VARIATIONAL BAYESIAN INFERENCE MODEL FOR SHIP DETECTION USING POLARIMETRIC SAR IMAGES

Group 21:

Vinit Singh (16PH20036)

Barun Das (16MT10012)

Kshitij Kumar Singh (16IM30023)

Objective

- To express the polarimetric SAR image as a tensor, and decompose the SAR image as the sum of a sparse component associated with ships and a sea clutter component. These components are denoted by some latent variables.
- Then, introduce hierarchical priors of the latent variables to establish the probabilistic model of ship detection.
- By using variational Bayesian inference, estimate the posterior distributions of the latent variables.
- Finally, the ship detection result is obtained in the iterative Bayesian inference process

SAR Imaging

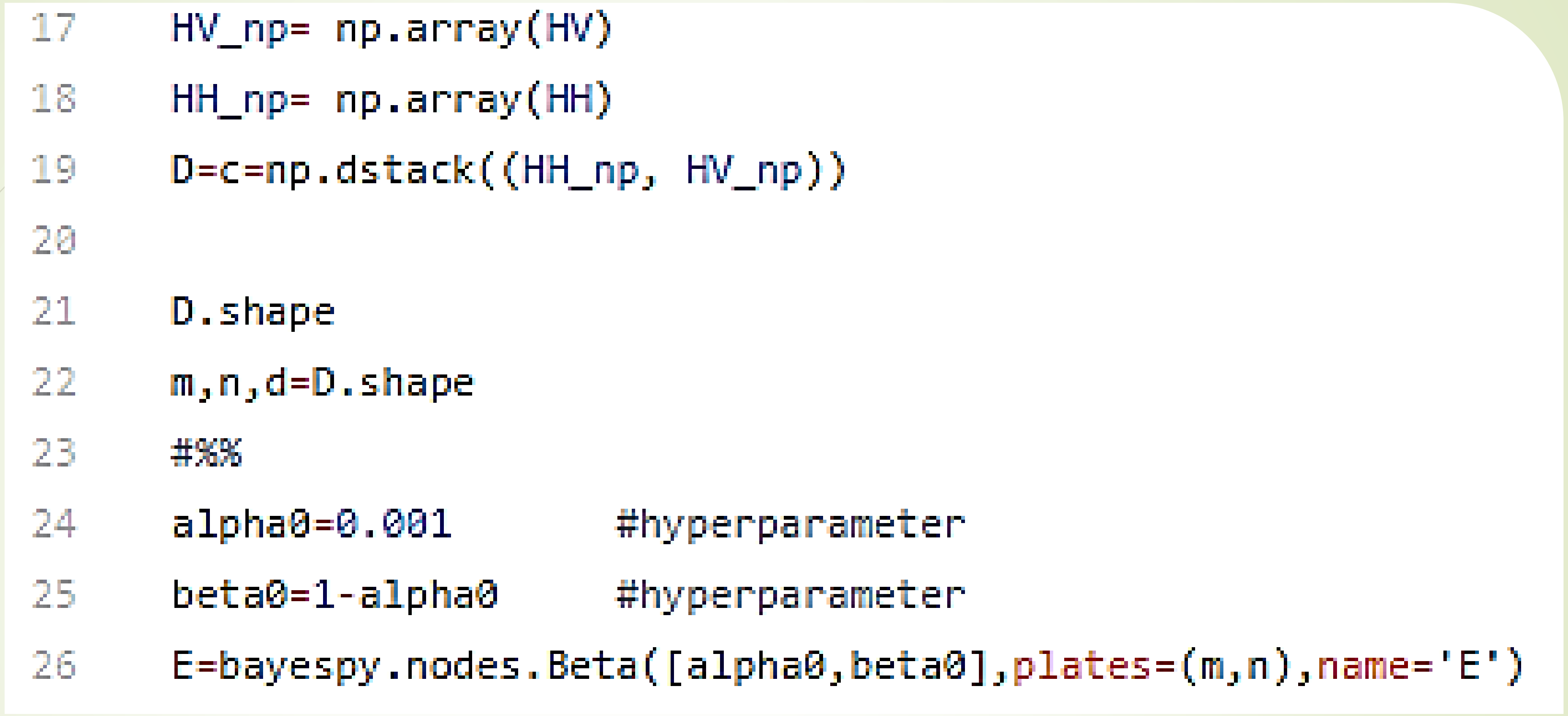
- **Synthetic-aperture radar** (**SAR**) is a form of radar that is used to create two-dimensional images or three-dimensional reconstructions of objects, such as landscapes. SAR uses the motion of the radar antenna over a target region to provide finer spatial resolution than conventional beam-scanning radars. SAR is typically mounted on a moving platform, such as an aircraft or spacecraft.
- The Sentinel-1 mission comprises a constellation of two polar-orbiting satellites, operating day and night performing C-band synthetic aperture radar imaging, enabling them to acquire imagery regardless of the weather.
- Sentinel-1 will work in a pre-programmed operation mode to avoid conflicts and to produce a consistent long-term data archive built for applications based on long time series.

