Closure phase interpretation using the pix2pix machine learning algorithm

Abby Morris and Tom Farr

MANCHESTER 1824
The University of Manchester

Introduction

- Self-calibration in interferometry uses assumptions which affects the output image
- Chael et al. used a regularized maximum likelihood algorithm to image from closure phases
- We used the pix2pix algorithm developed by Isola et al. to image source structures directly from closure phases
- In this talk we will cover the various architectures used in training, how we simulated large quantities of training data, and some of the key results that were obtained.

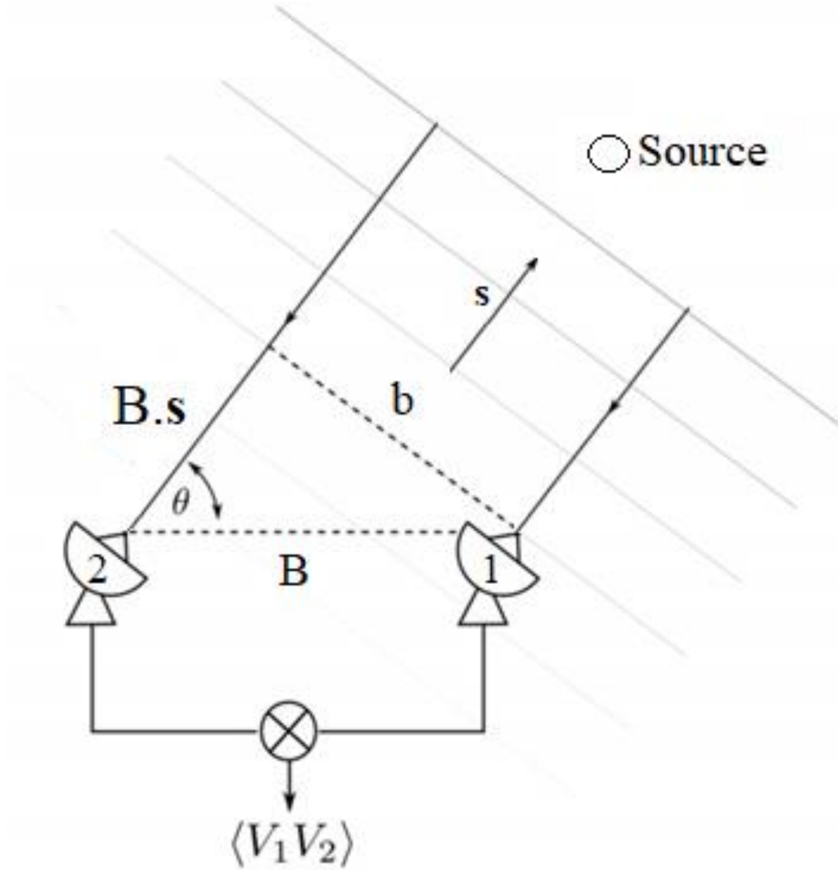
Objectives

- To prove machine learning is a viable option for imaging source structures from plots of closure phase.
- To replace the imaging method used by Chael et al. (2018), or to reduce the number of their imaging rounds needed.

The University of Manchester

Interferometers

$$\theta_{\rm res} = \frac{\lambda}{B_{\rm max}} \qquad (1)$$

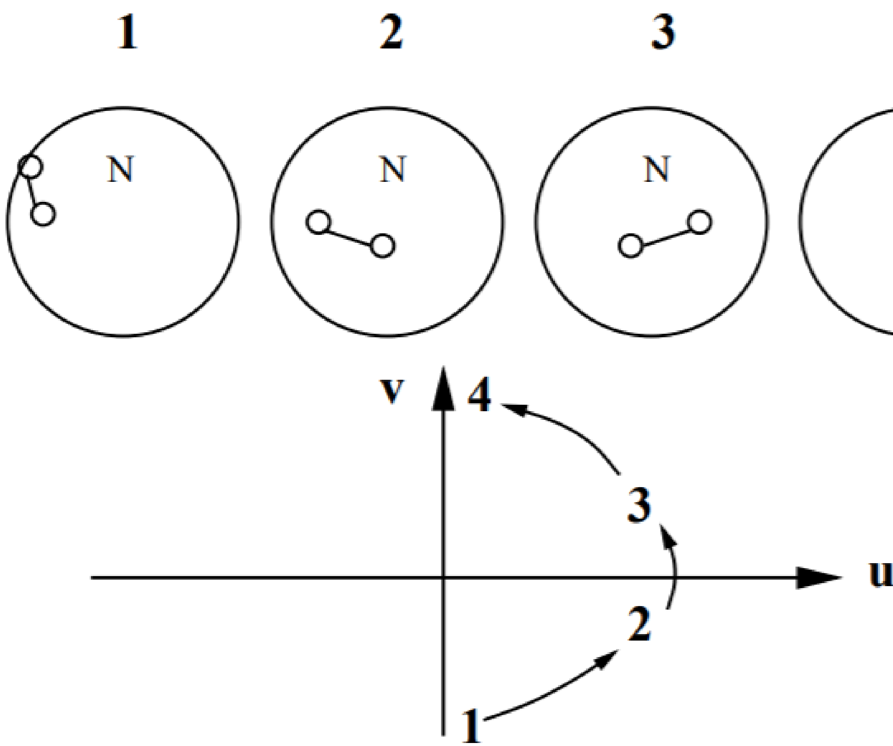


$$V(u,v) = \iint I(x,y) e^{-2\pi i(ux+vy)}\,\mathrm{d}x\,\mathrm{d}y \qquad (2)$$

$$I_D(x,y) = I(x,y) * \iint S(u,v) e^{-2\pi i(ux+vy)}\,\mathrm{d}u\,\mathrm{d}v \qquad (3)$$

