DSE - Mini Project

Exploring Machine Learning Methods for Fault Classification and Location in Distribution Systems

Ramakrishna Reddy Palle **21EE01026**
Tejdeep Reddy Gudise **21EE01022**

Contents

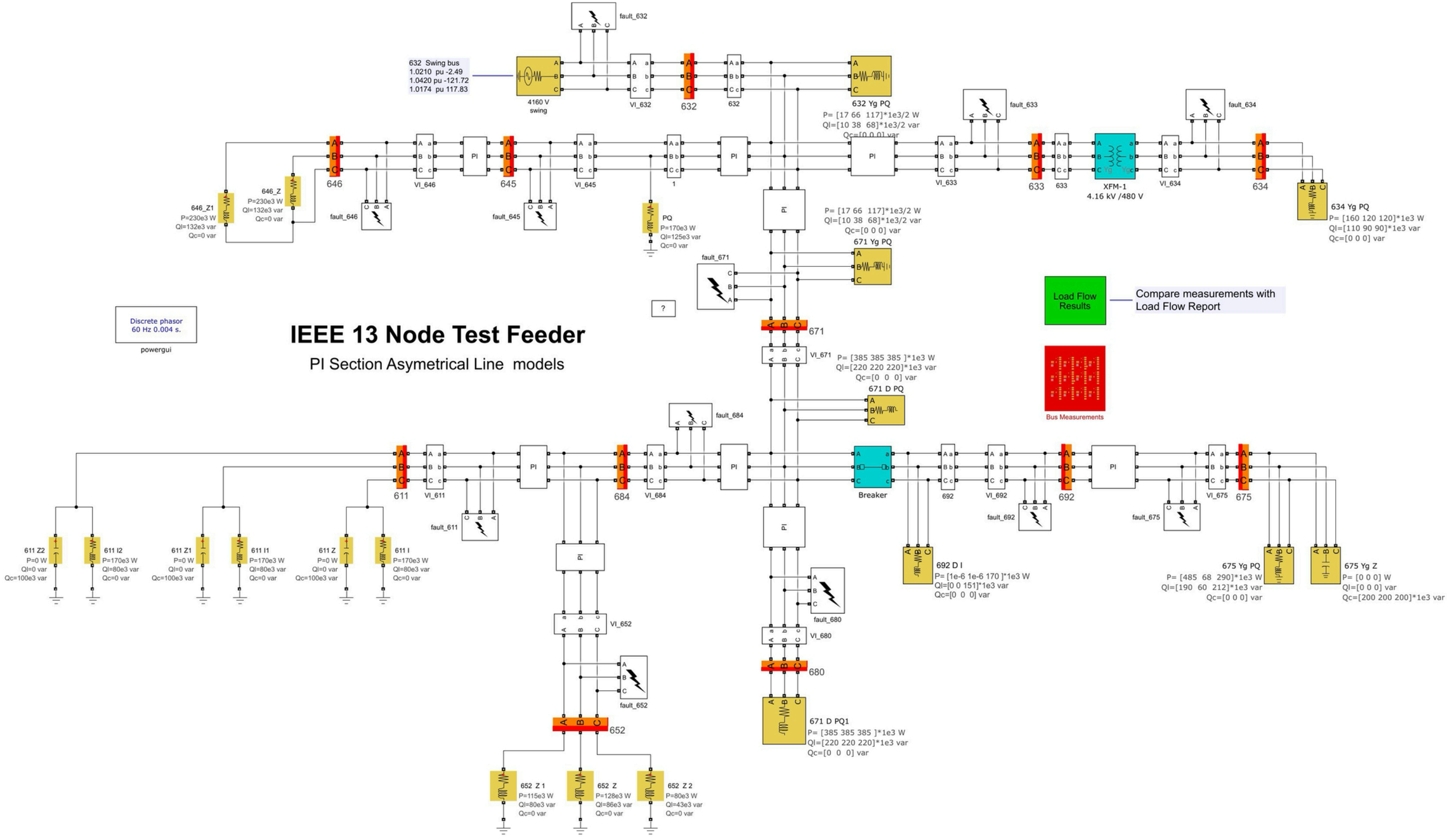
1. Problem Statement

Real time **fault classification and location** in a distribution system is vital for ensuring power system stability and preventing disruptions, power loss, equipment damage, and safety risks.