Variational Bayesian Inference

- Variational Bayesian methods are a family of techniques for approximating intractable integrals arising in Bayesian inference and machine learning. They are typically used in complex statistical models consisting of observed variables (usually termed "data") as well as unknown parameters and latent variables, with various sorts of relationships among the three types of random variables, as might be described by a graphical model. As is typical in Bayesian inference, the parameters and latent variables are grouped together as "unobserved variables".

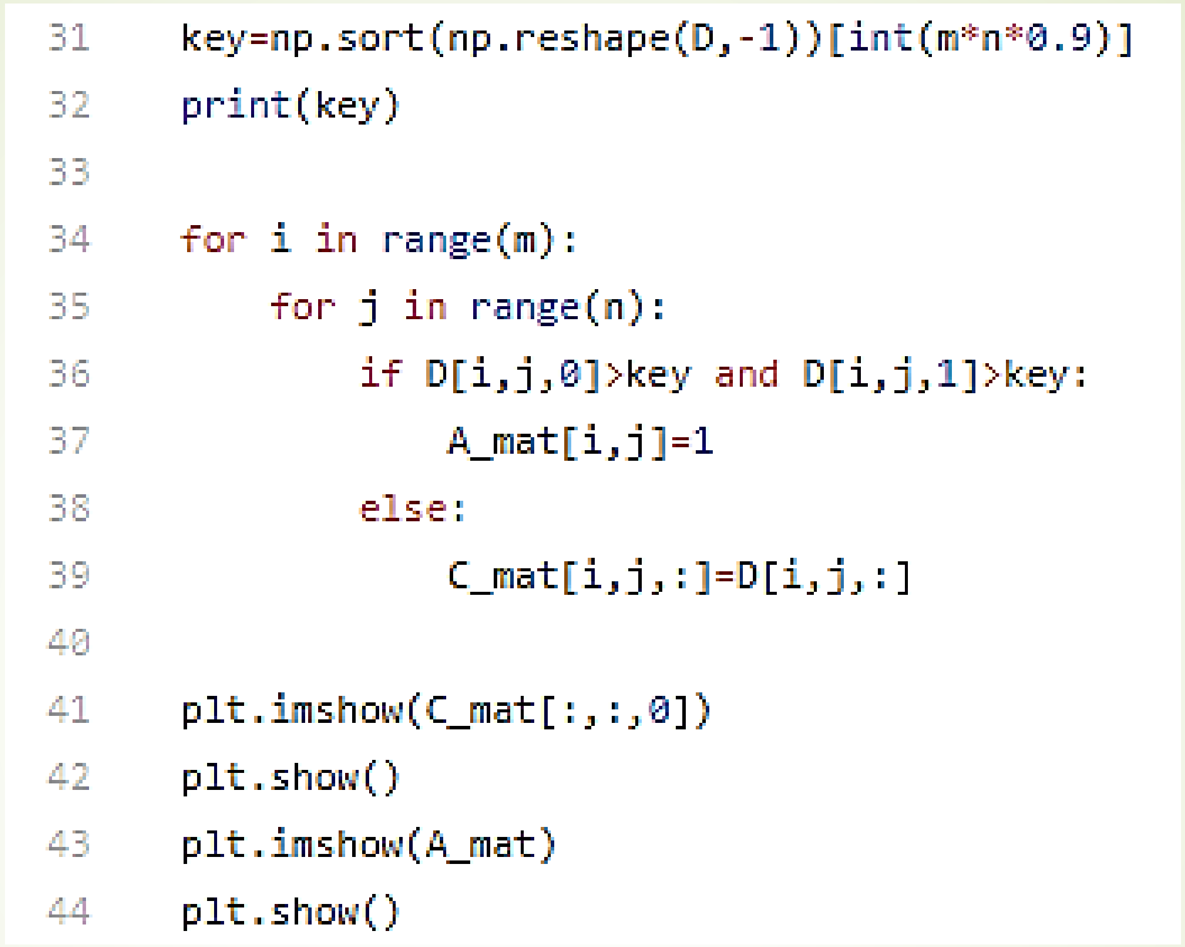
Our Approach

- Parameter initialization
- Image initialization
- Modelling the sea clutter component and ship component
- Training
- Running the Bayesian inference model
- Training the final model



