

# D-scan Documentation

## ***Release 1***

**will louth, Rielly Newbold**[B[D

Dec 27, 2019



---

## Contents

---

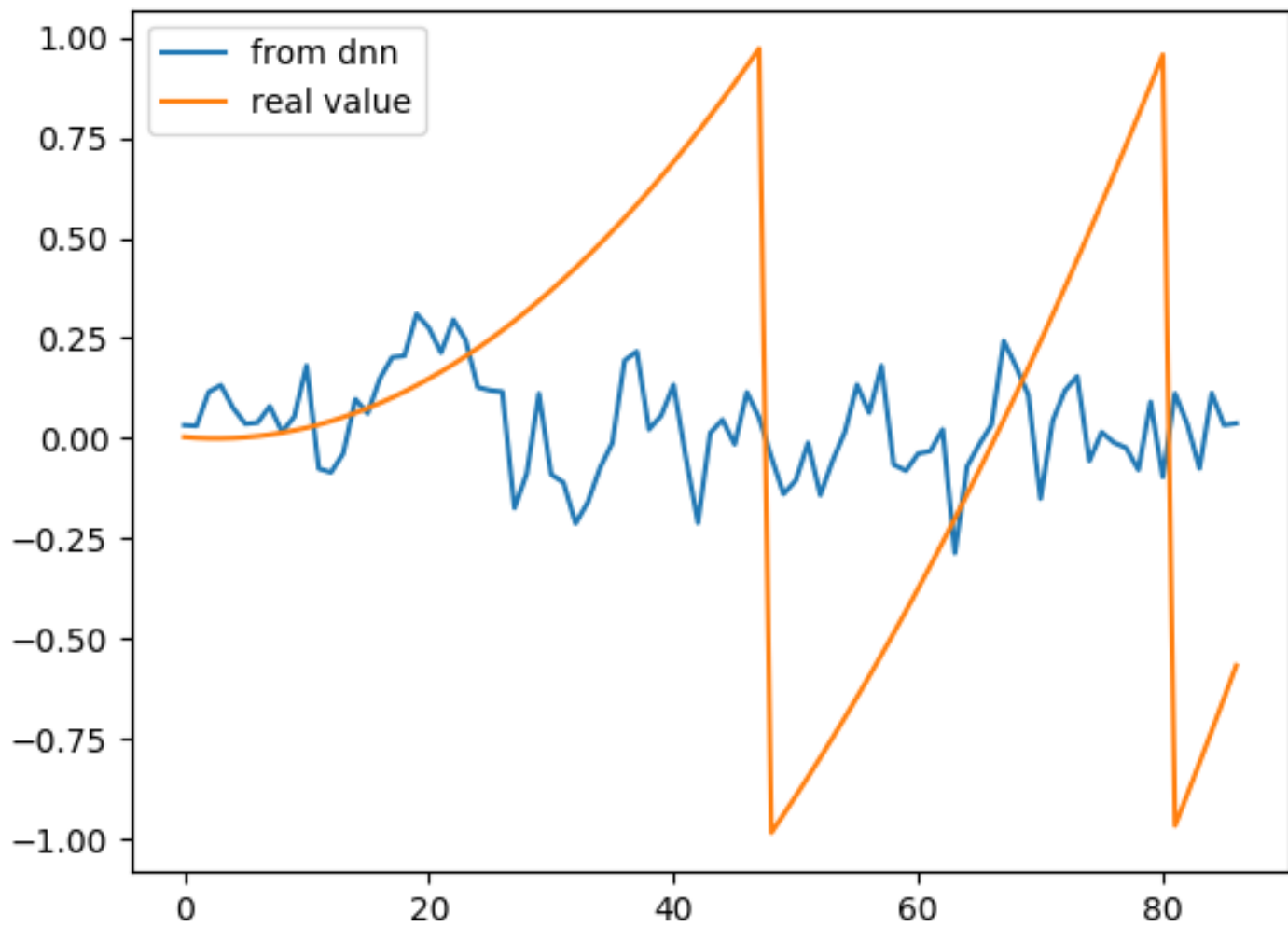
<b>1</b>	<b>Graphs</b>	<b>1</b>
1.1	first predictions . . . . .	1
1.2	maxpool predictions . . . . .	2
1.2.1	model . . . . .	4
1.2.2	training fit . . . . .	5
1.2.3	testing fit . . . . .	6
1.2.4	discussion . . . . .	6
<b>2</b>	<b>D_scan_creation</b>	<b>7</b>
2.1	pulse_class . . . . .	7
2.2	__init__ . . . . .	7
2.3	my_compress . . . . .	8
2.4	save_instance . . . . .	8
2.5	h5out . . . . .	8
2.6	make_dscan_all . . . . .	8
2.7	make_dsca_1z . . . . .	8
2.8	make_E_omega_plot . . . . .	8
2.9	make_E_omega_plot2 . . . . .	8
2.10	make_I_labda_plot . . . . .	9
2.11	make_E_t_plot . . . . .	9
2.12	make_E_omega_plot . . . . .	9
2.13	make_E_omega_plot2 . . . . .	9
2.14	make_I_t_plot . . . . .	9
2.15	make_plots . . . . .	9
2.16	D_trace . . . . .	9
2.17	E_gaussian . . . . .	10
2.18	E_omega_gen . . . . .	10
2.19	make_phi . . . . .	10
2.20	make_psi . . . . .	10
2.21	refractive_index . . . . .	11
2.22	D_scan . . . . .	11
<b>3</b>	<b>dnn_learn</b>	<b>13</b>
<b>4</b>	<b>dnn_predict</b>	<b>15</b>
<b>5</b>	<b>dnn_model</b>	<b>17</b>

<b>6</b>	<b>main</b>	<b>19</b>
<b>7</b>	<b>data parsing</b>	<b>21</b>
7.1	hdf5_check . . . . .	21
7.2	exp_dataio . . . . .	21
<b>8</b>	<b>about</b>	<b>23</b>
	<b>Index</b>	<b>29</b>

a timeline of predictions

### 1.1 first predections

the model used for the predictions. There was a problem with the input pipeline that mean that the input and output data that was used to train the network were not aligned. This meant that the model learnt the average of the data points and was not very good.

