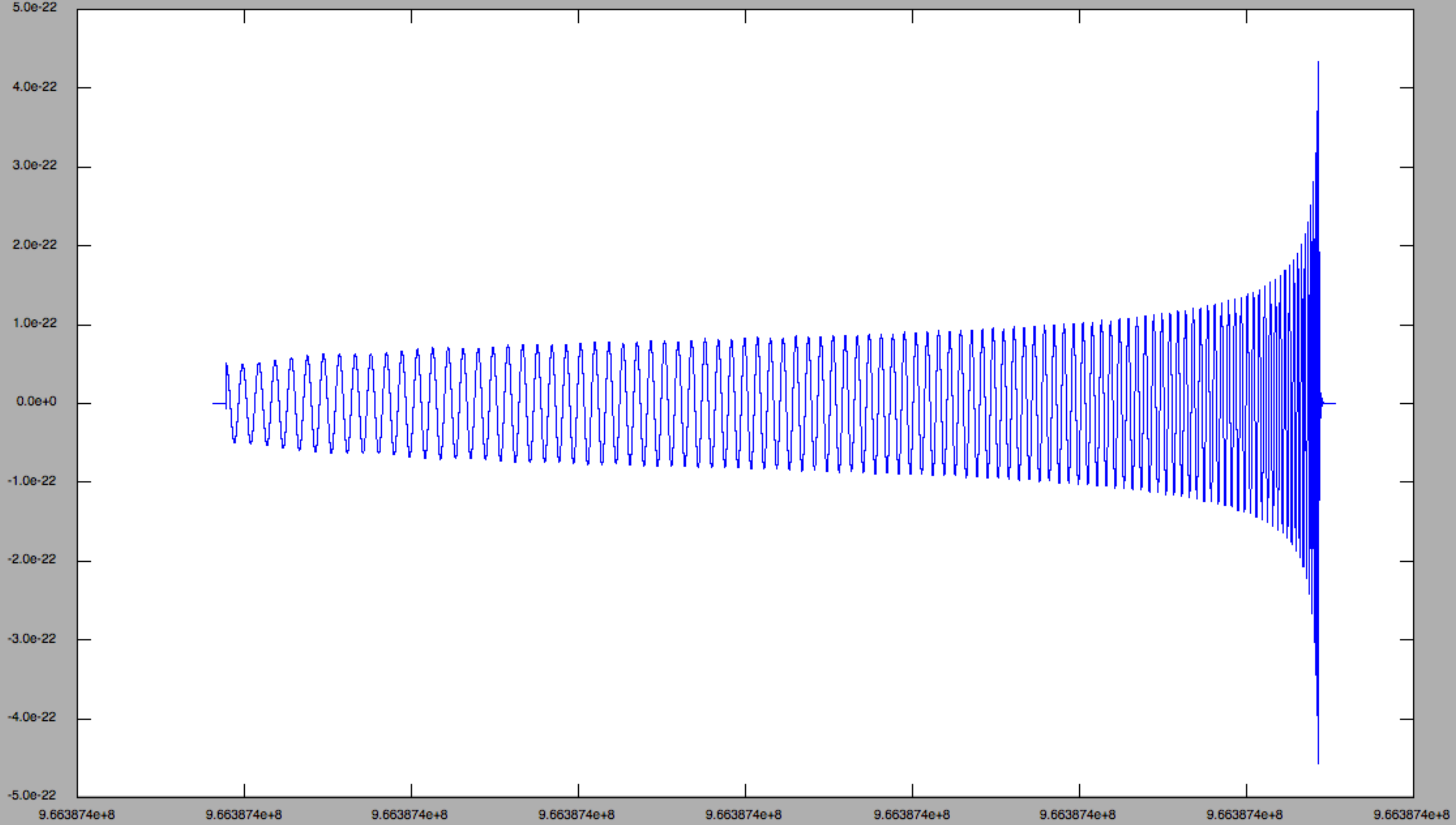
HACK DAY – LIGO CLASSIFICATION USING KERAS