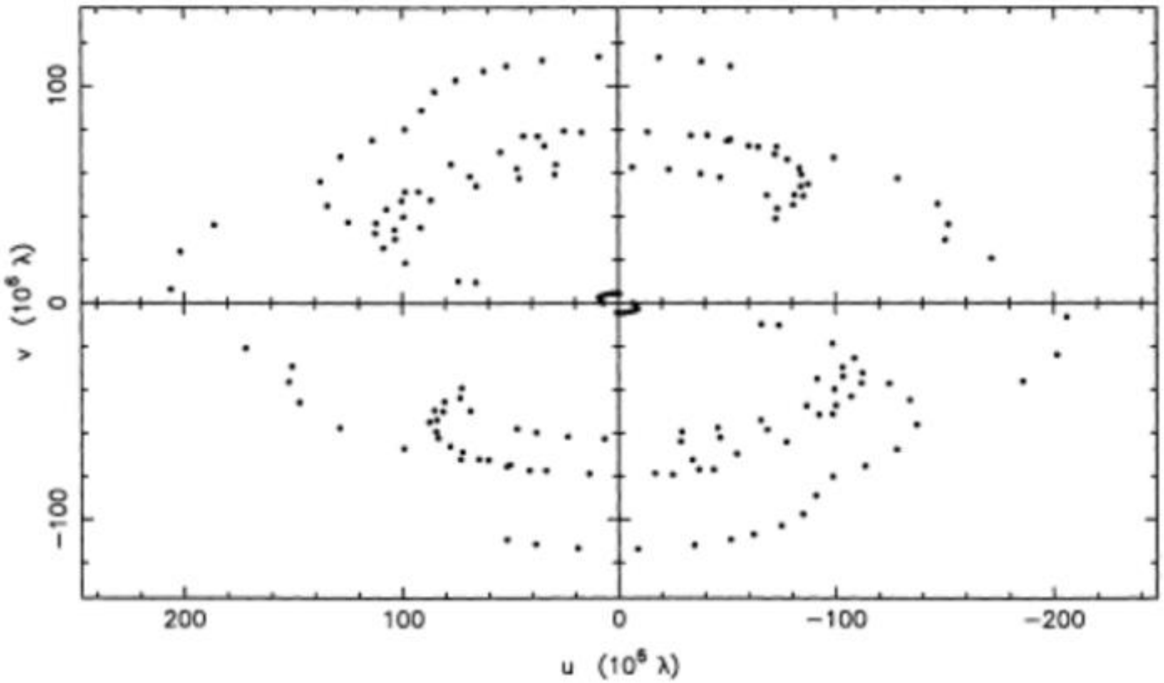
The University of Manchester

UV plane coverage



The University of Manchester

UV plane coverage

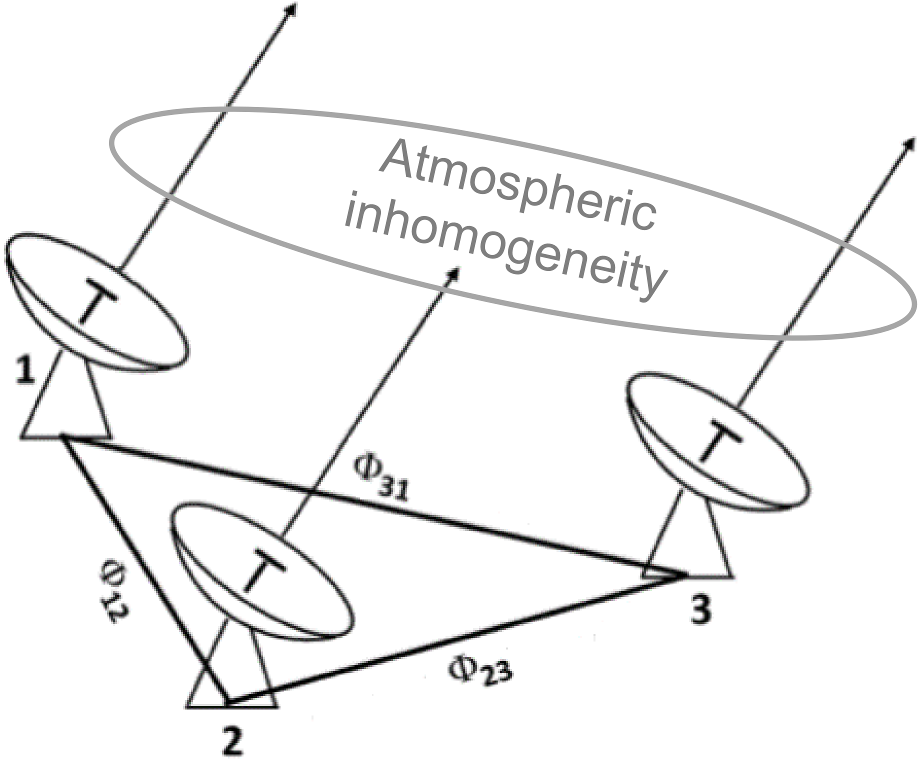


The University of Manchester

Closure phase

$$\Phi_{12} = \psi_{12} + e_1 - e_2 \quad (1)$$

$$C_{123} = \Phi_{12} + \Phi_{23} + \Phi_{31}$$
$$= \psi_{12} + \psi_{23} + \psi_{31} \quad (2)$$



The University of Manchester

Source flux requirements

$$V(u,v) = \iint (\delta(x,y) + \alpha\delta(x - x_0, y))e^{2\pi i(ux+vy)}\,dx\,dy \tag{1}$$

$$V_{12} = 1 + \alpha e^{2\pi i(u_2 - u_1)x_0} \tag{2}$$

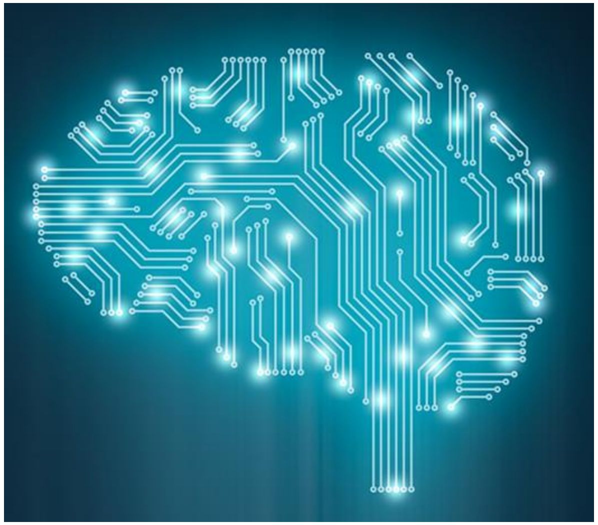
$$\phi_1 - \psi_{12} = \arcsin\left(\frac{1}{\alpha}\sin\psi_{12}\right) \tag{3}$$

$$C_{123} = \psi_{12} + \psi_{23} + \psi_{31} = \frac{1}{2}(\phi_1 + \phi_2 + \phi_3) = 0 \tag{4}$$

MANCHESTER
1824
The University of Manchester

Machine learning

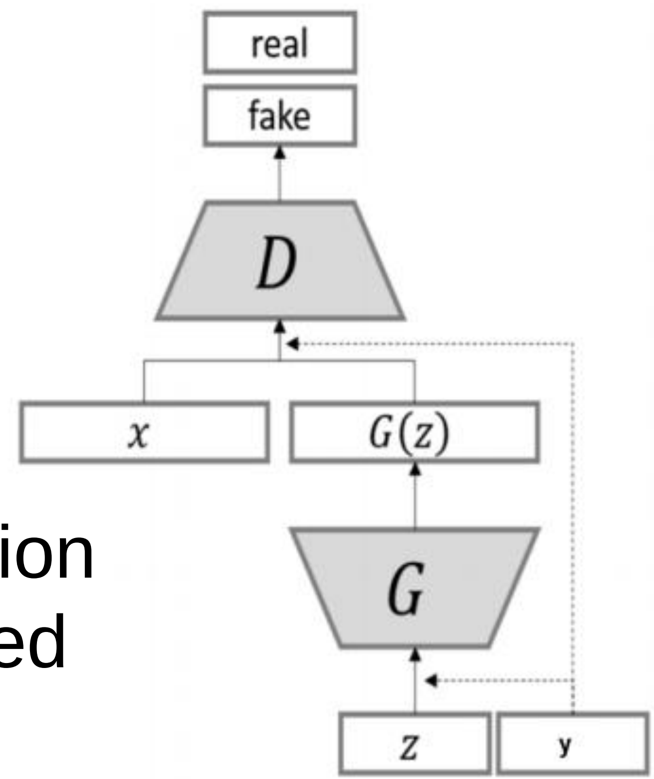
- Enables the computer to learn from data without explicit programming to do so (Arthur Samuel)
- Uses:
  - Image processing
  - Predictive models
  - Speech recognition
  - Medical diagnostics
  - Product recommendations



Conditional Generative Adversarial Networks (cGANs)

- GANs train a generator (G) against a discriminator (D)
- Loss function automatically altered for each data set

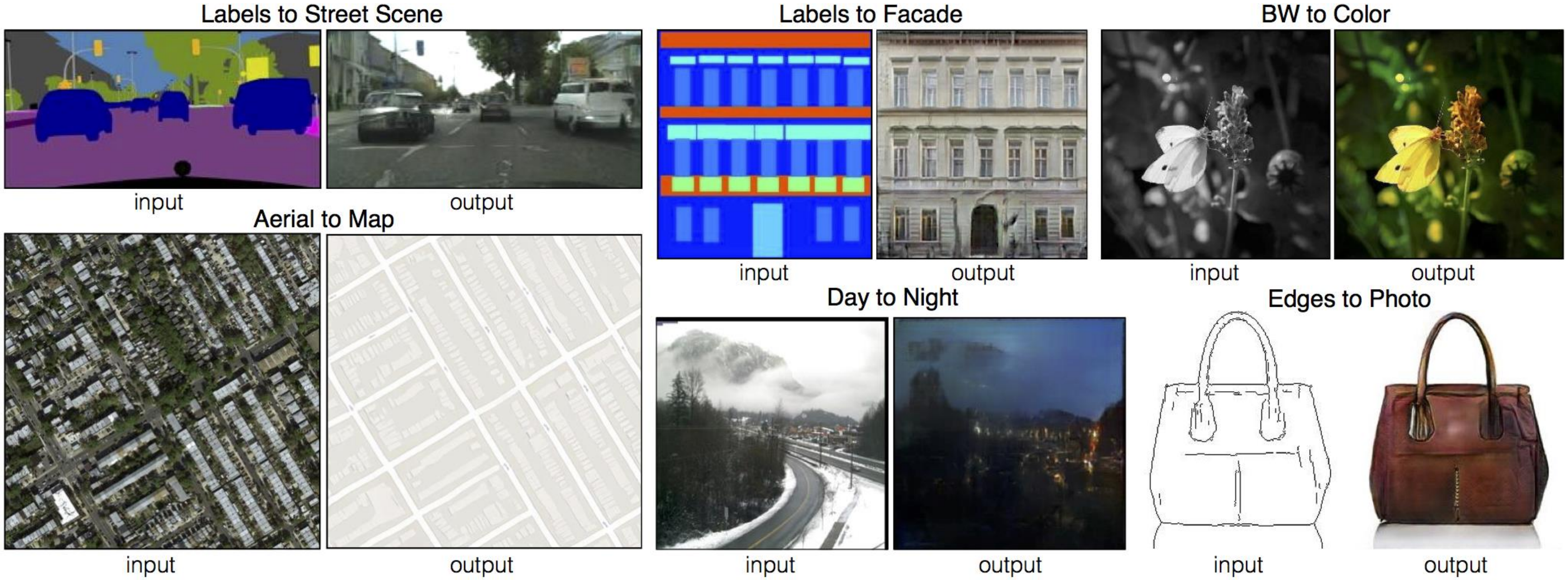
- cGANs have extra information about the images (y) inputted to G and D