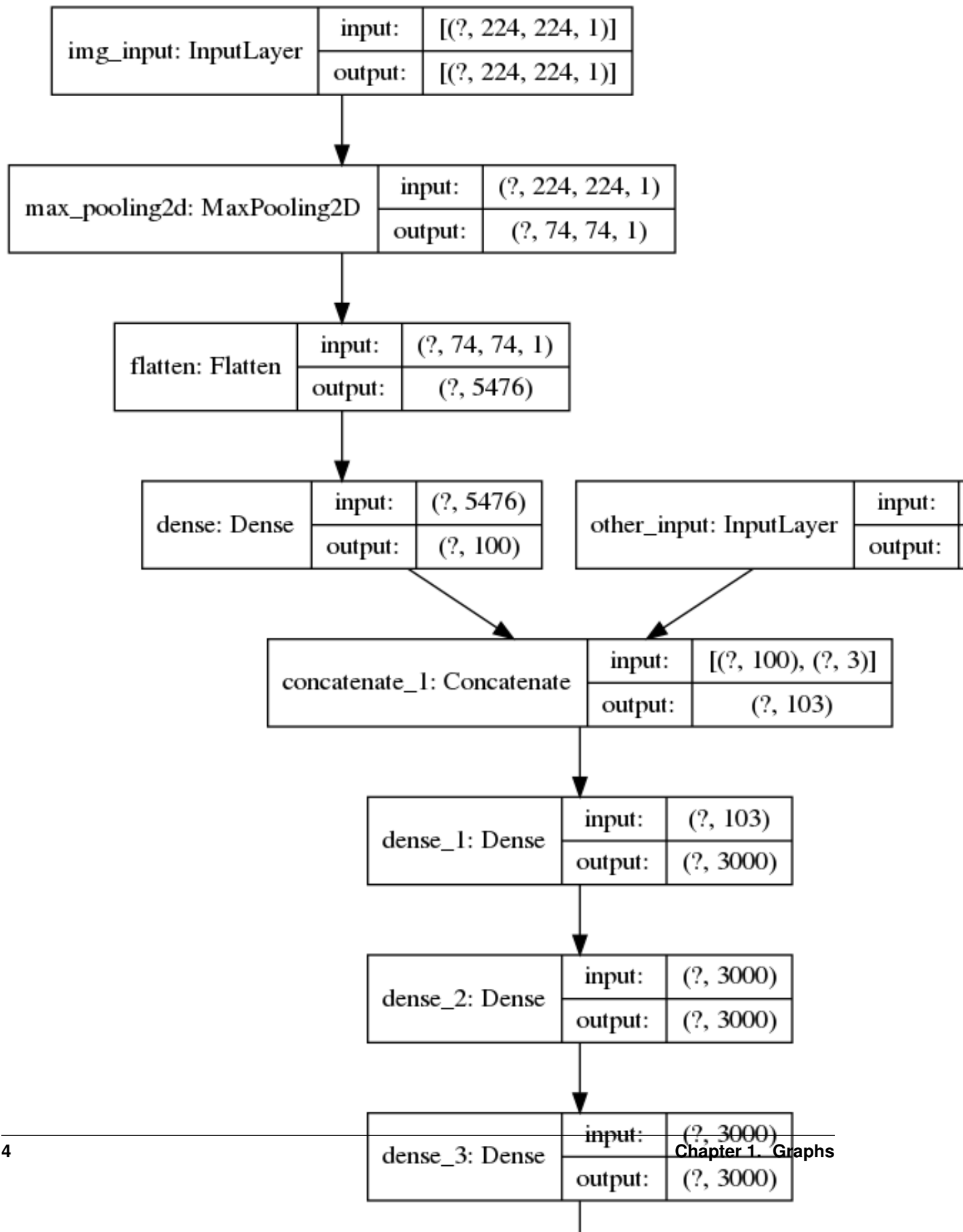


## 1.2 maxpool predections

The model used for these predictions was very simple and was used as a very basic baseline to compare other models from. This has been run on pulses with random quadratic phase profiles. There was no noise in the data.