DR. MATTHEW SMITH

ADACS, SWINBURNE UNIVERSITY OF TECHNOLOGY

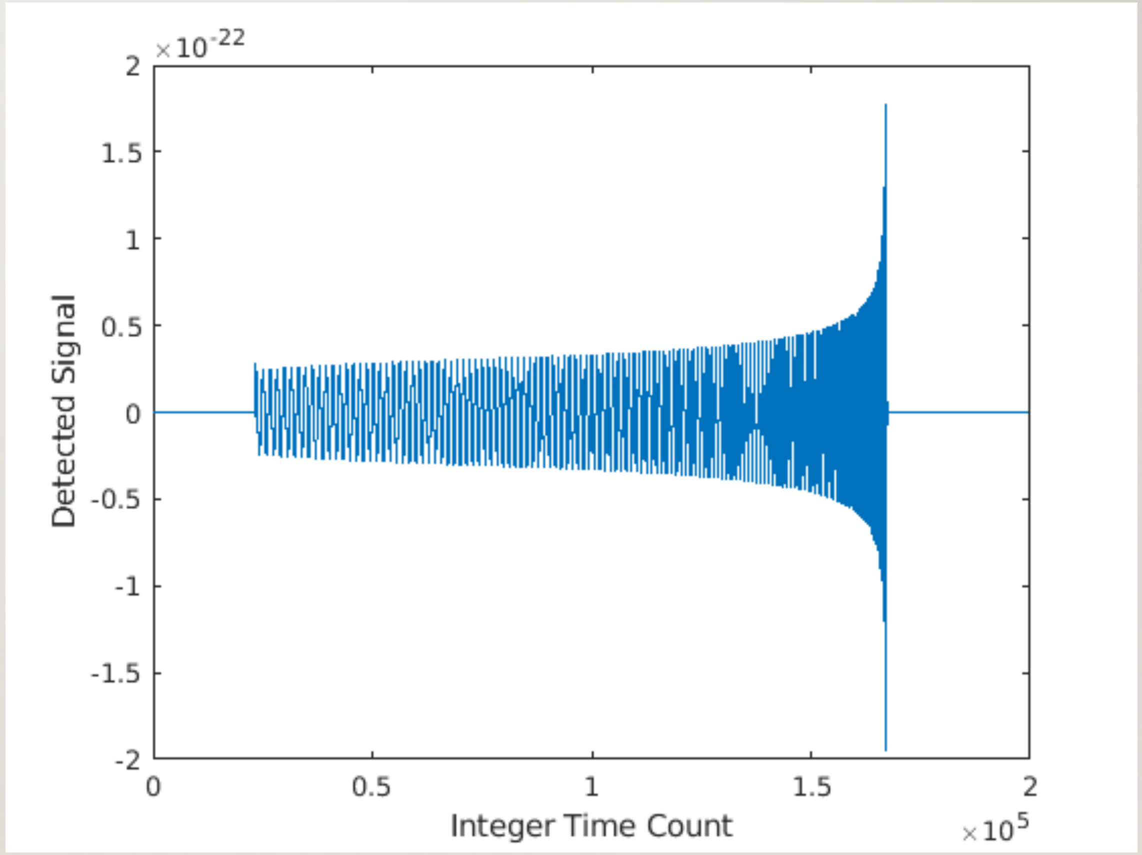
PROBLEM DEFINITION

- Welcome to the last part of this ML workshop.
- Today, your mission is simple: create a tool which, when fed a LIGO data sequence, can let the user know if there is a gravity wave present or not. Hence, this is a binary classification problem involving sequences.
- You have another option – using the material provided thus far, feel free to attempt to perform your own ML tasks. If you are interested in Image Classification, please use the reserved Ozstar nodes for submission.