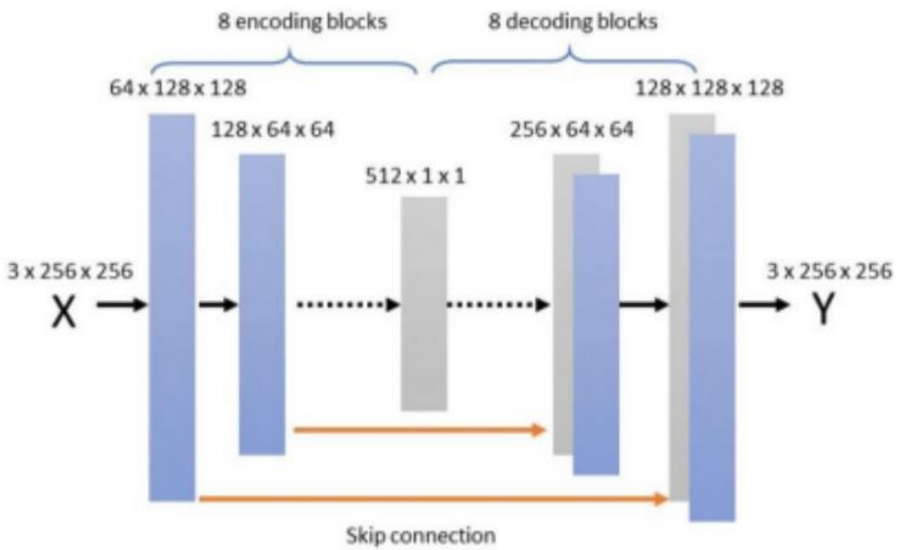
The University of Manchester

pix2pix algorithm

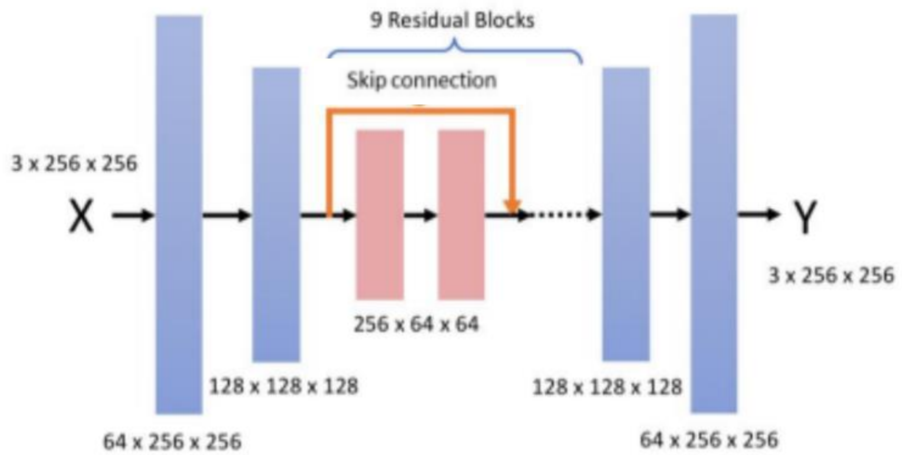
- Created by Isola et al. (2017) using a cGAN.
- The algorithm must also minimise the L1 loss.