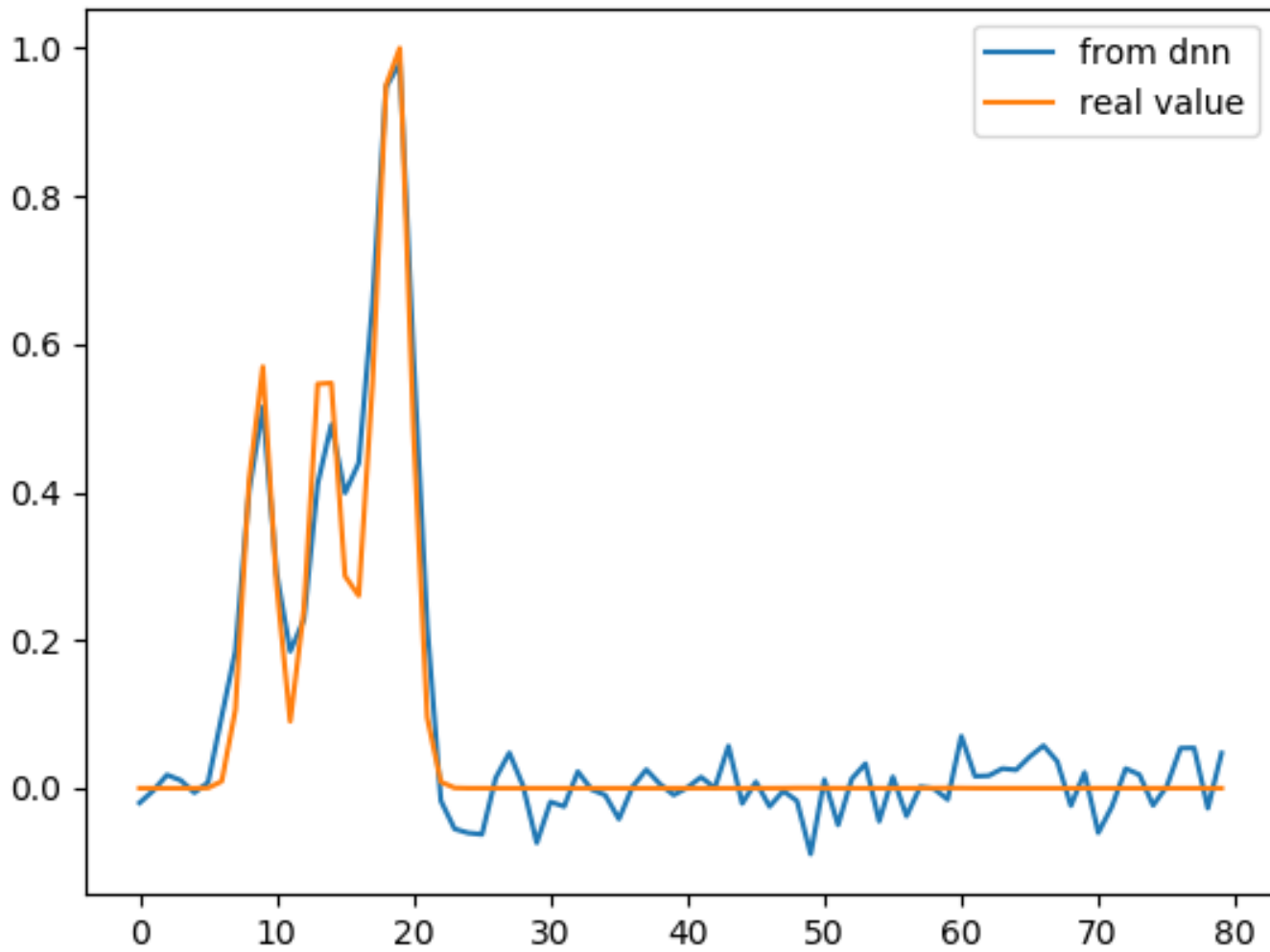


1.2.1 model

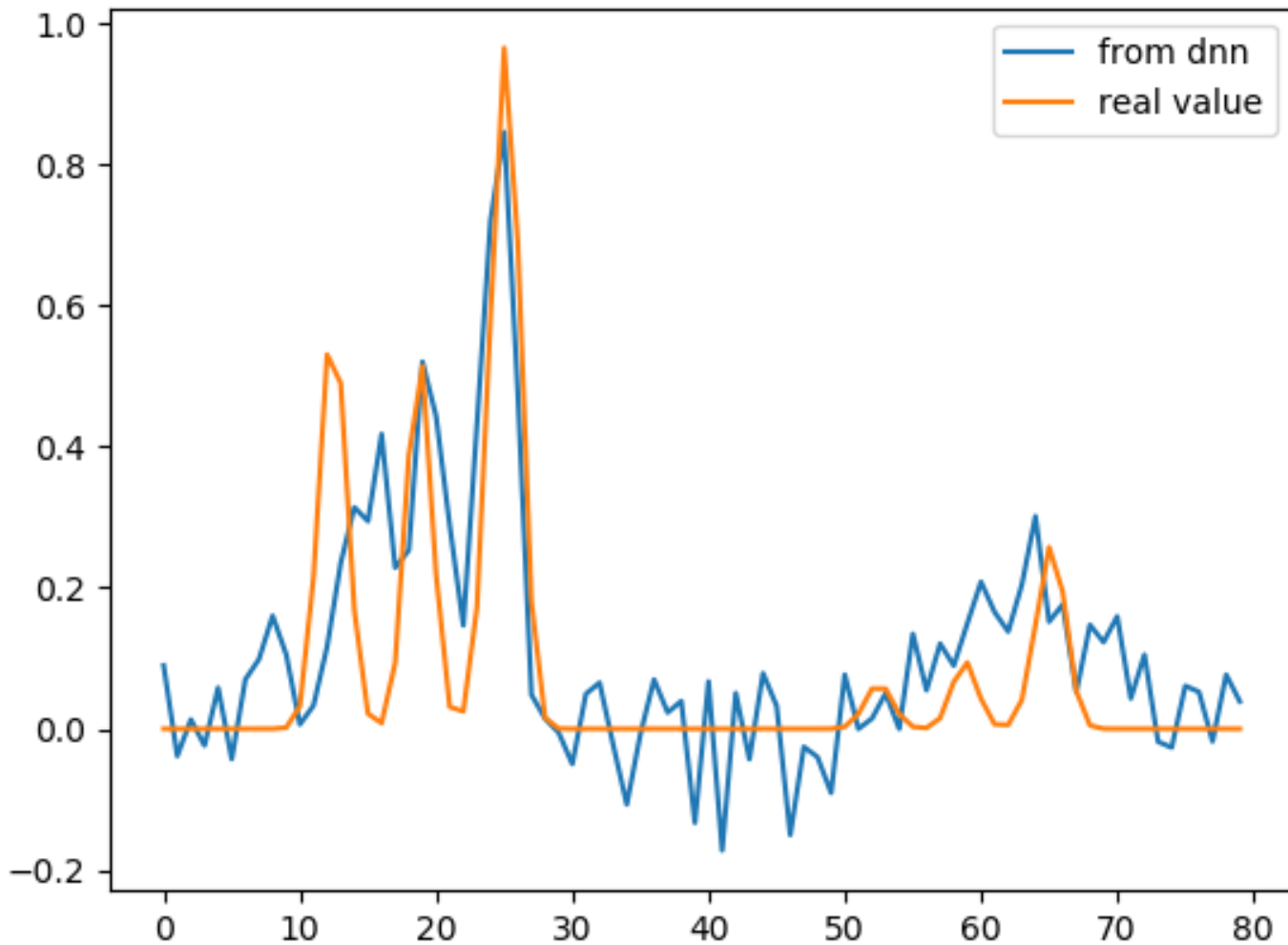




### 1.2.2 training fit



### 1.2.3 testing fit



### 1.2.4 discussion

There is a good fitting for the training data however on the testing data there is quite a poor fitting. This tends to happen when there is overfitting of the data. With such a simple model this happens quite quickly however the simplicity of the model also means that it is unable to distinguish between different dscans very well and as such will not generalise well to new pulses.

### 2.1 pulse\_class

```
class pulse_class
```

### 2.2 \_\_init\_\_

```
__init__(self, control, steps=1024, start=-256, stop=256, E_0=1, omega_0=10, delta=3, glass_dis=5e5,
         phase_control = "none")
```

the initialisation function.

#### Parameters

- **control** ("E\_omega" or "E\_t") – Takes the values either “E\_omega” or “E\_t”. It controls whether the pulse is created from a time or frequency space description.
- **steps** – The number of steps in time. Used to define the time axis.
- **start** – The start time (fs)
- **stop** – The stop time (fs)
- **E\_0** – The maximum amplitude of the electric field.
- **omega\_0** – The central frequency of the pulse
- **delta** – If control is “E\_t” it is the width of the gaussian in the time axis.
- **glass\_dis** – The maximum insertion distance of the glass in nm
- **phase\_control** ("none" or "phi\_quadratic" or "phi\_cubic") – “phi\_quadratic” gives a random quadratic phase in frequency space and “phi\_cubic” gives a random cubic phase in frequency space. “none” results in no phase being added.

**Return type** None

## 2.3 my\_compress

`my_compress(self, in_a, compression_ratio_x, compression_ratio_y)`

A custom made compression function that takes the average of the nearest points for a 2-D a

**Parameters**

- **in\_a** – the array that is to be compressed.
- **compression\_ratio\_x** – the factor by which the x axis is to be compressed by.
- **compression\_ratio\_y** – the factor by which the y axis is to be compressed by.

**Return type** compressed array

## 2.4 save\_instance

`save_instance(self)`

## 2.5 h5out

`h5out(self, arg1)`

Saves the compressed pulse into an hdf5 file in the pulses folder.

**Parameters** **arg1** – The sub-folder in which the pulse is to be saved in the pulses folder.

## 2.6 make\_dscan\_all

`make_dscan_all(self)`

creates a pcolor or imshow plot of the compressed dscan trace.