5.0e-22
4.0e-22
3.0e-22
2.0e-22
1.0e-22
0.0e+0
-1.0e-22
-2.0e-22
-3.0e-22
-4.0e-22
-5.0e-22
9.663874e+8
9.663874e+8
9.663874e+8
9.663874e+8
9.663874e+8
9.663874e+8
9.663874e+8
9.663874e+8
9.663874e+8

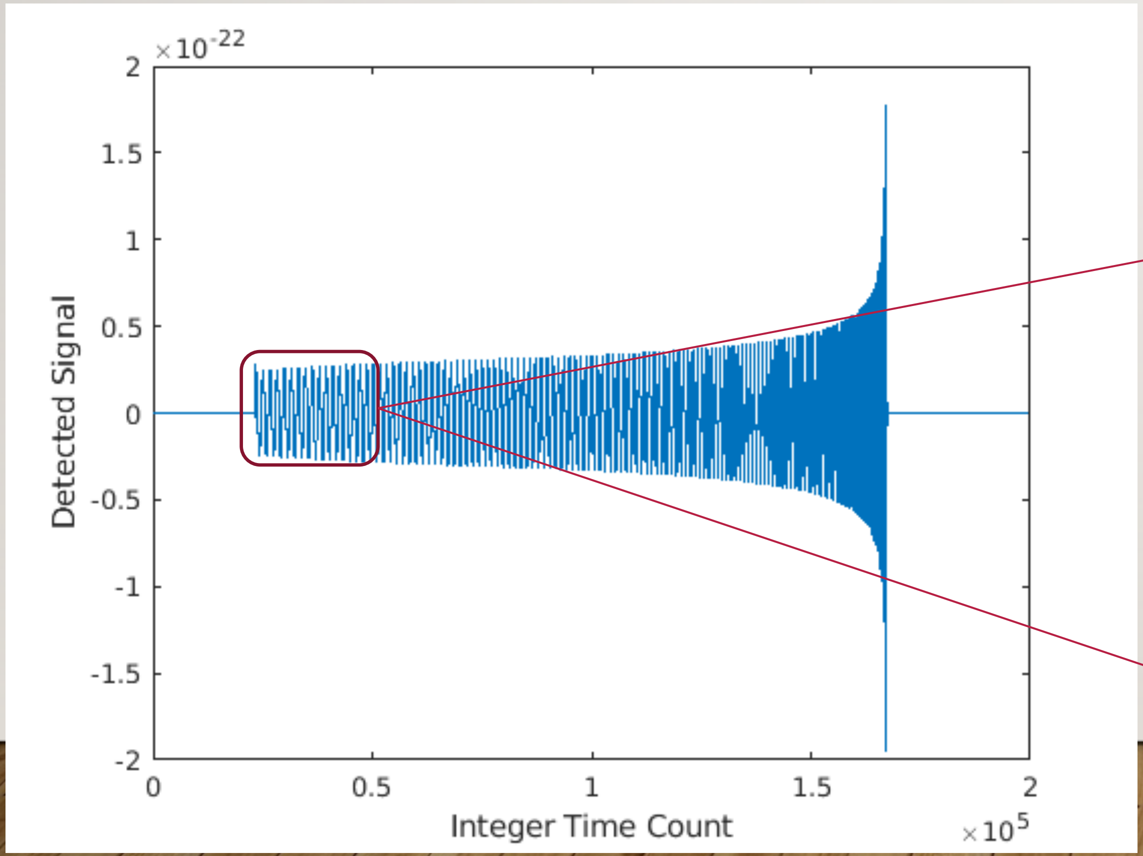
LIGO SIGNALS

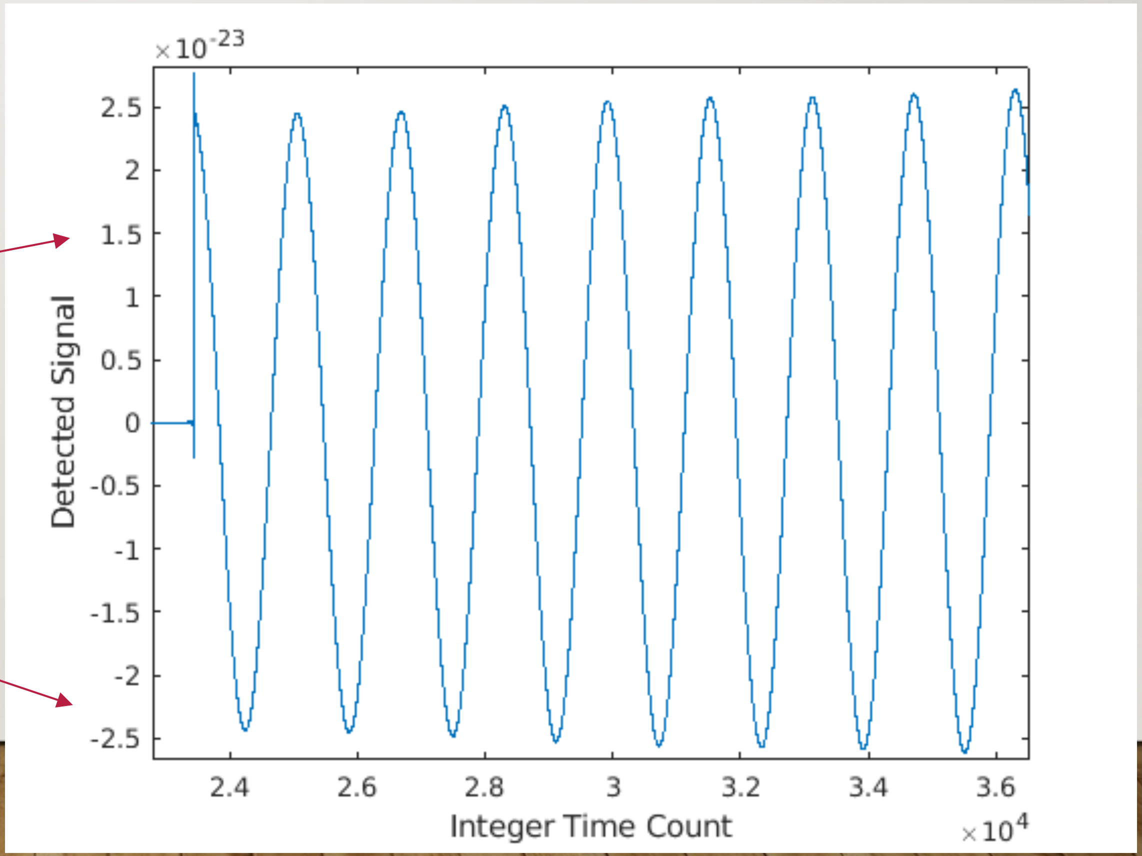
- Let’s introduce you to your target – this is what a simulated gravitational wave, as might be observed by LIGO, looks like without the noise.
- Let’s draw some attention to some features of this signal:
  - Average magnitude is around 0.5e-22, and
  - Each signal contains 200,000 pieces of data.
- This data is double precision, and saved to file in binary format.