Architectures of pix2pix



U-Net architecture

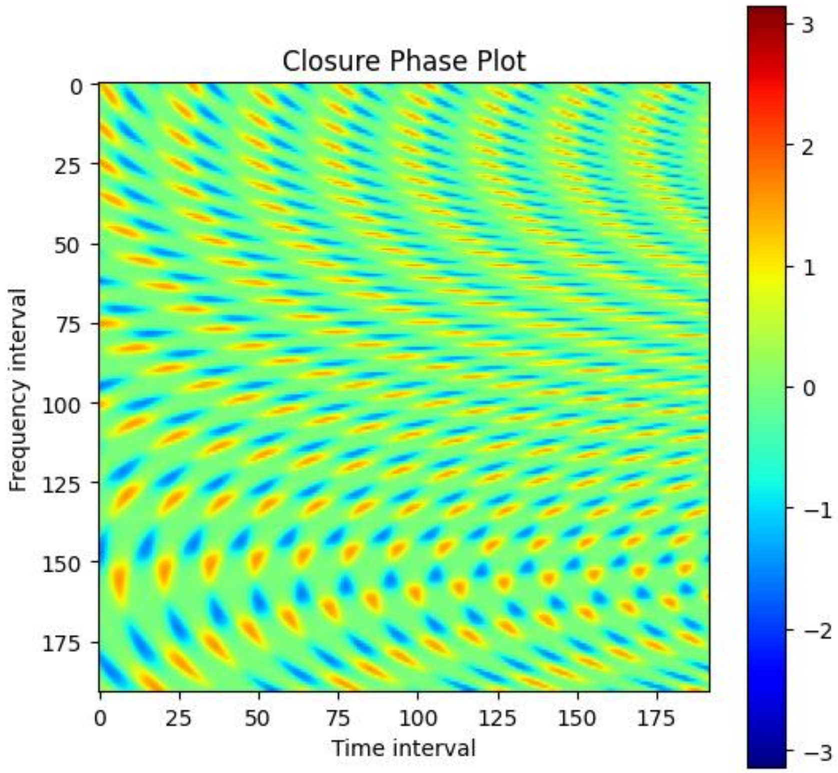


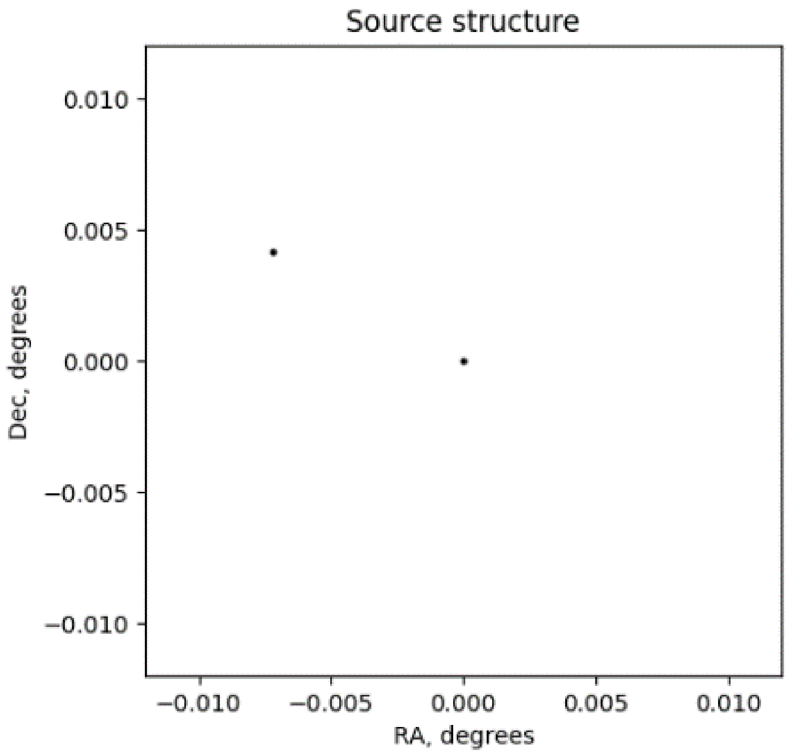
ResNet architecture

MANCHESTER 1824

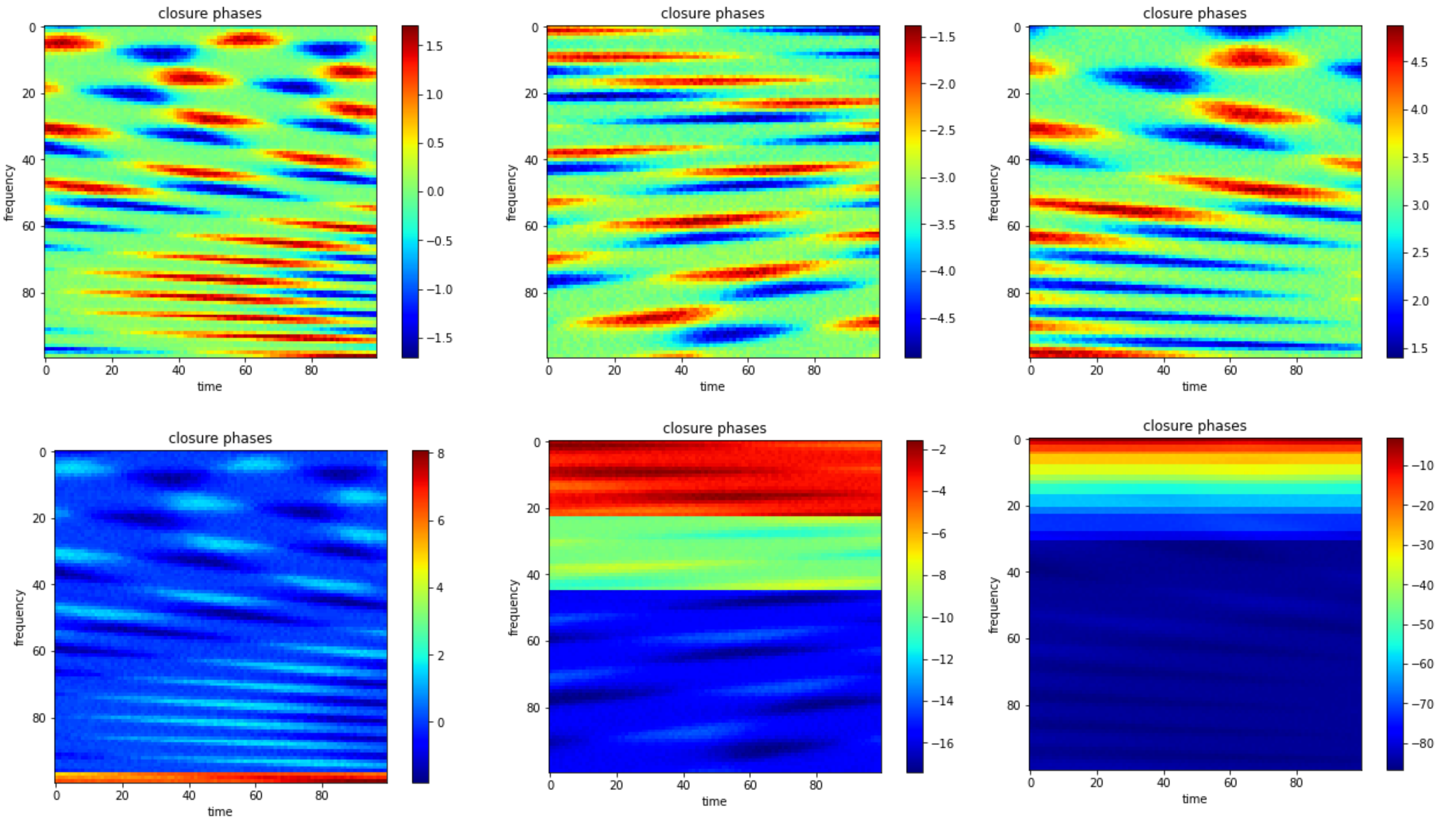
The University of Manchester

Image generation



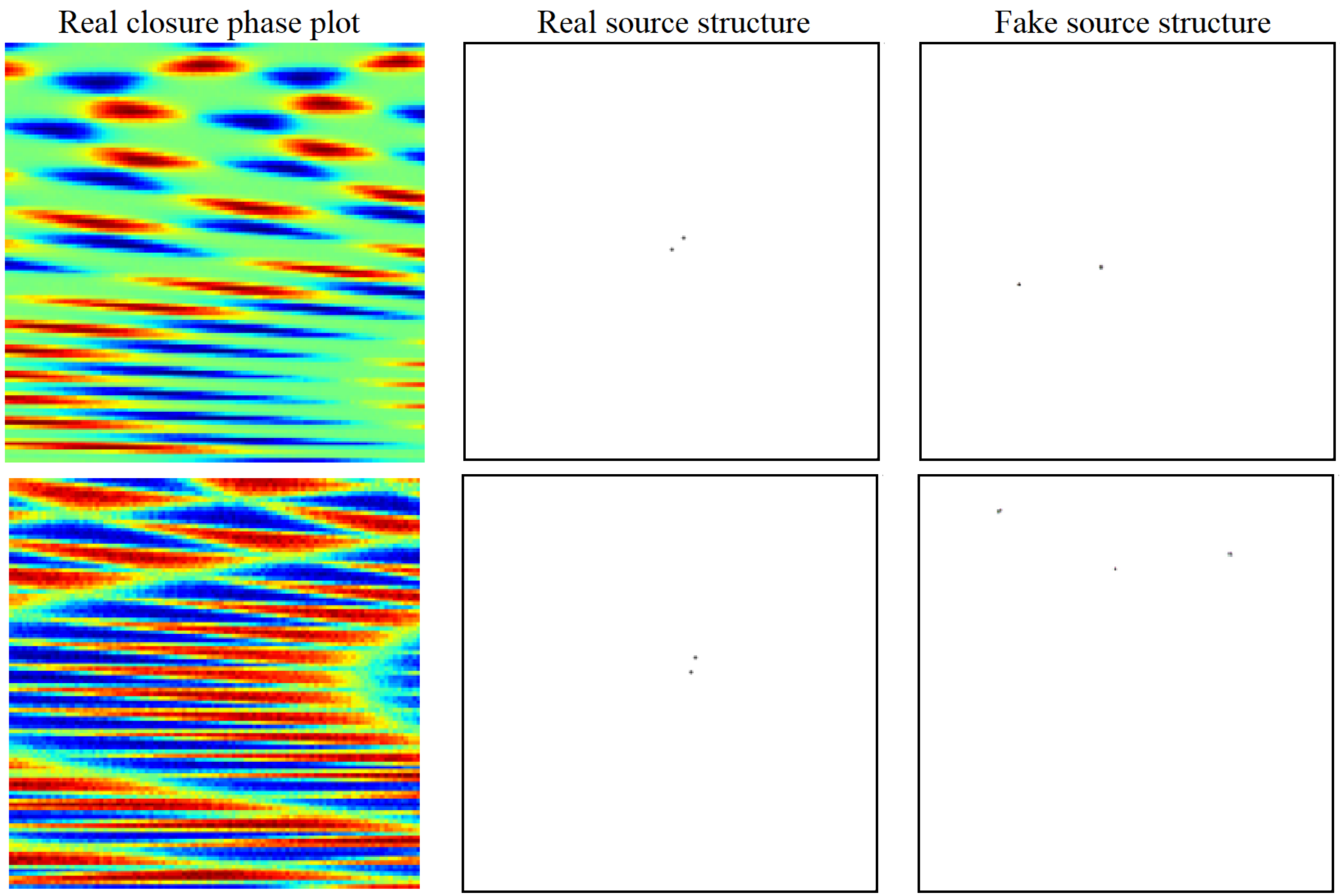


Pure python: image generation



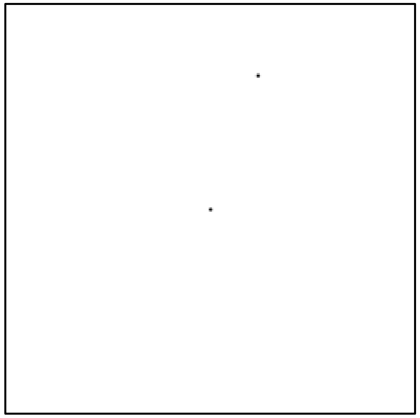
MANCHESTER 1824
The University of Manchester