## 2.7 make\_dsca\_1z

`make_dsca_1z(self)`

## 2.8 make\_E\_omega\_plot

`make_E_omega_plot(self)`

## 2.9 make\_E\_omega\_plot2

`make_I_omega_plot(self)`

## 2.10 make\_I\_labda\_plot

`make_I_labda_plot (self)`

## 2.11 make\_E\_t\_plot

`make_E_t_plot (self)`

## 2.12 make\_E\_omega\_plot

`make_E_omega_plot (self)`

## 2.13 make\_E\_omega\_plot2

`make_E_omega_plot2 (self)`

makes a plot of E as a function of omega but sliced above omega = 2 rad/fs

## 2.14 make\_I\_t\_plot

`make_I_t_plot (self)`

## 2.15 make\_plots

`make_plots (self, control="all")`

a control function for calling the other plot functions.

**Parameters** `control` ("all" or "dscan") – either generates all the plots or just the dscan.

## 2.16 D\_trace

`D_trace (self, E_omega, k, z, r)`

**Parameters**

- `E_omega` –
- `k` – Wavevector
- `z` – Glass insertion distance
- `r` – Response function

**Return type** the dscan at a single glass insertion.

## 2.17 E\_gaussian

**E\_gaussian** (*self*, *t*, *E\_0*, *omega\_0*, *delta\_t*)

**Parameters**

- **t** –
- **E\_0** –
- **omega\_0** –
- **delta\_t** –

**Return type** E, A gaussian distributed(in time) E field

## 2.18 E\_omega\_gen

**E\_omega\_gen** (*self*, *delta\_omega*=3, *omega\_cent*=2, *E\_0*=1)

**Parameters**

- **delta\_omega** –
- **omega\_cent** –
- **E\_0** –

**Return type** E\_omega a gaussian distributed(in omega) E field

## 2.19 make\_phi

**make\_phi** (*self*, *control*, *centred* = 2, *phi\_slope*=1.0)

**Parameters**

- **control** ("phi\_quadratic" or "phi\_cubic") –
- **centred** – the central frequency of either the cubic or quadratic function.
- **phi\_slope** – the gradient of the cubic or quadratic function

**Return type** phi, The phase of the Electric field in omega.

## 2.20 make\_psi

**make\_psi** (*self*, *control*, *t\_centred*=0, *psi\_slope*=1)

**Parameters**

- **control** ("psi\_quadratic" or "psi\_cubic") –
- **centred** – the central frequency of either the cubic or quadratic function.
- **psi\_slope** – the gradient of the cubic or quadratic function

**Return type** psi, The phase of the Electric field in time.

## 2.21 refractive\_index

**refractive\_index** (*self*, *wavelength*)

Parameters **wavelength** –

## 2.22 D\_scan

**D\_scan** (*self*, *E\_omega*, *omega*, *glass\_diss*)

Parameters

- **E\_omega** –
- **omega** –
- **glass\_diss** –





The file responsible for the learning of the model.