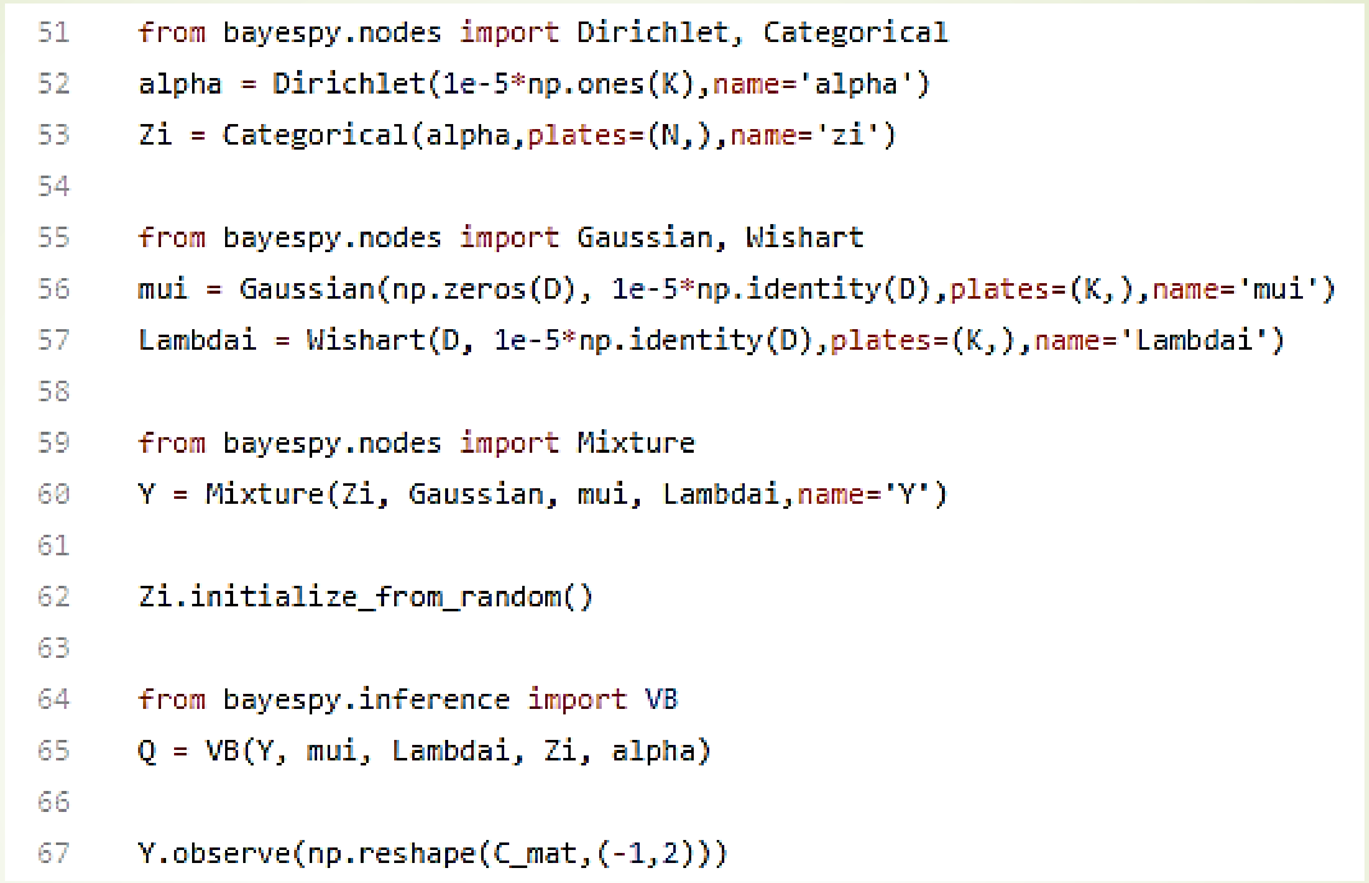
```
HV_np= np.array(HV)
HH_np= np.array(HH)
D=c=np.dstack((HH_np, HV_np))

D.shape
m,n,d=D.shape
#%%
alpha0=0.001       #hyperparameter
beta0=1-alpha0     #hyperparameter
E=bayespy.nodes.Beta([alpha0,beta0],plates=(m,n),name='E')
```

Parameter initialization



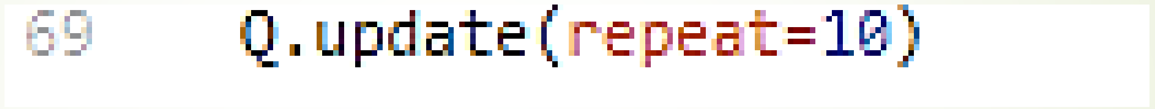
```
key=np.sort(np.reshape(D,-1))[int(m*n*0.9)]
print(key)

for i in range(m):
    for j in range(n):
        if D[i,j,0]>key and D[i,j,1]>key:
            A_mat[i,j]=1
        else:
            C_mat[i,j,:]=D[i,j,:]

plt.imshow(C_mat[:,:,0])
plt.show()
plt.imshow(A_mat)
plt.show()
```