Quick identification and isolation of faulty sections are crucial for maintaining an uninterrupted power supply. With advances in **Deep learning**, there is potential to develop efficient methods for real-time fault identification and isolation. The aim is to create a machine learning-based system to accurately locate and classify faults, improving the reliability and safety of power systems.

2. Data Generation

- Distribution System related datasets are not available in open source
- Also there are indefinite number of combinations of equipment/load possible in distribution.
- Hence we generate our own fault data specific to our system.
- Used the standard "**IEEE 13 Node Test Feeder**" to simulate faults and generate fault data using **SIMULINK, MATLAB.**

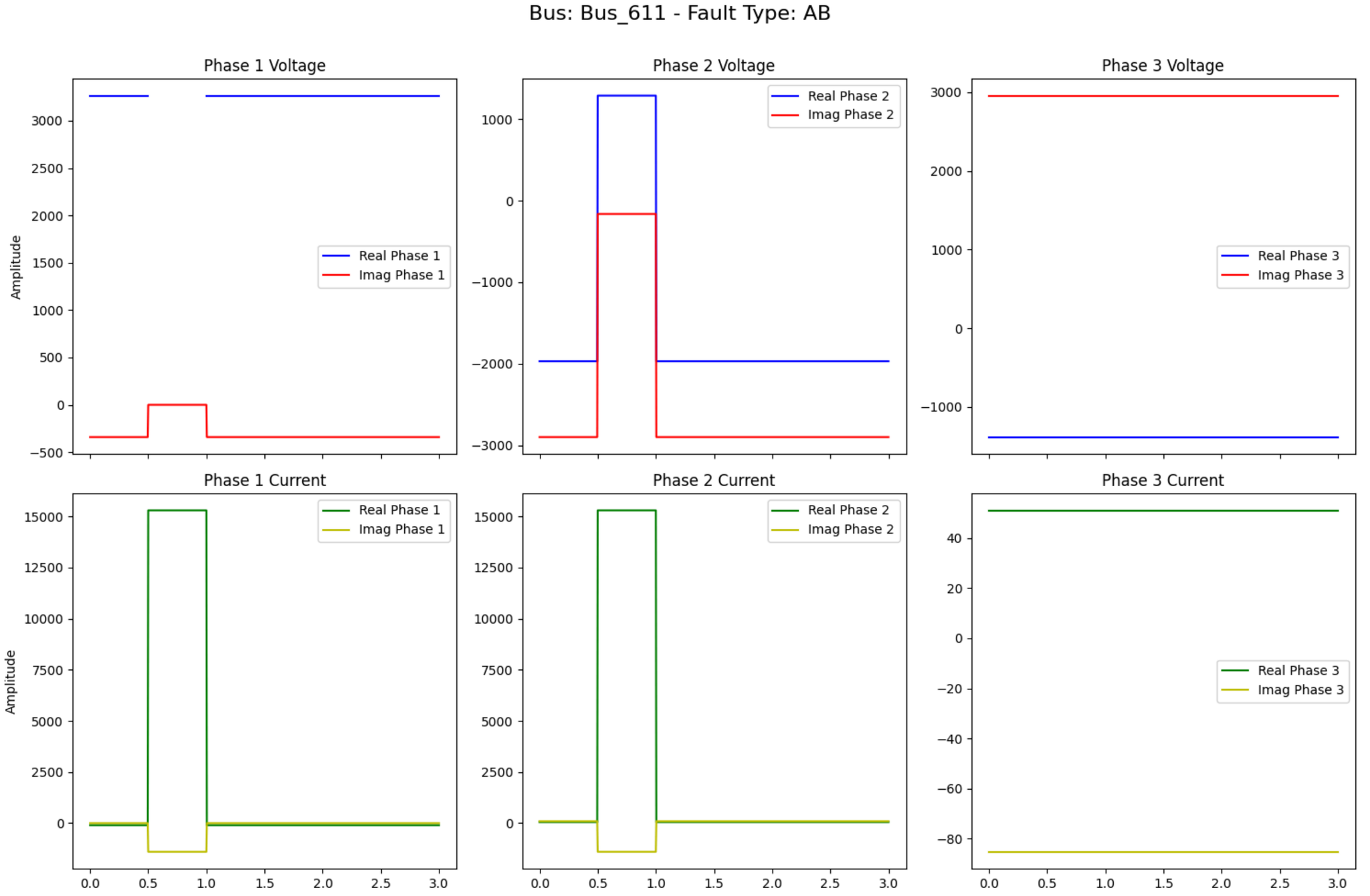

IEEE 13 Node Test Feeder
PI Section Asymetrical Line models
Discrete phasor
60 Hz 0.004 s.
powergui
Load Flow Results
Compare measurements with Load Flow Report
Bus Measurements
632 Swing bus
1.0210 pu -2.49
1.0420 pu -121.72
1.0174 pu 117.83
4160 V swing
632 Yg PQ
P= [17 66 117]*1e3/2 W
Ql=[10 38 68]*1e3/2 var
Qc=[0 0 0] var
634 Yg PQ
P= [160 120 120]*1e3 W