**dnn\_learn()**

**Parameters**

- **b\_n** – batch number
- **ep** – epoch number
- **s\_p\_e** – steps per epoch
- **validate** – bool that controls if validation is run every epoch and learning graph produced.
- **d** – tensorflow dataset containing glob of pulse file locations
- **input\_dscan** – tensorflow dataset containing the dscan info from each pulse
- **input\_other** – tensorflow dataset containing the other input parameters info from each pulse
- **output\_d** – tensorflow dataset containing the output Electric field that is to be predicted
- **dataset** – tensorflow dataset that is a zipped dataset of the inputs and output
- **test\_dataset** – tensorflow dataset for validation
- **train\_dataset** – tensorflow dataset for training
- **model** – tensorflow model that is loaded from file “model.h5”
- **history** – tensorflow history from model.fit()

**\_set\_in\_scan(*af*)**

Sets the `as_list` sizes for the input tensor. These must be set before the dataset is passed to `model.fit`

**Parameters** **af** – the tensor that will have its shape defined

**Return type** tensorflow tensor

**`_set_in_other`** (*af*)

Sets the `as_list` sizes for the input tensor. These must be set before the dataset is passed to `model.fit`

**Parameters** *af* – the tensor that will have its shape defined

**Return type** tensorflow tensor

**`_set_out`** (*af*)

Sets the `as_list` sizes for the input tensor. These must be set before the dataset is passed to `model.fit`

**Parameters** *af* – the tensor that will have its shape defined

**Return type** tensorflow tensor

**`read_in_scan`** (*filename*)

opens the hdf5 file (*filename*) and reads out the array stored in ‘input\_scan’

**Parameters** *filename* –

**Return type** 2D array

**`read_in_other`** (*filename*)

opens the hdf5 file (*filename*) and reads out the array stored in ‘input\_other’

**Parameters** *filename* –

**Return type** 1D array

**`read_out`** (*filename*)

opens the hdf5 file (*filename*) and reads out the array stored in ‘output’

**Parameters** *filename* –

**Return type** 1D array

**dnn\_predict()**

**Parameters**

- **new\_model** – the tensorflow model loaded from “model.h5”.
- **a** – contains a tensorflow glob of all the pulses in “pulses”.
- **in1** – is the dscan for a single pulse.
- **in2** – is the other inputs for a single pulse.
- **o1** – is the output electric field for a single pulse.
- **extra\_model** – a tensorflow model for extracting the value of an arbitrary layer in new\_model.
- **pred** – is the prediction of the arbitrary layer.
- **new\_predictions** – is the prediction of the output given the inputs.

Predicts the phase from a given d-scan trace.



A file that generates the model structure.

**dnn\_model()**

**Parameters**

- **input1\_temp** – used to turn grayscale input image into a 3 channels for rgb input
- **input1** – The input to the pre-trained models. Note this needs to be of shape 224x224x3.
- **input2** – The input for the model that holds the other inputs (L,R\_omega\_0,sinc\_width).
- **base\_model** – A tensorflow.keras.Application model. By default is a Densenet121 implementation with pre-trained weights.
- **x** – contains the layers that are after the pre-trained model but before the concatenation with input2.
- **merged\_model** – the layers that are after the concatenation.
- **outputs** – contains the output layer of the model.
- **model** – contains the tensorflow model of all the layers. Will be saved to “model.h5”.

**def vgg\_block(layer\_in, n\_filters, n\_conv, filter\_size=(3,3), in\_activation='relu', in\_padding=**

The functional blocks of the vgg architecture.

**Parameters**

- **layer\_in** –
- **n\_filters** –
- **n\_conv** –
- **filter\_size** –
- **in\_activation** –
- **in\_padding** –

**Return type** tensorflow keras layer

**inception\_module** (*layer\_in, f1, f2\_in, f2\_out, f3\_in, f3\_out, f4\_out*)

Functional blocks of the inception architecture. Be are that this model architecture get very large very fast.

**Parameters**

- **layer\_in** –
- **f1** –
- **f2\_in** –
- **f2\_out** –
- **f3\_in** –
- **f3\_out** –
- **f4\_out** –

**residual\_module** (*layer\_in, n\_filters*)

Functional block of the residual architecture.

**Parameters**

- **layer\_in** –
- **n\_filters** –

## CHAPTER 6

---

main

---

a file that generates the pulses that are used in the machine learning





#### 7.1 hdf5\_check

a file for reading hdf5 files to interrogate them and find their structure.

#### 7.2 exp\_dataio

a file for parsing the data that is produced in the experiment and given to us in .mat files.