Pure python: initial machine learning trial

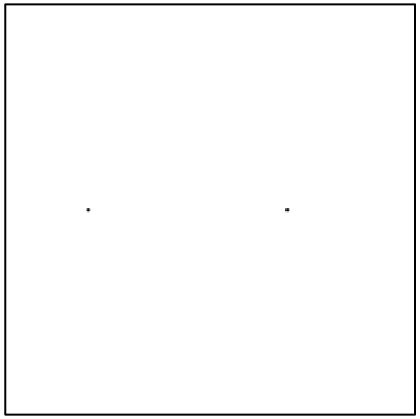


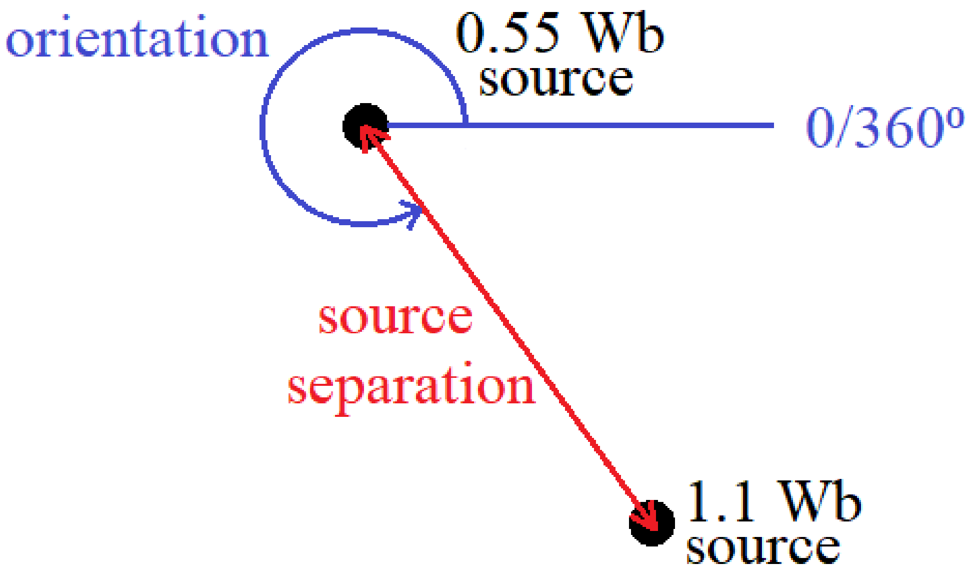
Defining orientation and separation

Varying orientation



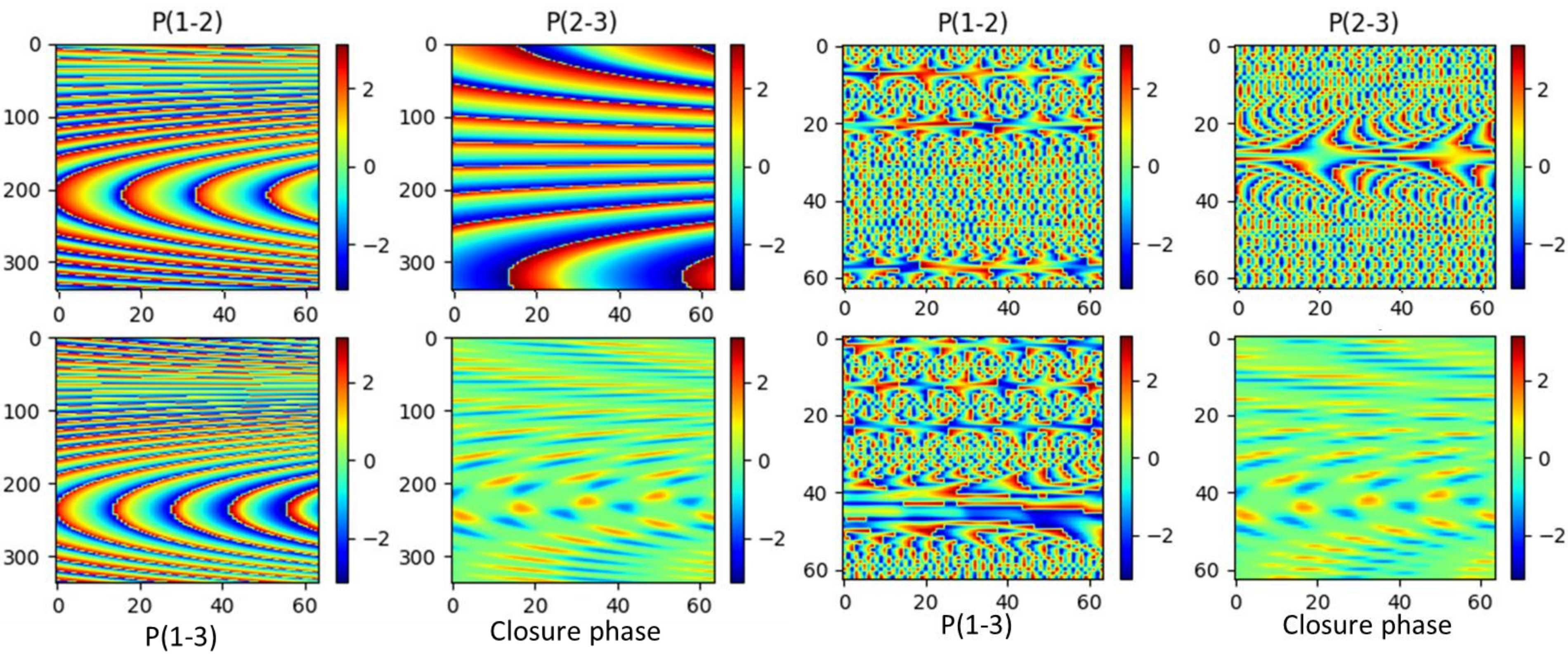
Varying separation





MANCHESTER 1824
The University of Manchester

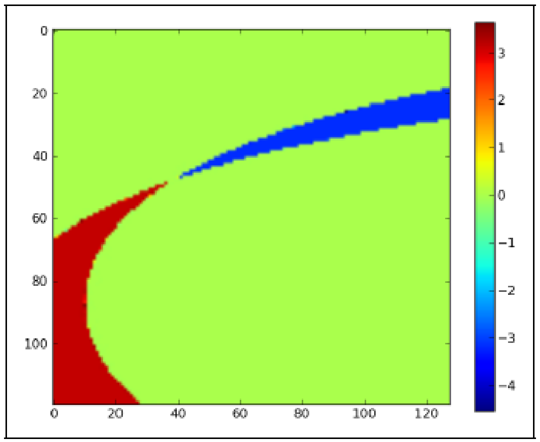
Python with CASA: image generation



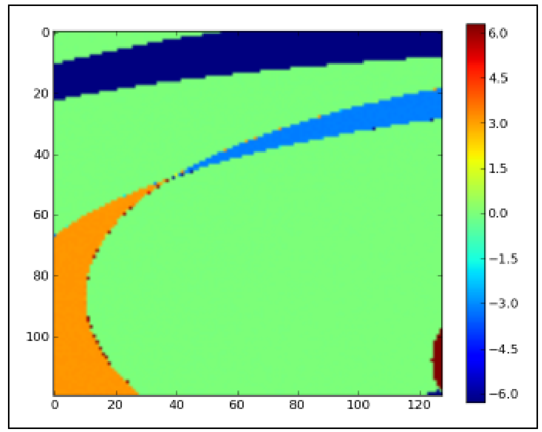
10 arcseconds from centre

510 arcseconds from centre

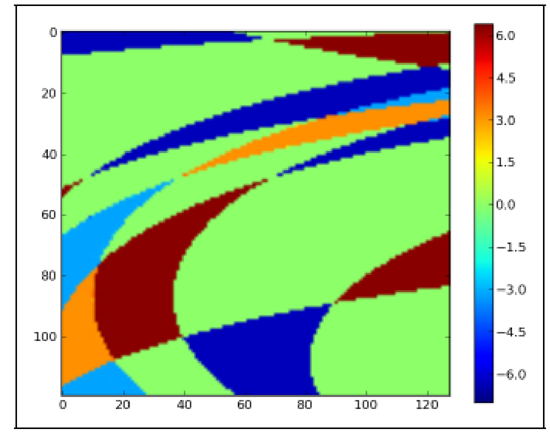
Python with CASA: image generation



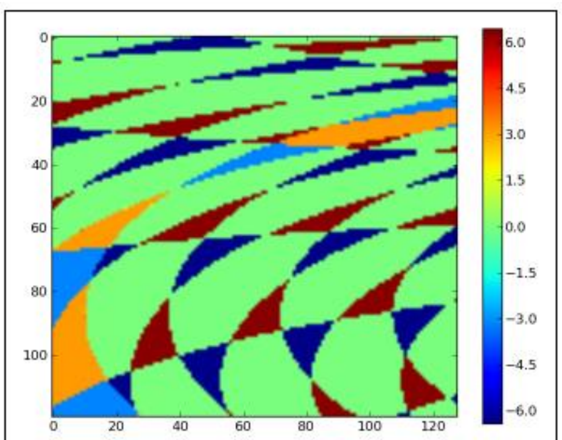
0 arcseconds



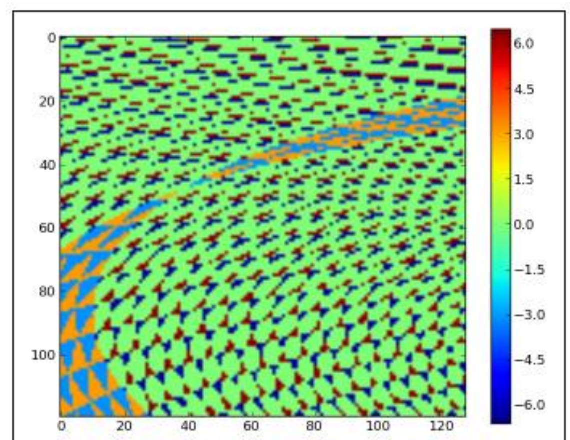
3 arcseconds



6 arcseconds



20 arcseconds



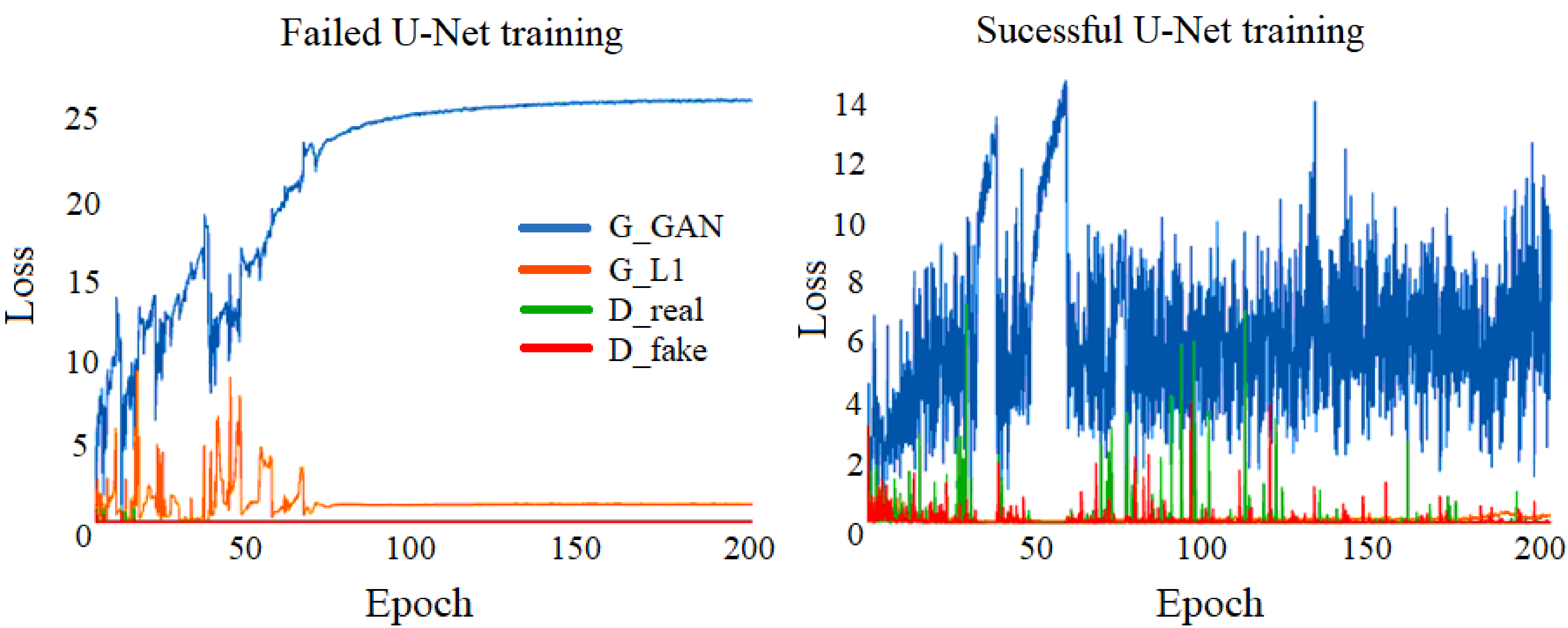
100 arcseconds

MANCHESTER 1824
The University of Manchester

Python with CASA: Parallelisation

- Production time for one image is 90 seconds
- Server has 96 cores
- Predict 10,000 images should take 8 hours 40 minutes using 35 cores
- Actually took 69 hours and 10 minutes
- CASA has inherent parallelisation