Image Initialization



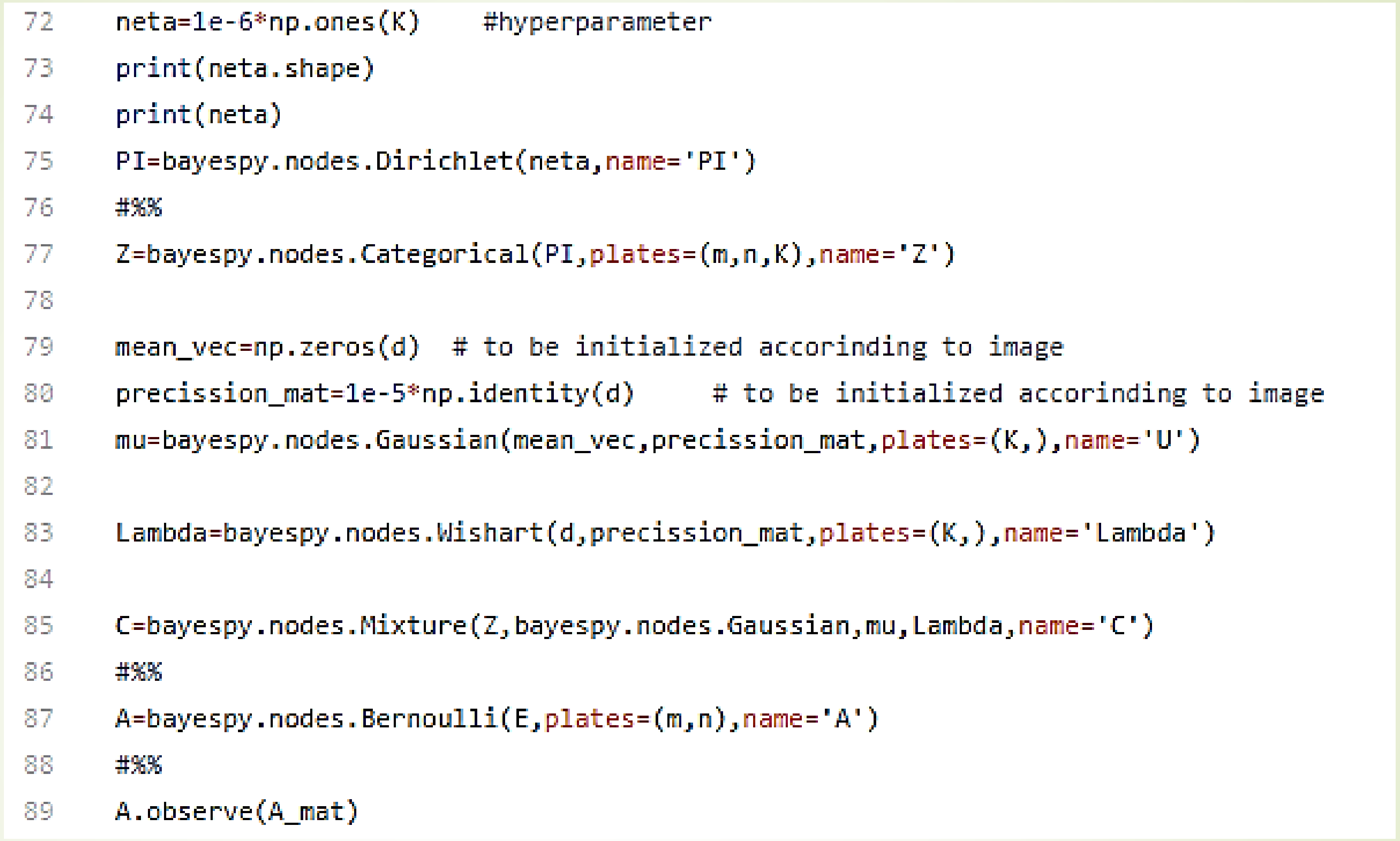
```python
from bayespy.nodes import Dirichlet, Categorical
alpha = Dirichlet(1e-5*np.ones(K),name='alpha')
Zi = Categorical(alpha,plates=(N,),name='zi')

from bayespy.nodes import Gaussian, Wishart
mui = Gaussian(np.zeros(D), 1e-5*np.identity(D),plates=(K,),name='mui')
Lambdai = Wishart(D, 1e-5*np.identity(D),plates=(K,),name='Lambdai')

from bayespy.nodes import Mixture
Y = Mixture(Zi, Gaussian, mui, Lambdai,name='Y')

Zi.initialize_from_random()

from bayespy.inference import VB
Q = VB(Y, mui, Lambdai, Zi, alpha)

Y.observe(np.reshape(C_mat,(-1,2)))
```