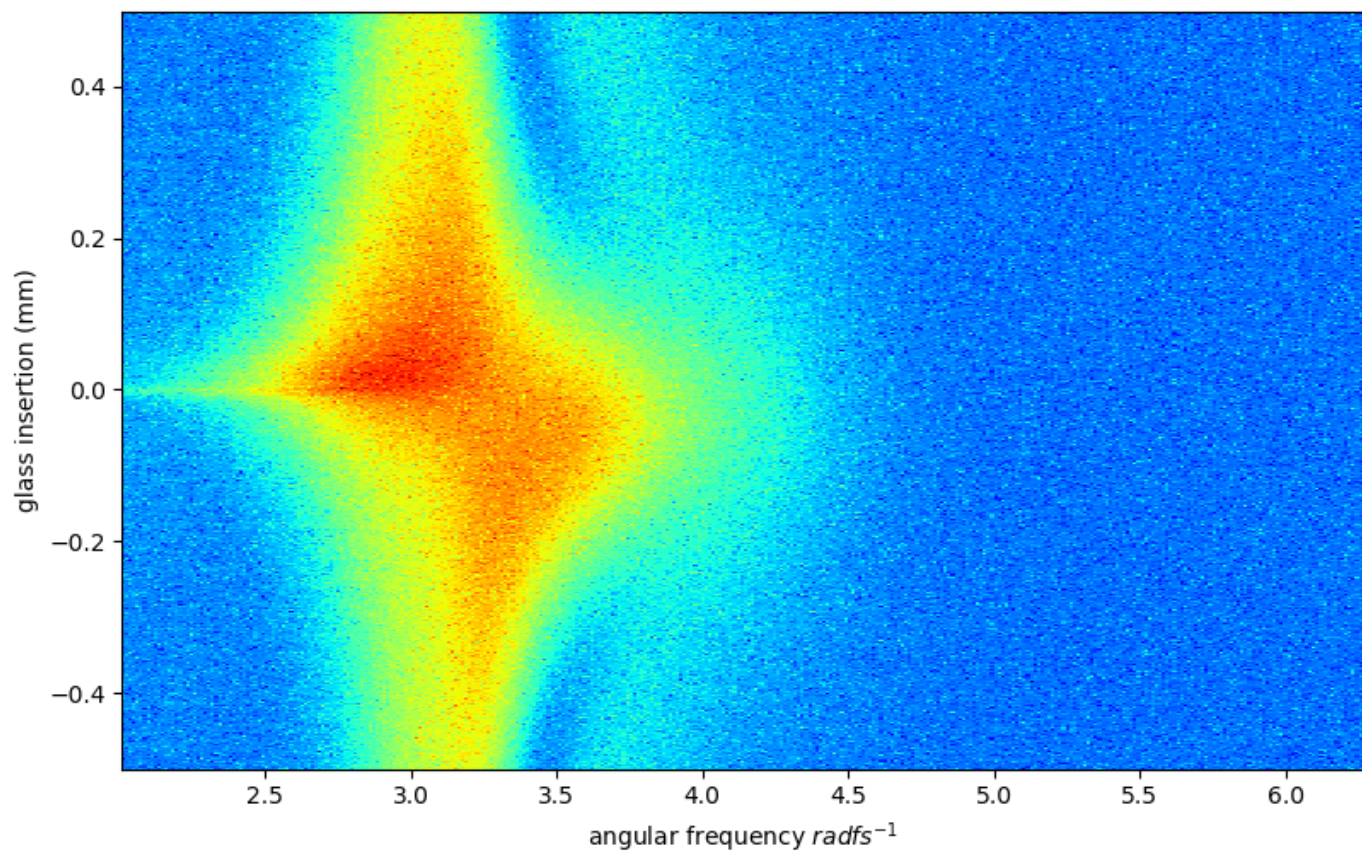
## CHAPTER 8

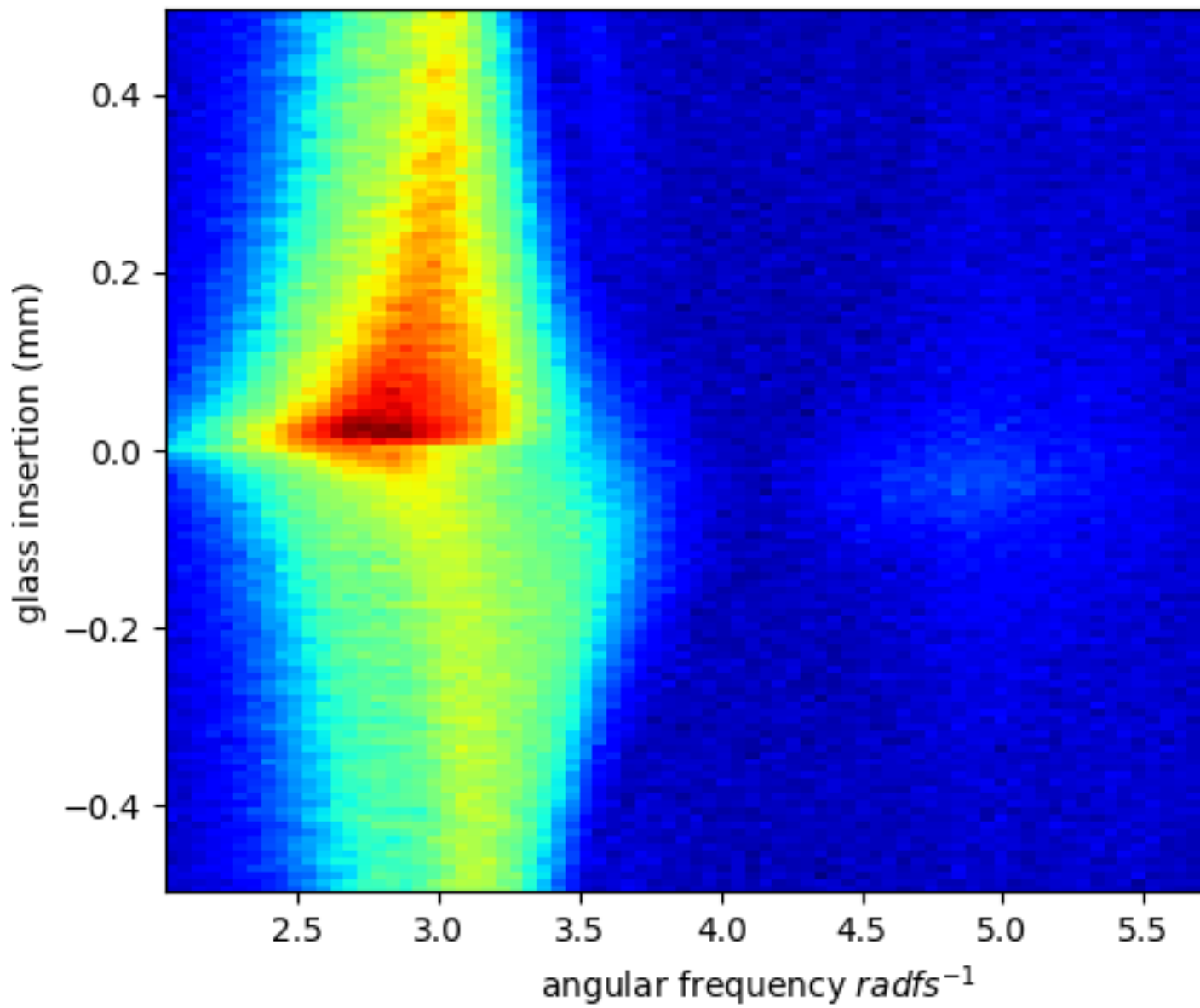
---

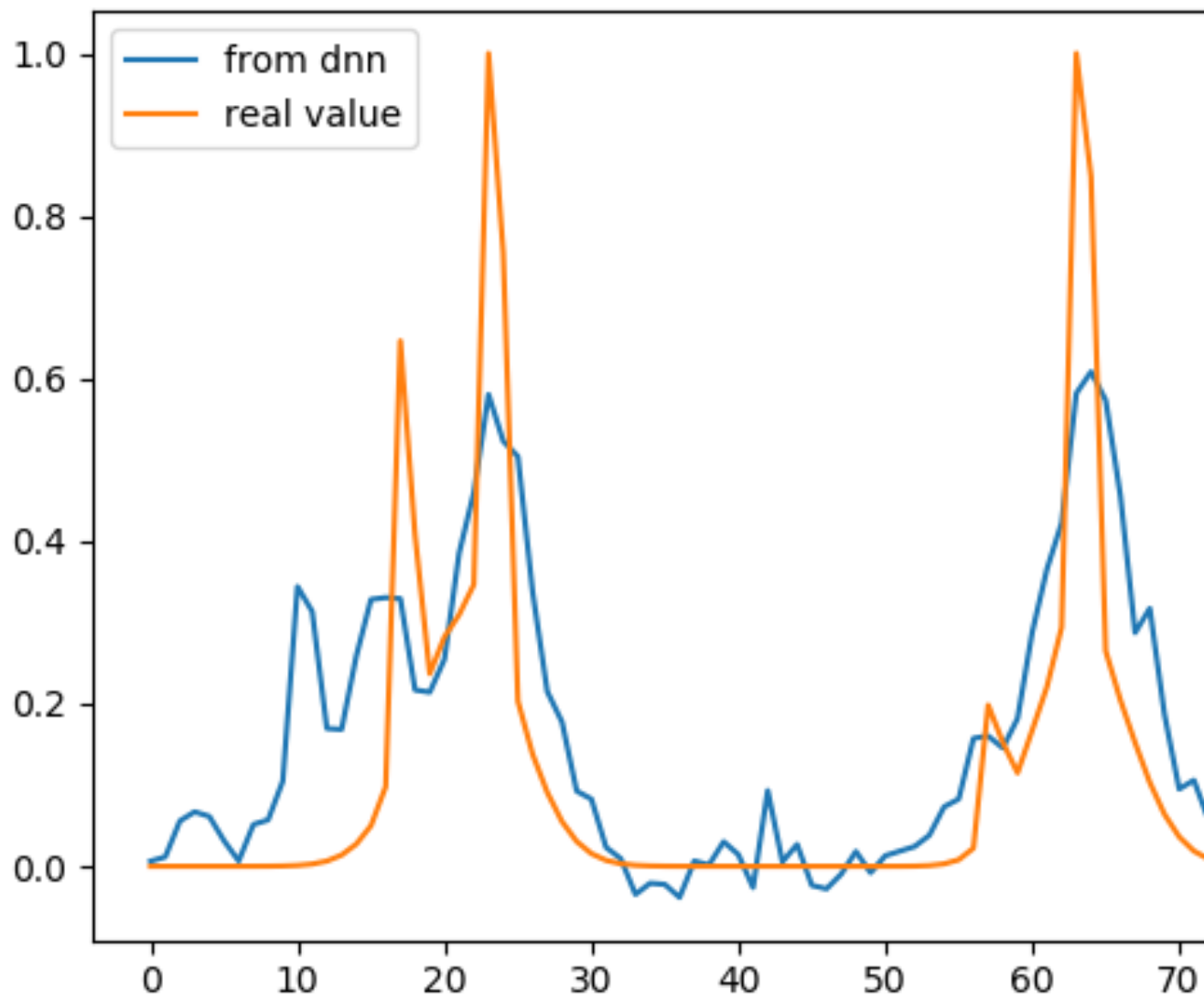
about

---

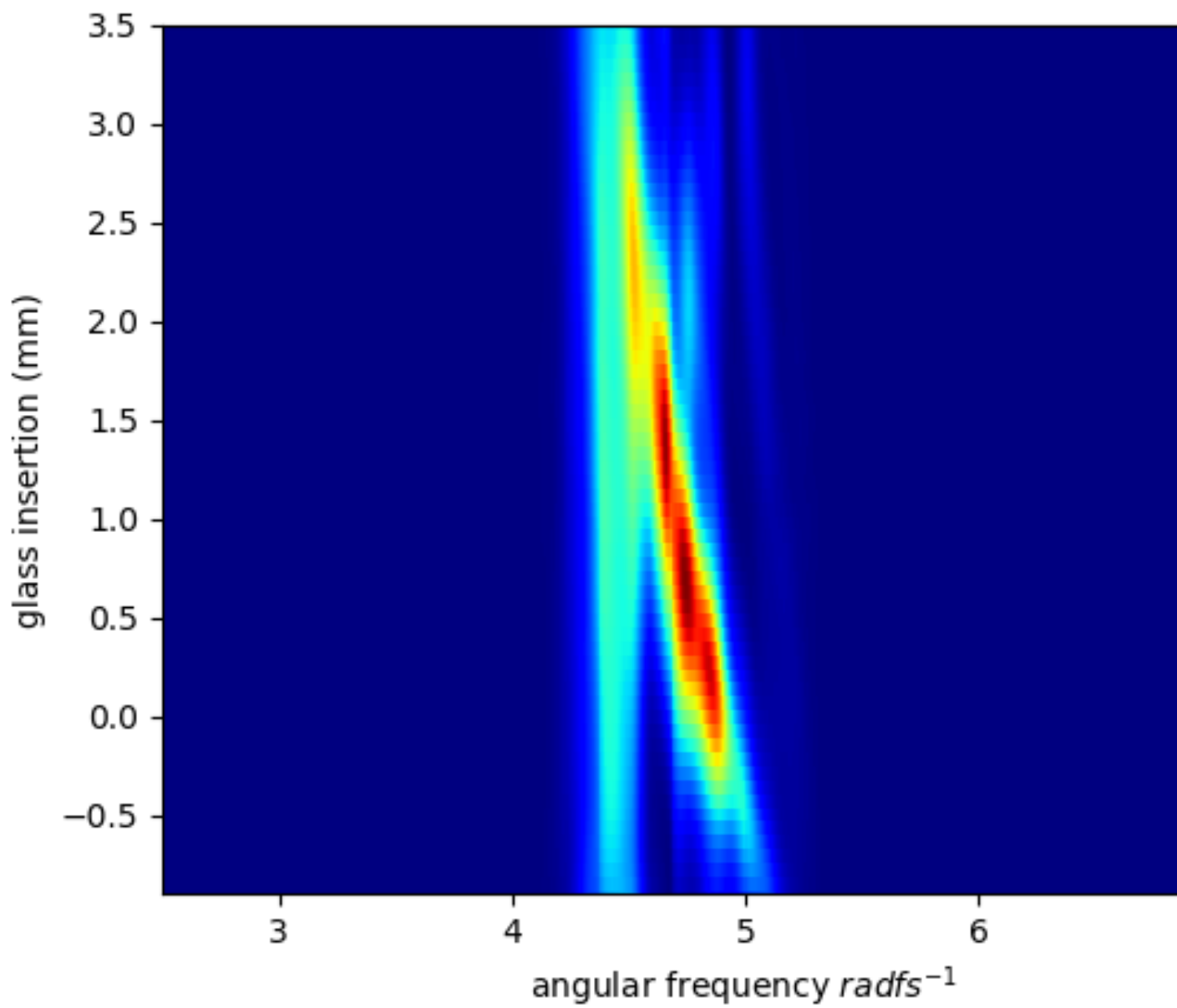
This is an MSci physics project by a pair of physics undergraduates at Imperial College London. The aim of the project was to first implement the work done in ‘paper’, and then to apply this to some experimental data. Beyond this there were aims to provide dscan characterisation “in real time”. There are also attempts being made to improve upon the speed and accuracy of the neural network used in the previously mentioned paper.











- [genindex](#)
- [modindex](#)
- [search](#)