LIGO SIGNALS

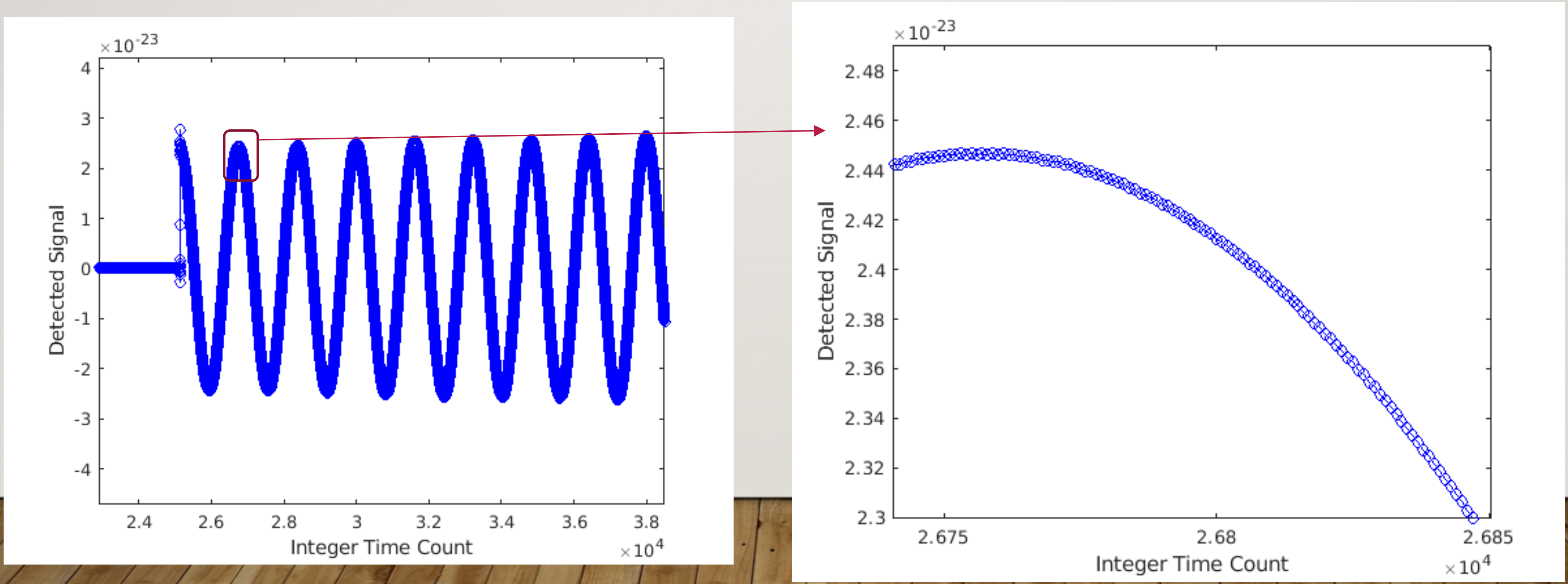
- Let’s have a closer look at the data – it is oscillatory with increasing frequency and magnitude.