Modelling sea clutter (c) and ship (a) component



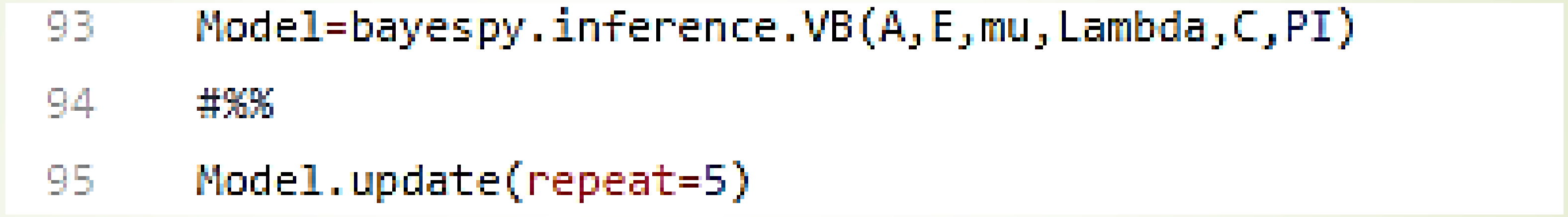
```
    Q.update(repeat=10)
```

Training the model



```
neta=1e-6*np.ones(K)    #hyperparameter
print(neta.shape)
print(neta)
PI=bayespy.nodes.Dirichlet(neta,name='PI')
#%%
Z=bayespy.nodes.Categorical(PI,plates=(m,n,K),name='Z')

mean_vec=np.zeros(d)  # to be initialized accorinding to image
precission_mat=1e-5*np.identity(d)     # to be initialized accorinding to image
mu=bayespy.nodes.Gaussian(mean_vec,precission_mat,plates=(K,),name='U')

Lambda=bayespy.nodes.Wishart(d,precission_mat,plates=(K,),name='Lambda')

C=bayespy.nodes.Mixture(Z,bayespy.nodes.Gaussian,mu,Lambda,name='C')
#%%
A=bayespy.nodes.Bernoulli(E,plates=(m,n),name='A')
#%%
A.observe(A_mat)
```

Running the Bayesian inference model

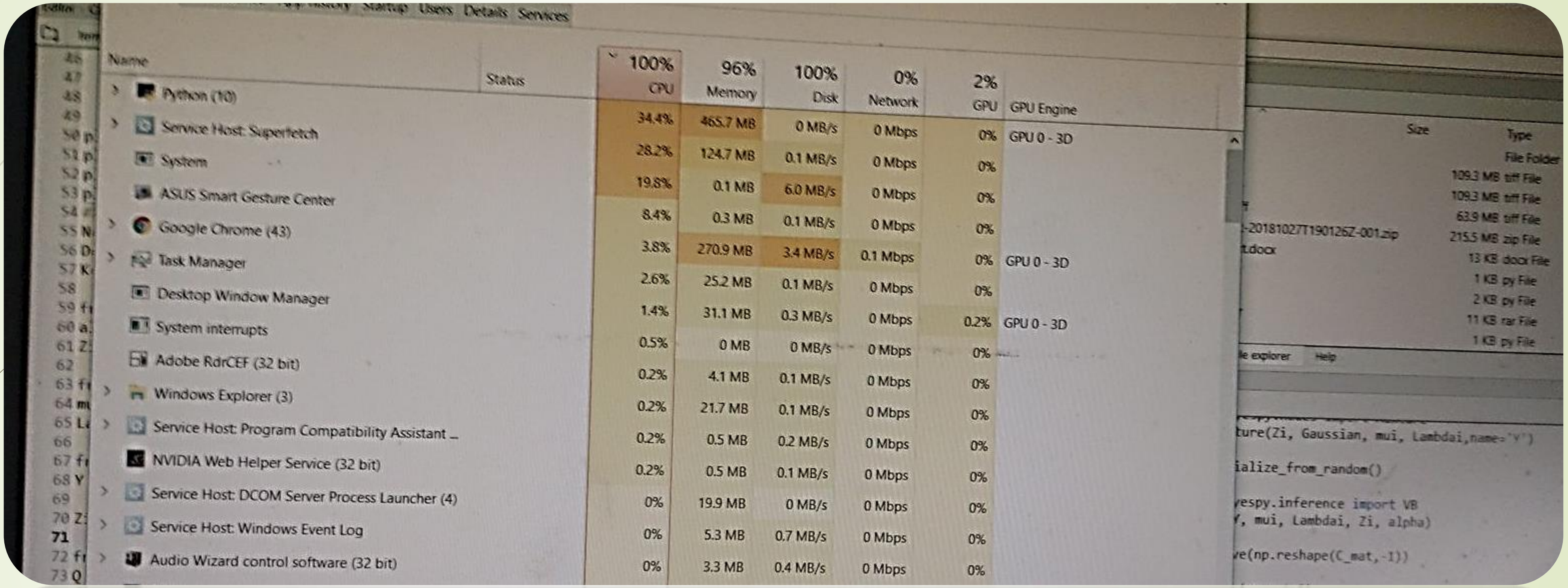


```
Model=bayespy.inference.VB(A,E,mu,Lambda,C,PI)
#%%
Model.update(repeat=5)
```