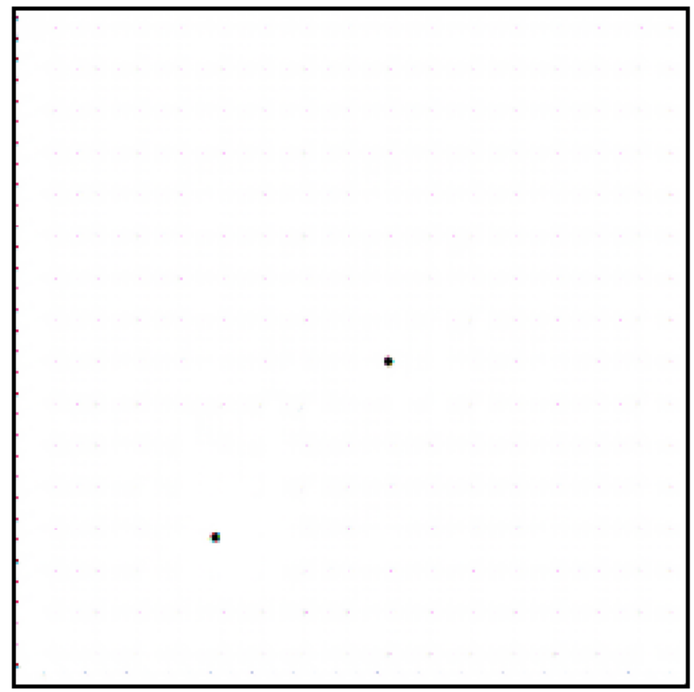
Monitoring loss



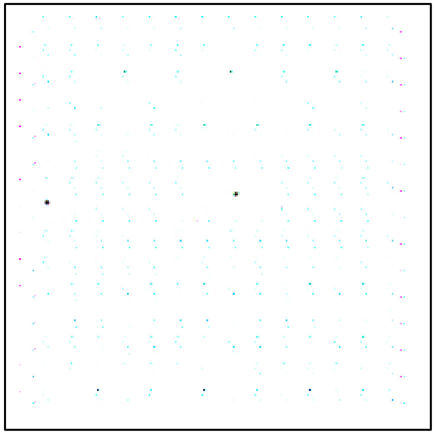
Monitoring loss

Failed U-Net training

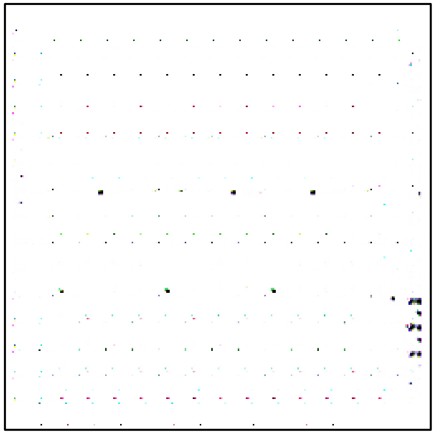
Sucessful U-Net training



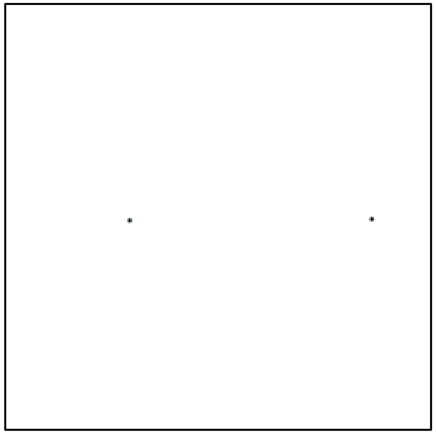
Source separation - Results



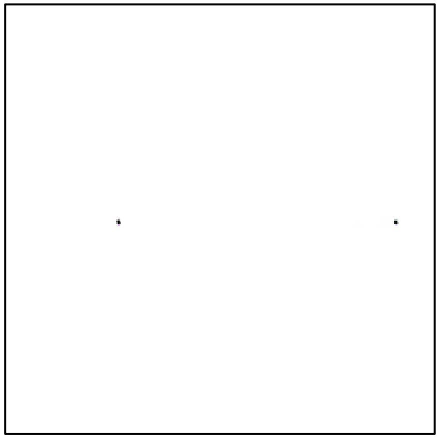
U-Net 256



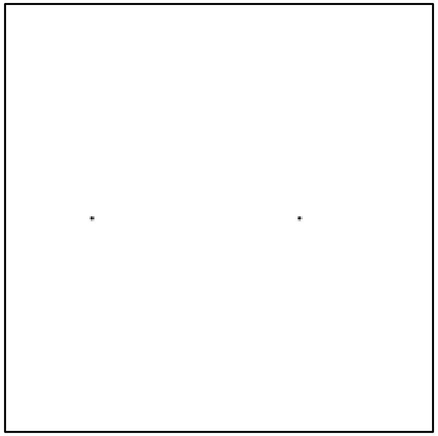
U-Net 128



ResNet 9 blocks



ResNet 6 blocks



Real source structure

MANCHESTER 1824
The University of Manchester

Source separation - Results

Generator Architecture	Correct Structure (%)	Average number of sources	Average difference in source separation (arcsec)	Average percentage difference in source separation (%)
U-Net 256	83.3	2.27	3.13	16.3
U-Net 128	0	38.93	-	-
ResNet 9 blocks	86.7	1.97	7.75	30.2
ResNet 6 blocks	83.3	2.03	5.07	21.9

The University of Manchester

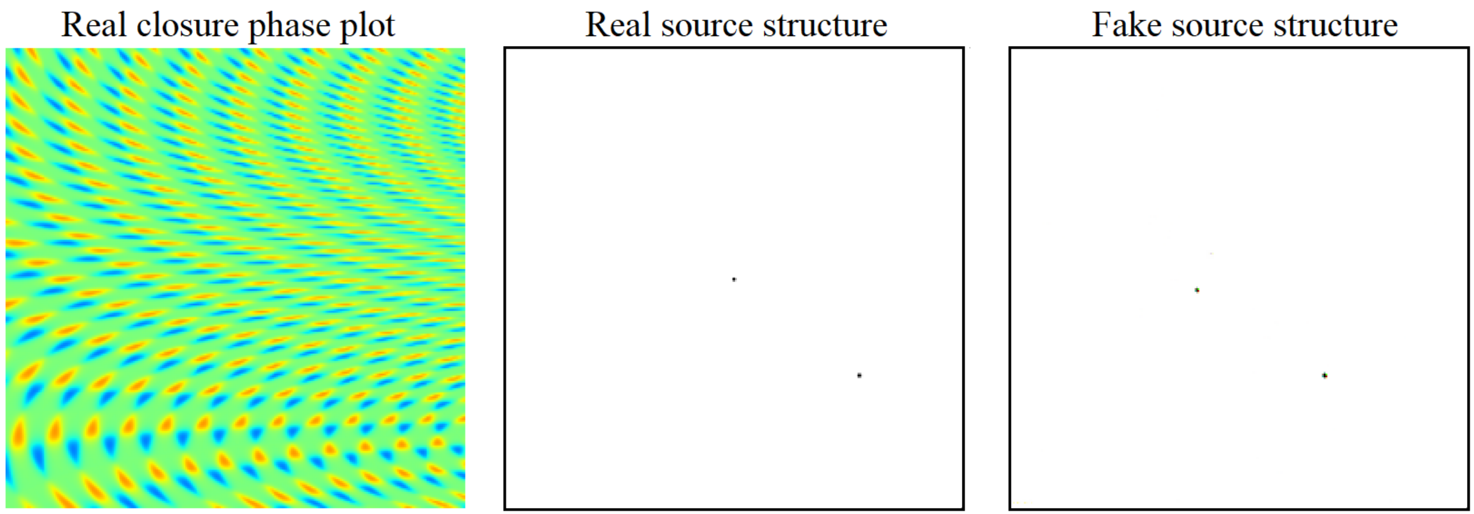
Orientation results

Generator architecture	Epochs	Average source number	Proportion with 2 sources (%)	Average source separation (arcsecs)	Average difference in orientation (^{o})
ResNet 9 blocks	200	2.39	26	29.4	45.6
ResNet 9 blocks	400	7.62	0	-	-
U-Net 256	200	2.08	92	35.9	24.9
U-Net 256	400	2.4	68	35.3	32.2
ResNet 6 blocks	200	1.64	28	36.4	49.8
U-Net 128	200	2.72	32	40.2	50.7

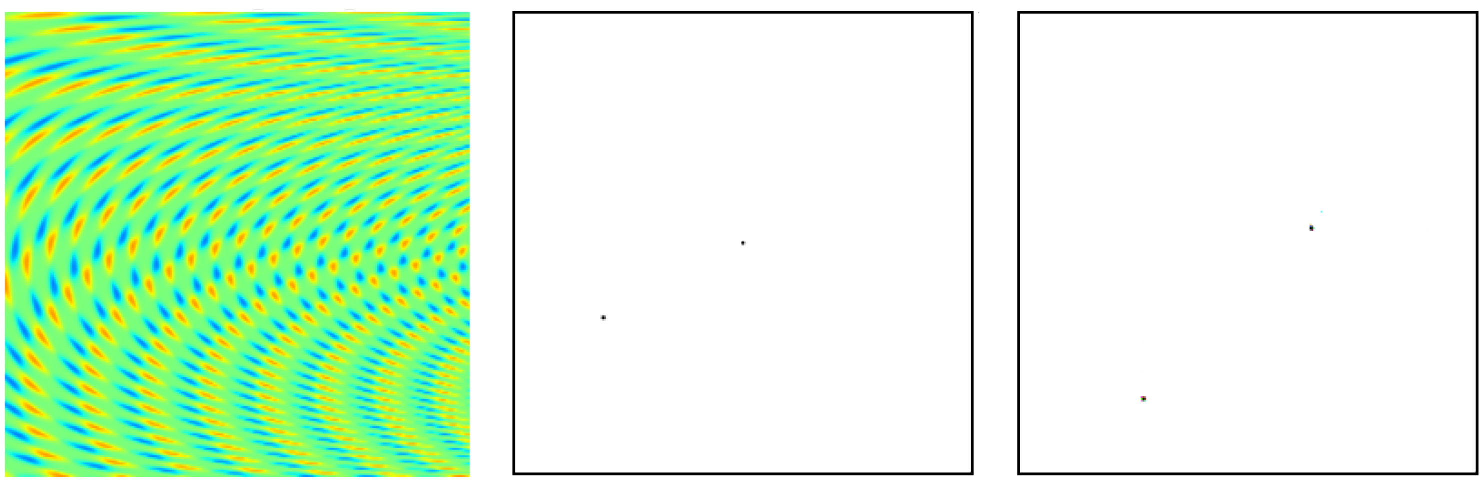
- 70x70 PatchGAN discriminator architecture
- 9,899 training images, 50 test images