Ql=[110 90 90]*1e3 var
Qc=[0 0 0] var
XFM-1
4.16 kV /480 V
646_Z1
P=230e3 W
Ql=132e3 var
Qc=0 var
646_Z
P=230e3 W
Ql=132e3 var
Qc=0 var
PQ
P=170e3 W
Ql=125e3 var
Qc=0 var
671 Yg PQ
P= [17 66 117]*1e3/2 W
Ql=[10 38 68]*1e3/2 var
Qc=[0 0 0] var
VI_671
P= [385 385 385]*1e3 W
Ql=[220 220 220]*1e3 var
Qc=[0 0 0] var
671 D PQ
692 D I
P= [1e-6 1e-6 170]*1e3 W
Ql=[0 0 151]*1e3 var
Qc=[0 0 0] var
675 Yg PQ
P= [485 68 290]*1e3 W
Ql=[190 60 212]*1e3 var
Qc=[0 0 0] var
675 Yg Z
P= [0 0 0] W
Ql=[0 0 0] var
Qc=[200 200 200]*1e3 var
671 D PQ1
P= [385 385 385]*1e3 W
Ql=[220 220 220]*1e3 var
Qc=[0 0 0] var
611 Z2
P=0 W
Ql=0 var
Qc=100e3 var
611 I2
P=170e3 W
Ql=80e3 var
Qc=0 var
611 Z1
P=0 W
Ql=0 var
Qc=100e3 var
611 I1
P=170e3 W
Ql=80e3 var
Qc=0 var
611 Z
P=0 W
Ql=0 var
Qc=100e3 var
611 I
P=170e3 W
Ql=80e3 var
Qc=0 var
652 Z 1
P=115e3 W
Ql=80e3 var
Qc=0 var
652 Z
P=128e3 W
Ql=86e3 var
Qc=0 var
652 Z 2
P=80e3 W
Ql=43e3 var
Qc=0 var
fault_632
fault_633
fault_634
fault_646
fault_645
fault_671
fault_684
fault_611
fault_692
fault_675
fault_680
fault_652
VI_632
VI_633
VI_634
VI_646
VI_645
VI_684
VI_611
VI_692
VI_675
VI_680
VI_652
Breaker
PI
632
633
634
646
645
671
684
611
692
675
680
652
1
?