Training for the final model

Problems faced

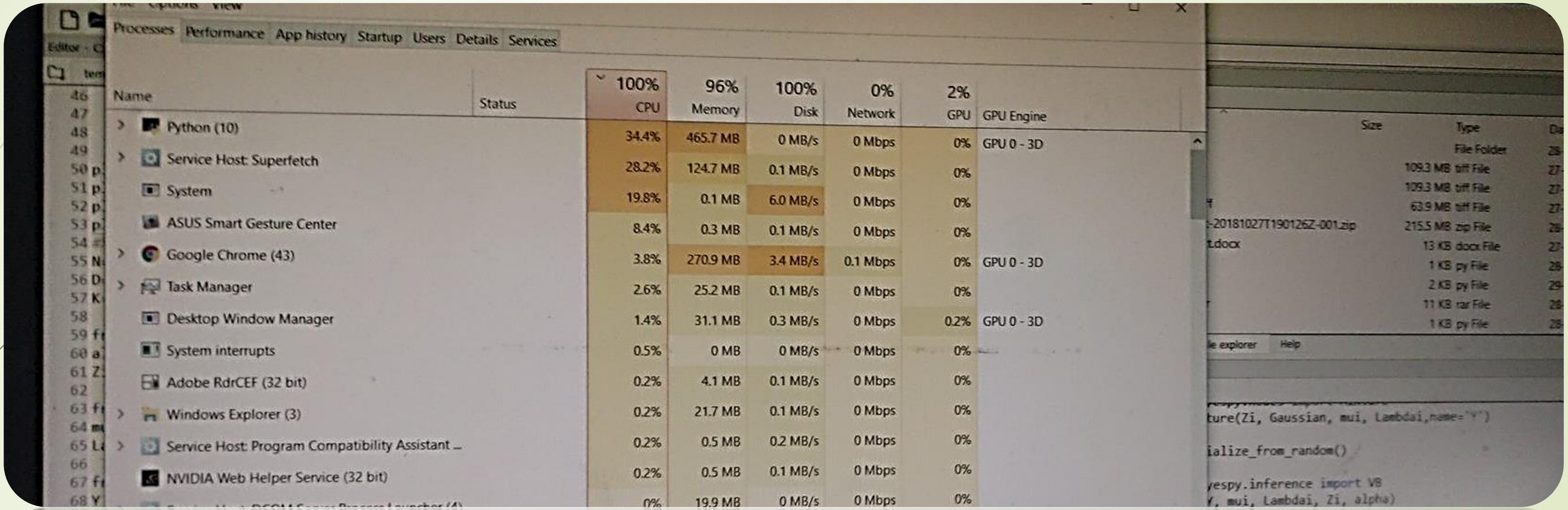
- The model ran for a small part of the image
- The actual image was too big and posed computational problems
- Our machines were unable to run computations for such a large set of pixels.
- We repeatedly ran out of computation power



Name	Status	100% CPU	96% Memory	100% Disk	0% Network	2% GPU	GPU Engine
Python (10)		34.4%	465.7 MB	0 MB/s	0 Mbps	0%	GPU 0 - 3D
Service Host: Superfetch		28.2%	124.7 MB	0.1 MB/s	0 Mbps	0%	
System		19.8%	0.1 MB	6.0 MB/s	0 Mbps	0%	
ASUS Smart Gesture Center		8.4%	0.3 MB	0.1 MB/s	0 Mbps	0%	
Google Chrome (43)		3.8%	270.9 MB	3.4 MB/s	0.1 Mbps	0%	GPU 0 - 3D
Task Manager		2.6%	25.2 MB	0.1 MB/s	0 Mbps	0%	
Desktop Window Manager		1.4%	31.1 MB	0.3 MB/s	0 Mbps	0.2%	GPU 0 - 3D
System interrupts		0.5%	0 MB	0 MB/s	0 Mbps	0%	
Adobe RdrCEF (32 bit)		0.2%	4.1 MB	0.1 MB/s	0 Mbps	0%	
Windows Explorer (3)		0.2%	21.7 MB	0.1 MB/s	0 Mbps	0%	
Service Host: Program Compatibility Assistant ...		0.2%	0.5 MB	0.2 MB/s	0 Mbps	0%	
NVIDIA Web Helper Service (32 bit)		0.2%	0.5 MB	0.1 MB/s	0 Mbps	0%	
Service Host: DCOM Server Process Launcher (4)		0%	19.9 MB	0 MB/s	0 Mbps	0%	
Service Host: Windows Event Log		0%	5.3 MB	0.7 MB/s	0 Mbps	0%	
Audio Wizard control software (32 bit)		0%	3.3 MB	0.4 MB/s	0 Mbps	0%	

```
ture(Zi, Gaussian, mui, Lambdai,name='Y')

ialize_from_random()

yespy.inference import VB
Y, mui, Lambdai, Zi, alpha)

ve(np.reshape(C_mat,-1))
```