Examples of best orientation results

ResNet 9 blocks
70x70 PatchGAN
200 epochs

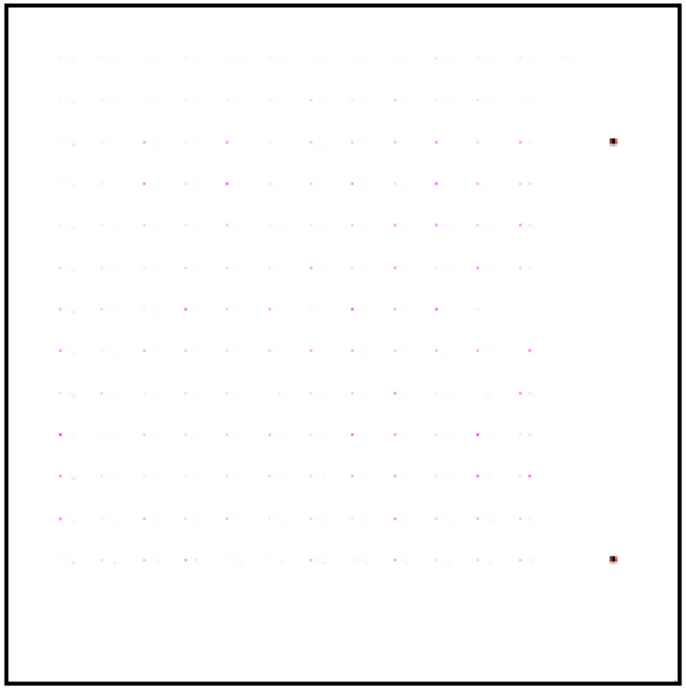


U-Net 256
70x70 PatchGAN
200 epochs

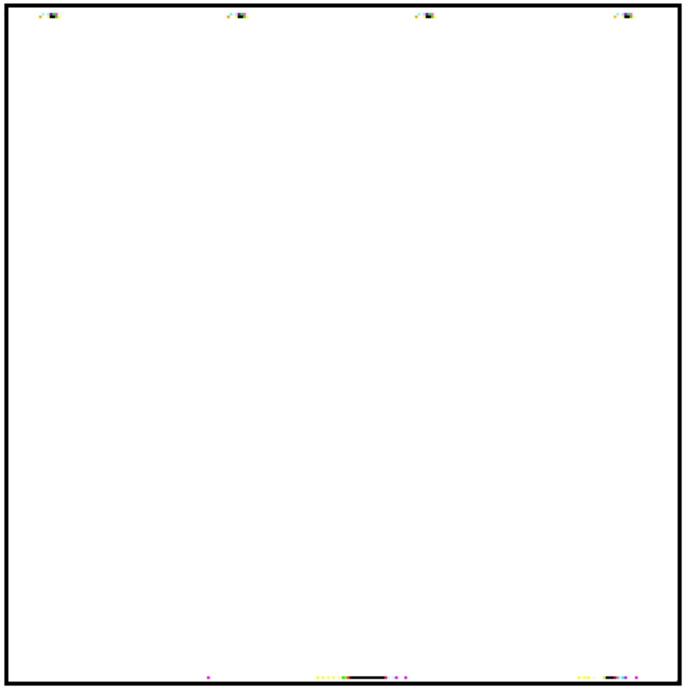


Orientation: Examples of overfitting

U-Net 256, 400 epochs

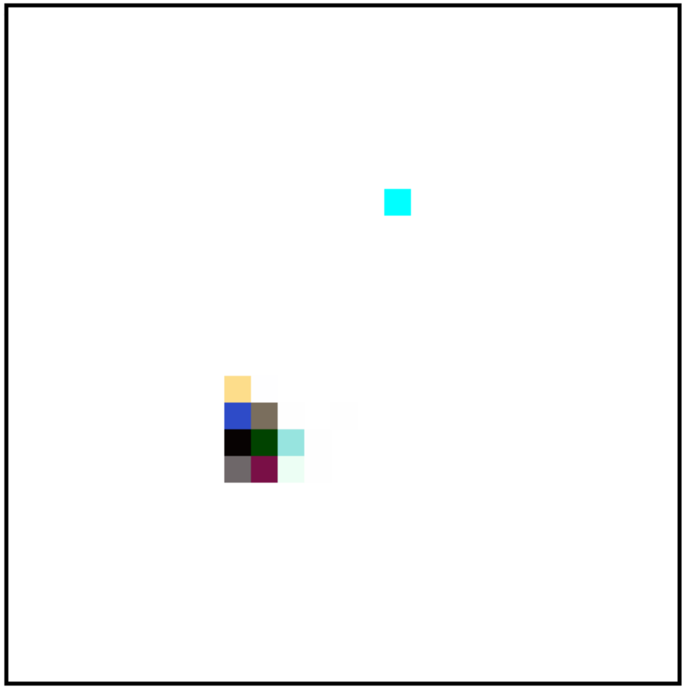


ResNet 9 blocks, 400 epochs

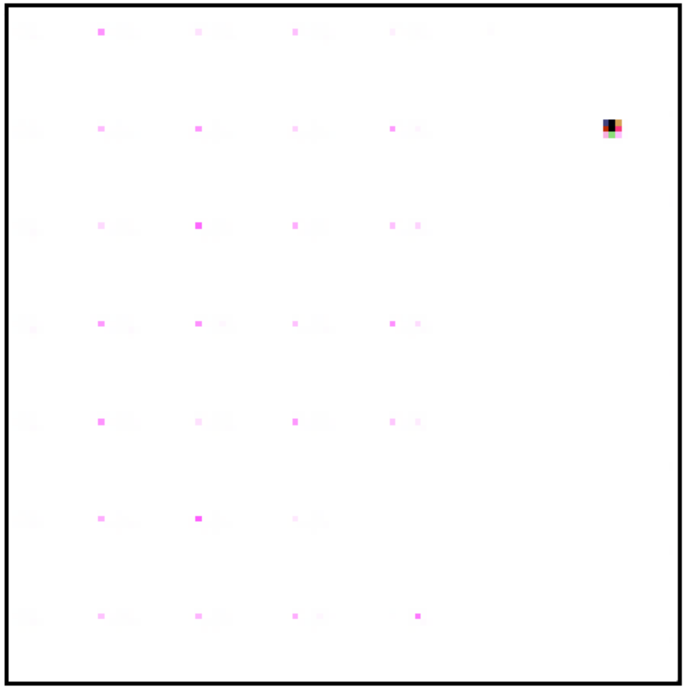


Errors seen in orientation images

U-Net 256, 200 epochs

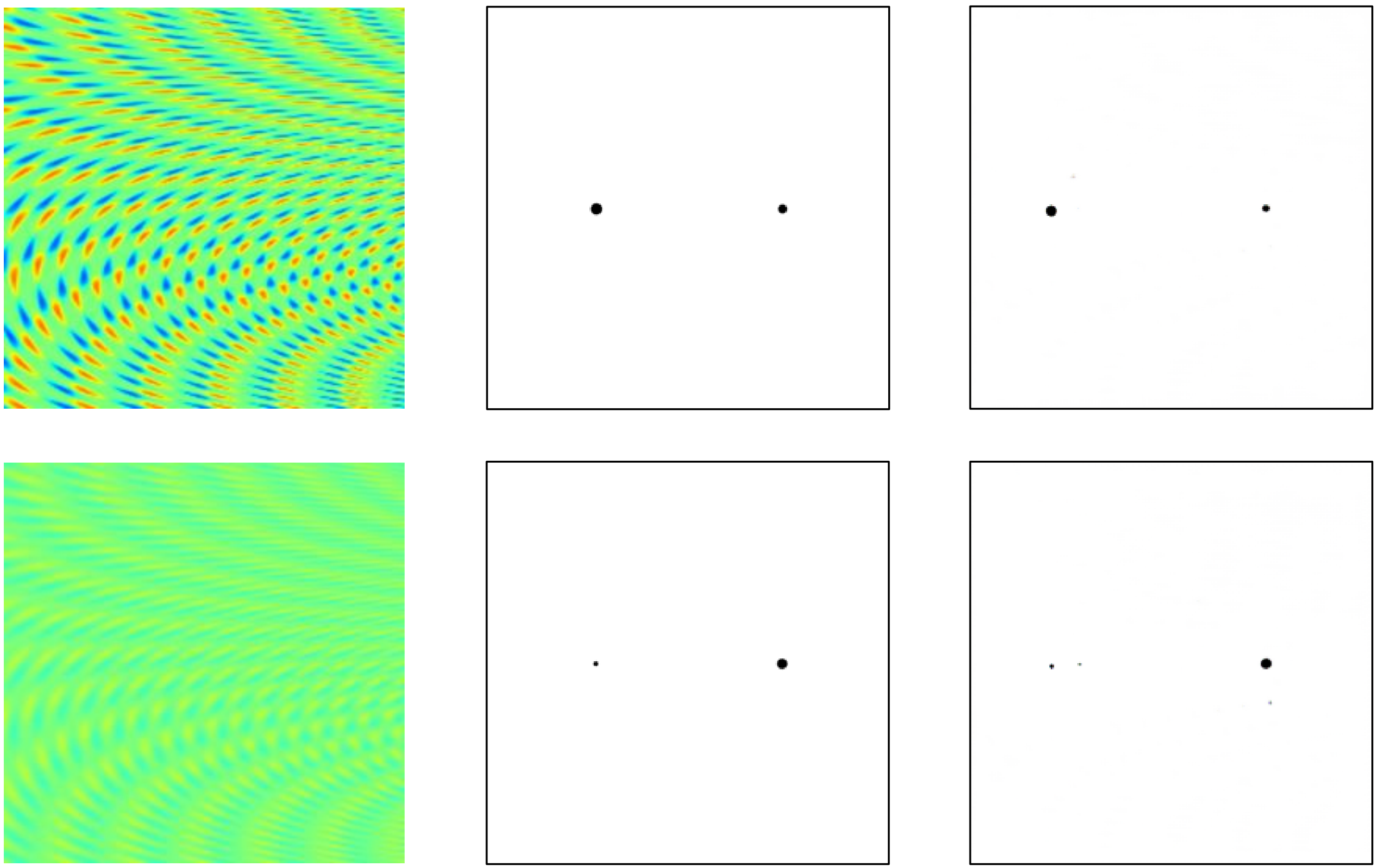


U-Net 256, 400 epochs



Flux

Real Closure Phase Plot | Real source structure | Fake source structure



MANCHESTER 1824
The University of Manchester

Further improvements

- Improve source detection code
- Train to model multiple closure phase variables
- Train using noisier data
- Training with plots produced from different numbers of sources