## Symbols

`__init__()` (built-in function), 7  
`_set_in_other()` (built-in function), 13  
`_set_in_scan()` (built-in function), 13  
`_set_out()` (built-in function), 14

## D

`D_scan()` (built-in function), 11  
`D_trace()` (built-in function), 9  
`dnn_learn()` (built-in function), 13  
`dnn_model()` (built-in function), 17  
`dnn_predict()` (built-in function), 15

## E

`E_gaussian()` (built-in function), 10  
`E_omega_gen()` (built-in function), 10

## H

`h5out()` (built-in function), 8

## I

`inception_module()` (built-in function), 18

## M

`make_dsca_1z()` (built-in function), 8  
`make_dscan_all()` (built-in function), 8  
`make_E_omega_plot()` (built-in function), 8, 9  
`make_E_omega_plot2()` (built-in function), 9  
`make_E_t_plot()` (built-in function), 9  
`make_I_labda_plot()` (built-in function), 9  
`make_I_omega_plot()` (built-in function), 8  
`make_I_t_plot()` (built-in function), 9  
`make_phi()` (built-in function), 10  
`make_plouts()` (built-in function), 9  
`make_psi()` (built-in function), 10  
`my_compress()` (built-in function), 8

## P

`pulse_class` (built-in class), 7

## R

`read_in_other()` (built-in function), 14  
`read_in_scan()` (built-in function), 14  
`read_out()` (built-in function), 14  
`refractive_index()` (built-in function), 11  
`residual_module()` (built-in function), 18

## S

`save_instance()` (built-in function), 8