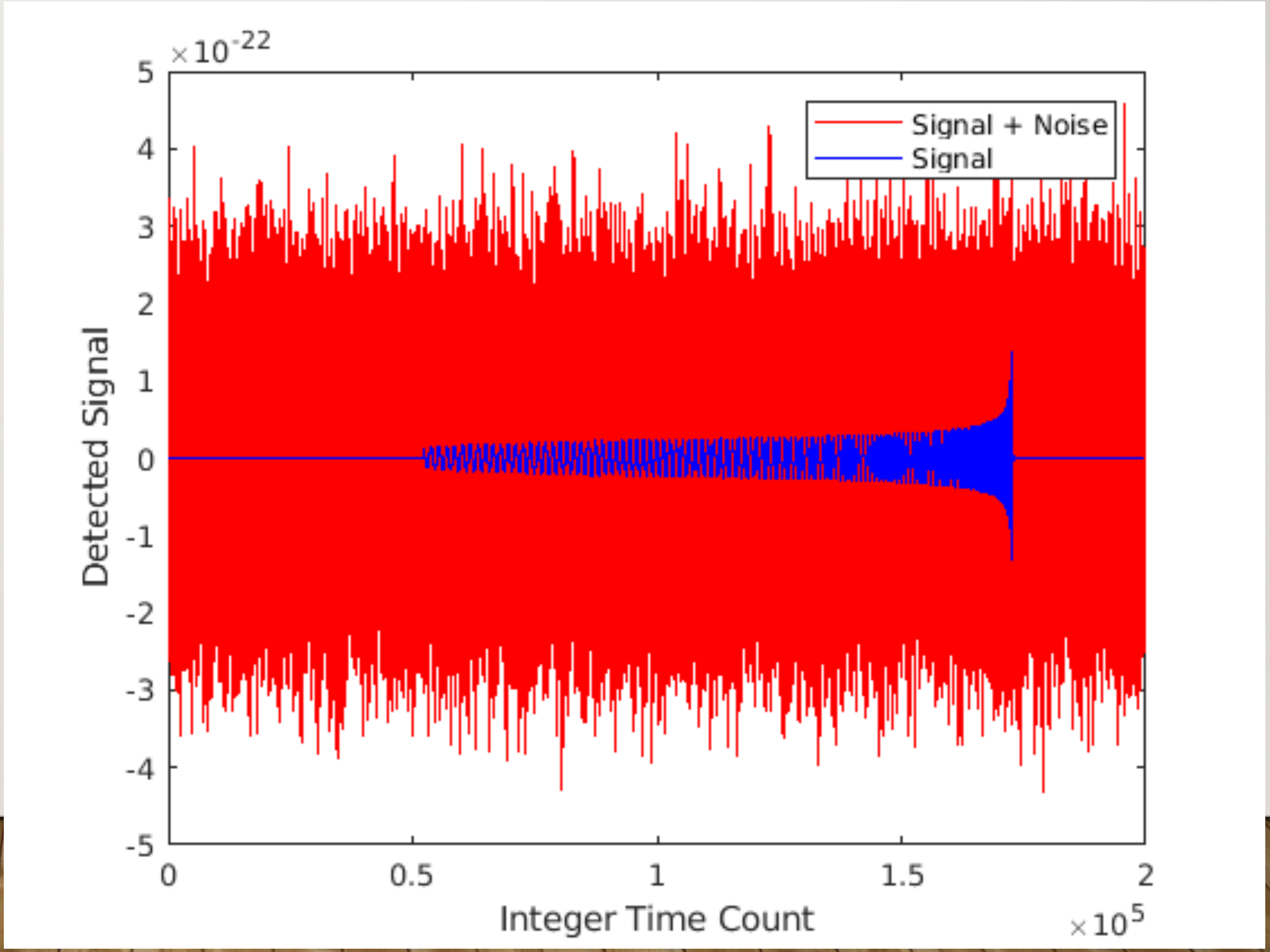
LIGO SIGNALS

- If you check the frequency of reported data against the wave frequency, we can see this data is excellently resolved.