- Investigate Variational Autoencoders (VAEs)

Conclusion

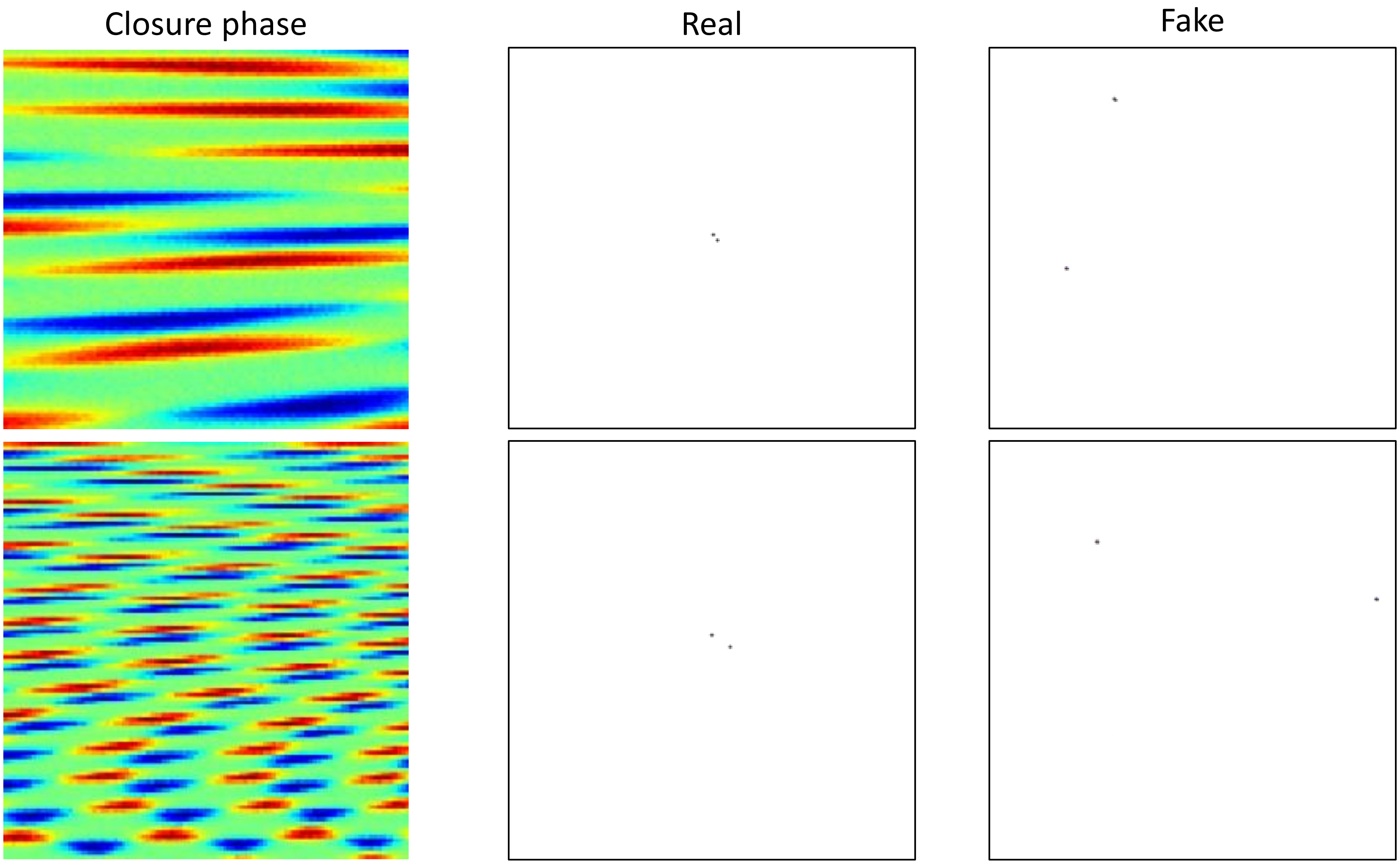
- Lots of future potential
- Saves computational power
- U-Net for varying parameters
- ResNet for constant parameters
- More work before real data

Any questions?

Special thanks to:

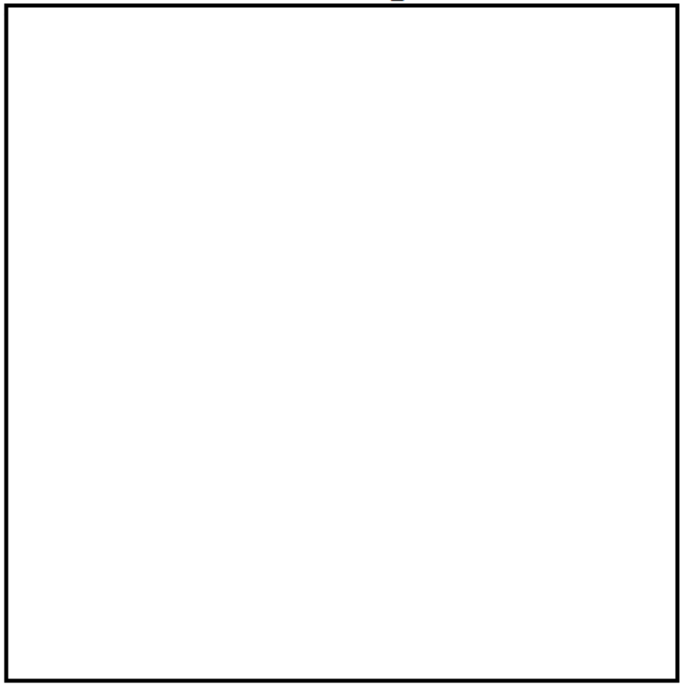
- Dr Neal Jackson
- Dr Philippa Hartley

Python CP generation

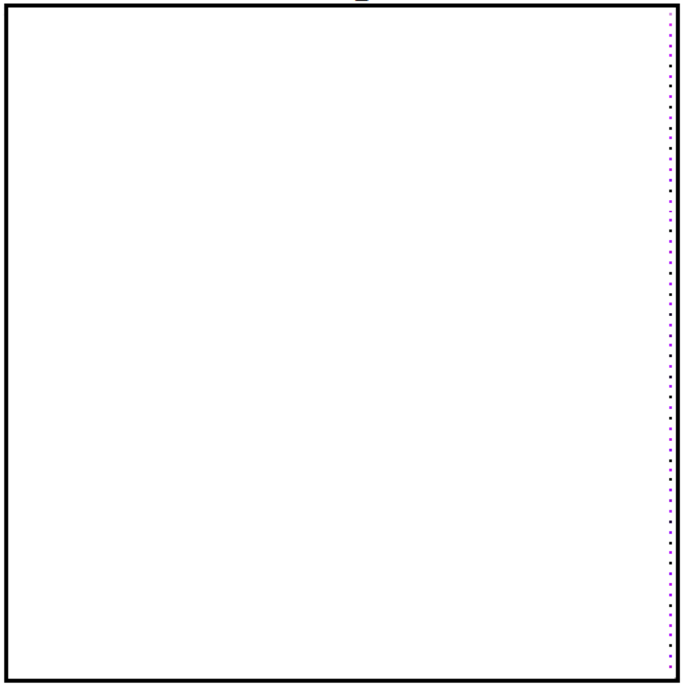


The pixel PatchGAN

ResNet, pixel



U-Net, pixel



The University of Manchester

Loss functions

$$\mathcal{L}_{\mathrm{cGAN}}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log D(x, G(x, z))] \quad (1)$$

$$\mathcal{L}_{\mathrm{L1}}(G) = \mathbb{E}_{\mathrm{x,y,z}}[|x - G(y, z)|] \quad (2)$$

Source separation dataset

70x70
3 layers

Pixel

70x70
6 layers

Real

The University of Manchester

UV plane comparison