Issues with computational load

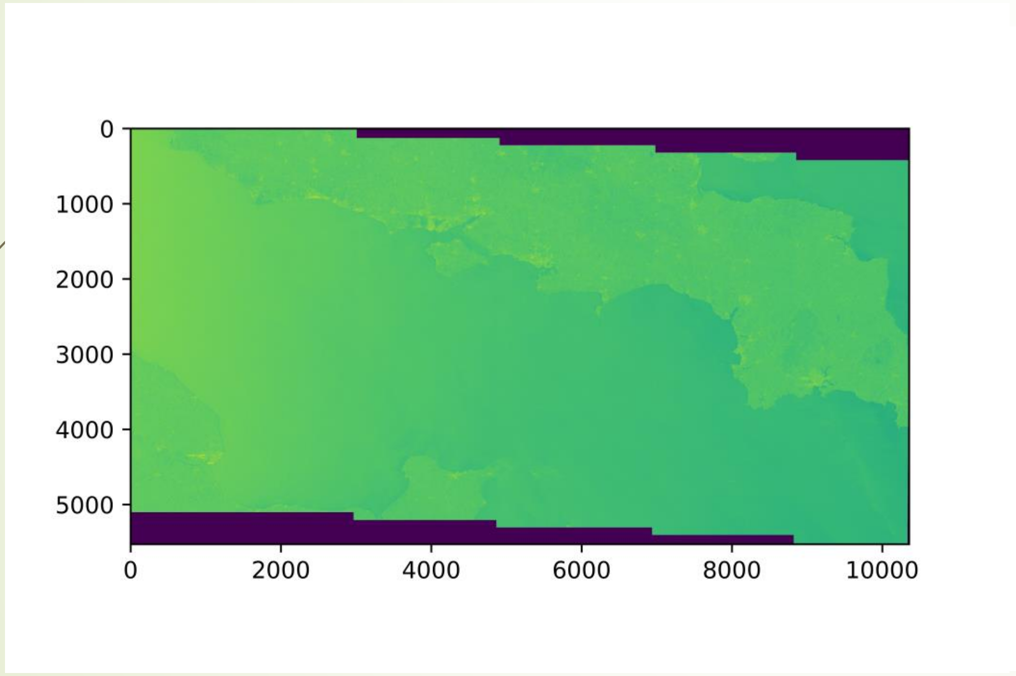


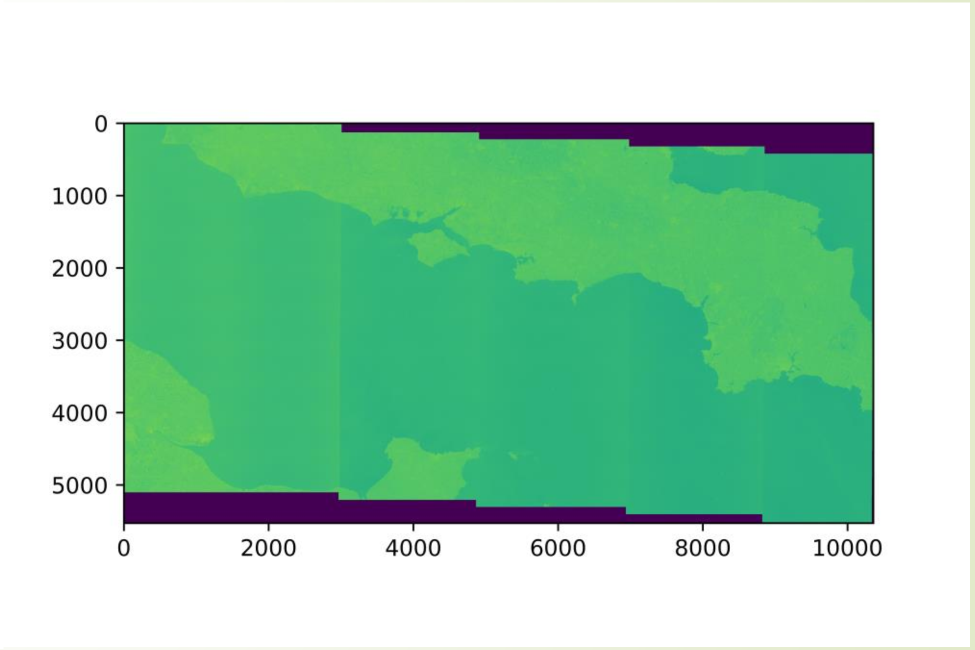
Name	Status	100% CPU	96% Memory	100% Disk	0% Network	2% GPU	GPU Engine
Python (10)		34.4%	465.7 MB	0 MB/s	0 Mbps	0%	GPU 0 - 3D
Service Host: Superfetch		28.2%	124.7 MB	0.1 MB/s	0 Mbps	0%	
System		19.8%	0.1 MB	6.0 MB/s	0 Mbps	0%	
ASUS Smart Gesture Center		8.4%	0.3 MB	0.1 MB/s	0 Mbps	0%	
Google Chrome (43)		3.8%	270.9 MB	3.4 MB/s	0.1 Mbps	0%	GPU 0 - 3D
Task Manager		2.6%	25.2 MB	0.1 MB/s	0 Mbps	0%	
Desktop Window Manager		1.4%	31.1 MB	0.3 MB/s	0 Mbps	0.2%	GPU 0 - 3D
System interrupts		0.5%	0 MB	0 MB/s	0 Mbps	0%	
Adobe RdrCEF (32 bit)		0.2%	4.1 MB	0.1 MB/s	0 Mbps	0%	
Windows Explorer (3)		0.2%	21.7 MB	0.1 MB/s	0 Mbps	0%	
Service Host: Program Compatibility Assistant ...		0.2%	0.5 MB	0.2 MB/s	0 Mbps	0%	
NVIDIA Web Helper Service (32 bit)		0.2%	0.5 MB	0.1 MB/s	0 Mbps	0%	
[illegible]		0%	19.9 MB	0 MB/s	0 Mbps	0%	