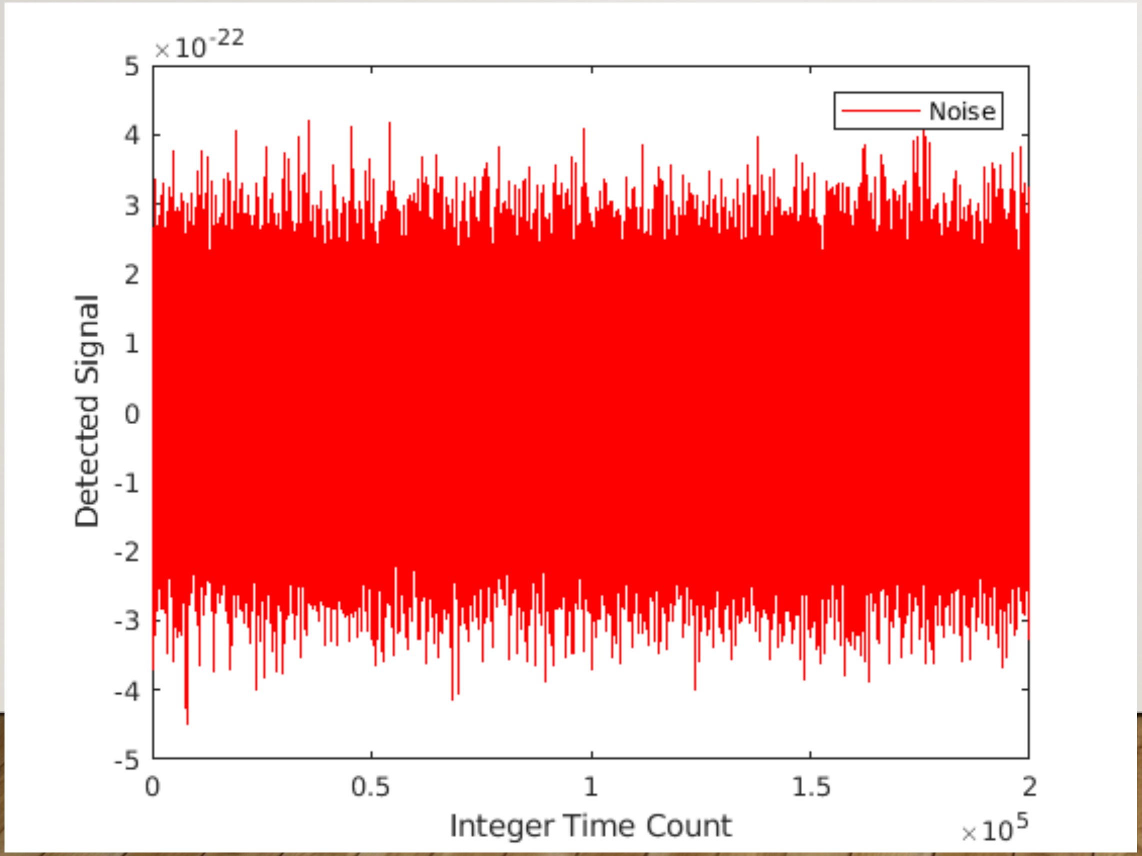
LIGO SIGNALS

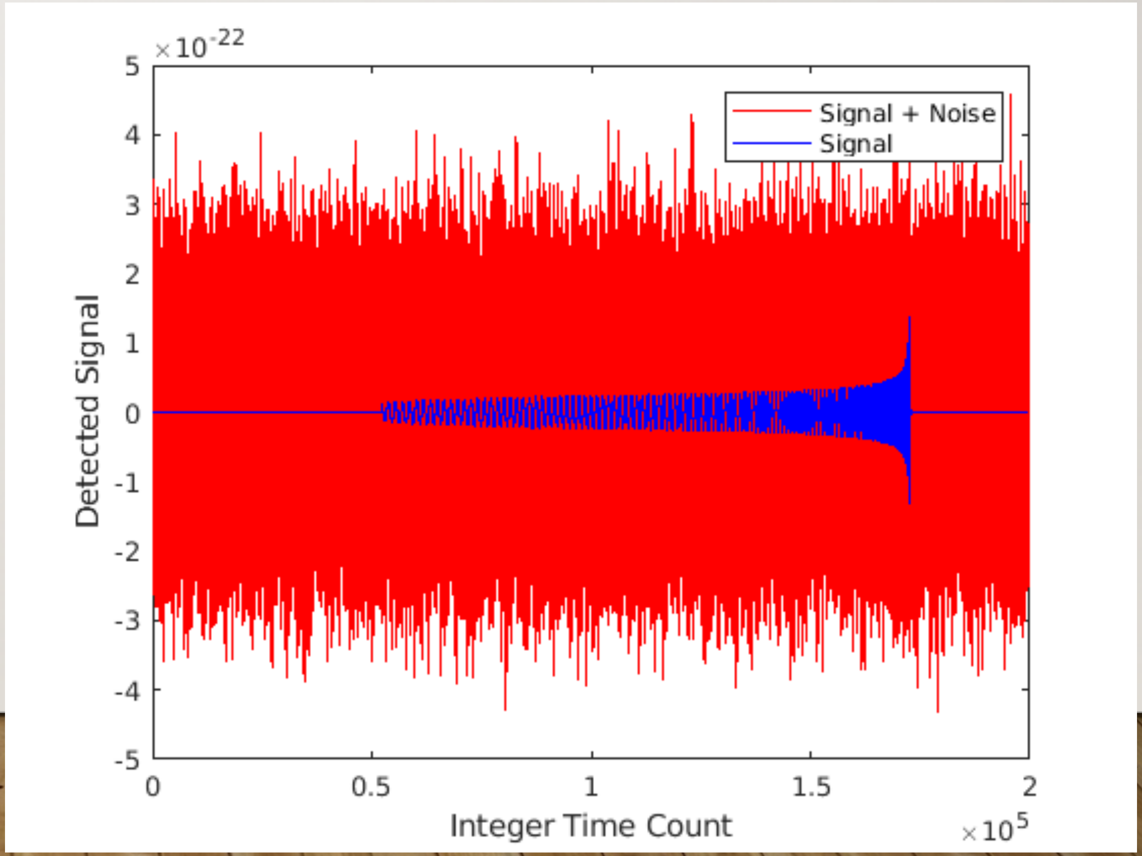
- Many experimental observations contain noise - this one is no exception.
- Thus, on top of this signal, we have added noise – which is taken from a normally distributed random number generator.
- As a matter of fact, the noise present here is much smaller than the actual noise present in real LIGO signals – but we want today to be at least a little fun.



