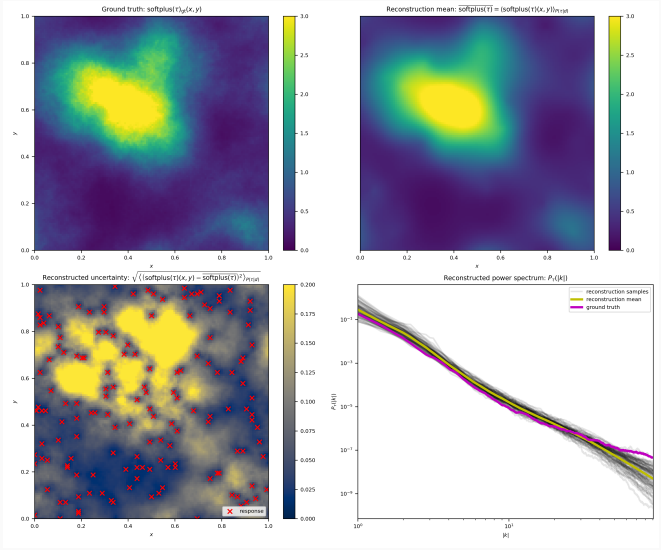


→ not 2- but (# of pixels + # spectrum parameters)-dimensional probability distributions!

NIFTy - Variational Inference [FLE21]



NIFTy - Variational Inference [FLE21]



Coding Session - Resources

Coding tutorial: **https://gitlab.mpcdf.mpg.de/ift/tutorial_nifty_resolve**

- Interactive CorrelatedFields tutorial: **demo_CorrelatedFields.ipynb**
- VLBI imaging demo: **demo_radio.ipynb**

Code

- nifty8: **https://gitlab.mpcdf.mpg.de/ift/nifty/-/tree/NIFTy_8**
- resolve: **https://gitlab.mpcdf.mpg.de/ift/resolve**

Docs

- nifty8: **https://ift.pages.mpcdf.de/nifty/nifty8**
- resolve: **https://ift.pages.mpcdf.de/resolve**

Questions? Discussion!

References

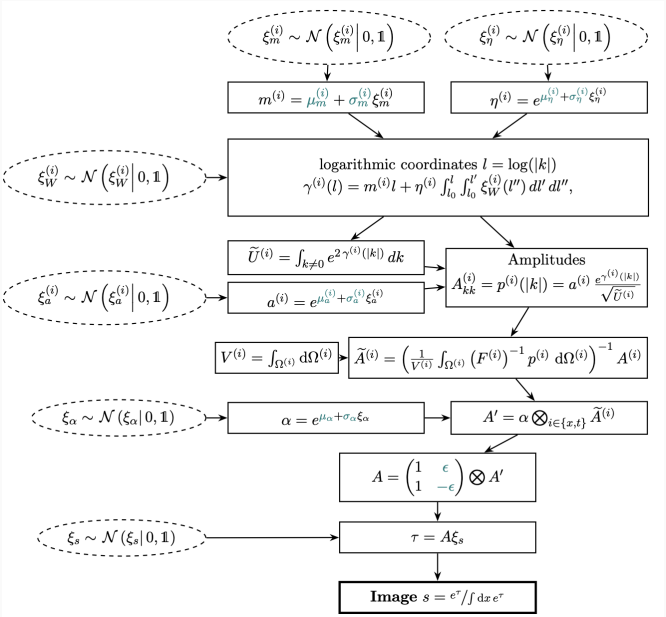
Philipp Arras, Mihai Baltac, Torsten A Ensslin, Philipp Frank, Sebastian Hutschenreuter, Jakob Knollmueller, Reimar Leike, Max-Niklas Newrzella, Lukas Platz, Martin Reinecke, et al.
Nifty5: Numerical information field theory v5.
Astrophysics Source Code Library, pages ascl–1903, 2019.

Philipp Arras, Philipp Frank, Philipp Haim, Jakob Knollmüller, Reimar Leike, Martin Reinecke, and Torsten Enßlin.
Variable structures in m87* from space, time and frequency resolved interferometry.
Nature Astronomy, 6(2):259–269, 2022.

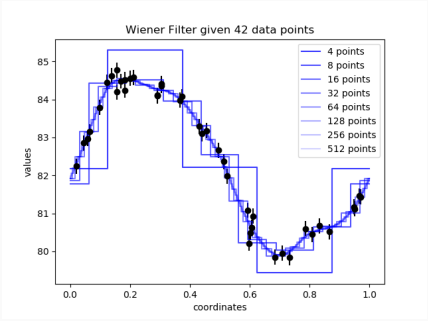
Philipp Frank, Reimar Leike, and Torsten A. Enßlin.
Geometric variational inference.
Entropy, 23(7), 2021.

Backup

Backup


$\xi_m^{(i)} \sim \mathcal{N}\left(\xi_m^{(i)} \middle| 0, \mathbb{1}\right)$
$\xi_\eta^{(i)} \sim \mathcal{N}\left(\xi_\eta^{(i)} \middle| 0, \mathbb{1}\right)$
$m^{(i)} = \mu_m^{(i)} + \sigma_m^{(i)} \xi_m^{(i)}$
$\eta^{(i)} = e^{\mu_\eta^{(i)} + \sigma_\eta^{(i)} \xi_\eta^{(i)}}$
$\xi_W^{(i)} \sim \mathcal{N}\left(\xi_W^{(i)} \middle| 0, \mathbb{1}\right)$
logarithmic coordinates $l = \log(|k|)$
$\gamma^{(i)}(l) = m^{(i)} l + \eta^{(i)} \int_{l_0}^{l} \int_{l_0}^{l'} \xi_W^{(i)}(l'')\, dl'\, dl''$,
$\widetilde{U}^{(i)} = \int_{k \neq 0} e^{2\gamma^{(i)}(|k|)}\, dk$
$\xi_a^{(i)} \sim \mathcal{N}\left(\xi_a^{(i)} \middle| 0, \mathbb{1}\right)$
$a^{(i)} = e^{\mu_a^{(i)} + \sigma_a^{(i)} \xi_a^{(i)}}$
Amplitudes
$A_{kk}^{(i)} = p^{(i)}(|k|) = a^{(i)} \frac{e^{\gamma^{(i)}(|k|)}}{\sqrt{\widetilde{U}^{(i)}}}$
$V^{(i)} = \int_{\Omega^{(i)}} \mathrm{d}\Omega^{(i)}$
$\widetilde{A}^{(i)} = \left(\frac{1}{V^{(i)}} \int_{\Omega^{(i)}} \left(F^{(i)}\right)^{-1} p^{(i)}\, \mathrm{d}\Omega^{(i)}\right)^{-1} A^{(i)}$
$\xi_\alpha \sim \mathcal{N}(\xi_\alpha | 0, \mathbb{1})$
$\alpha = e^{\mu_\alpha + \sigma_\alpha \xi_\alpha}$
$A' = \alpha \bigotimes_{i \in \{x,t\}} \widetilde{A}^{(i)}$
$A = \begin{pmatrix} 1 & \epsilon \\ 1 & -\epsilon \end{pmatrix} \otimes A'$
$\xi_s \sim \mathcal{N}(\xi_s | 0, \mathbb{1})$
$\tau = A \xi_s$
Image $s = e^\tau / \int \mathrm{d}x\, e^\tau$

Backup




```python
import nifty8 as ift

for n in range(7):
        # 1-dimensional regular grid space
        # with 2^(n + 2) pixels
        space = ift.RGSpace(2**(n + 2))

        ...
```