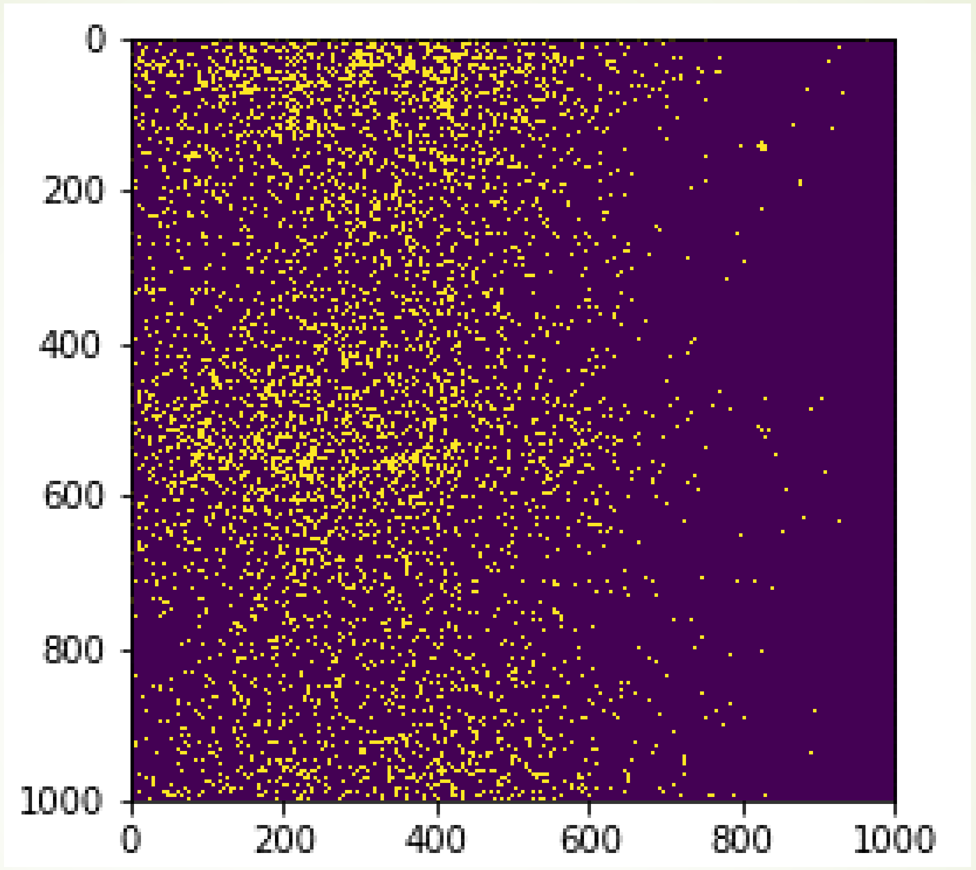
```
ture(Zi, Gaussian, mui, Lambdai,name='Y')

ialize_from_random()

yespy.inference import VB
Y, mui, Lambdai, Zi, alpha)
```

Issues with computational load

Input Images (.png for illustration)







Result (.png for illustration)