LIGO SIGNALS

- So your mission – create an AI which examines a data packet of 200,000 doubles and classifies it as either (0) not having a gravity wave, or (1) containing a gravitational wave.





WHERE TO START..

- Start with the codes provided in ADACS_ML_A and ADACS_ML_B.
- These will need to be modified to:
  - Load a data stream containing 200,000 doubles from file.
  - Filter the data in an attempt to remove the noise and reveal the gravitational wave present,
  - Reducing the problem size – perhaps use 200 to 2000 data points instead of the full 200,000
  - Employ dropout in an attempt to prevent overfitting on the noise present.

- You should then save your keras model, and create an inference script (infer.py) which loads the JSON data and model weights and is used for categorising a new LIGO signal.

WHERE TO START..

- When you git cloned this material (ADACS_ML_C), you copied the LIGO signal data you require in the Test_Easy and Train_Easy directories.
- The naming format of these files is: X_LIGO_ZZ.dat where ZZ is an integer from 1 to 400 for the training data and 1 to 100 for the test data. The file containing the classification is Y_LIGO_ZZ.dat (same ID).
- You'll also find the filtering and dropout PDF included with this data; don't be afraid to scp it to your local computer for your reference.

ADVICE?

- Refer to the previous PDF's containing information on filtering and dropout.
- Experiment with filter frequency cutoff values.
- Experiment with the number of neurons in each layer, and the number of layers.
- Experiment with dropout values – don't use 1.0 (which would be silly) but using 0.0 is OK (this is essentially no dropout)
- Write a new view.py script which allows you to view the filtered signal – if you can't see the gravity wave, then there is a good chance your AI won't be able to see it. Use this script to test filter frequency cutoff values.

ADVICE?

- Rewrite your codes to run multiple training sessions – these are known as ensembles - to get a better idea of the average influence of your filter cutoff and dropout values.
- Fix the random seed – by setting the seed, the random numbers used as part of the ML algorithms will be consistent, and your experiments will be repeatable.
- NORMALIZE YOUR DATA. The data contained in the /Test and /Train folders is small (1e-22, 1e-23) – you should divide this by some characteristic small number (perhaps 1e-23) in order to avoid numerical precision problems.
- Work in groups – cooperate with one or two other people. Don't be afraid to communicate.

WHAT CAN YOU EXPECT?

- Without dropout, you might get up to 70% accuracy, depending on how many neurons you use, your filter and the complexity of your network. Simple is better.
- With dropout and a good filter, an average accuracy of 95% should be obtainable. I’d say this is a good target for this workshop.
- You should be able to train your AI in a couple of seconds – if you need more than 2 minutes to train your AI, you are trying to hard. Change your filter, use fewer data points, check precision – if you are still stuck and your AI isn’t learning, consult with other groups.

GOOD LUCK

- Any questions? Put your hand up, or email me: msmith@astro.swin.edu.au