3. Time Domain Fault Data Analysis

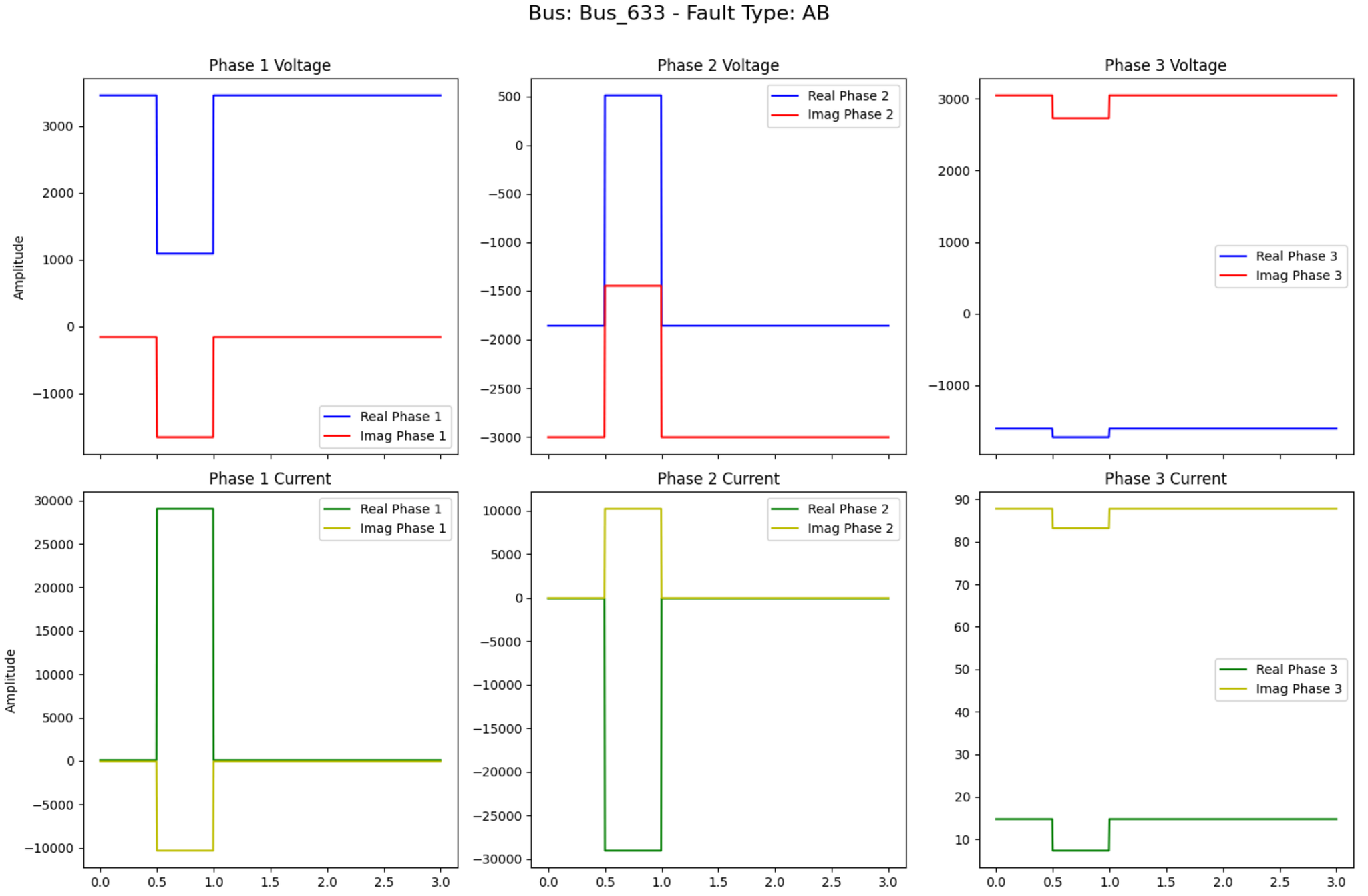
Simulink Model Parameters:

- **Fault Resistance, Ground Resistance :** 0.01 ohm
- **Snubber Resistance :** 1e6
- **Simulation Time :** 3 sec
- **Fixed Steps :** 0.004 sec
- **Fault Switching Times :** [0.5, 1] sec
- **Measurements :** 3 phase VI block at each bus, total of 12 buses
- Total of **12** faults {*AB, AC, AG, BC, BG, CG, ABC, ABG, ACG, BCG, ABCG, NoFault*}
- Fault simulation done using **MATLAB** script automation - 1 fault at 1 bus per simulation
- Measurments saved in ***.csv*** format during automation.

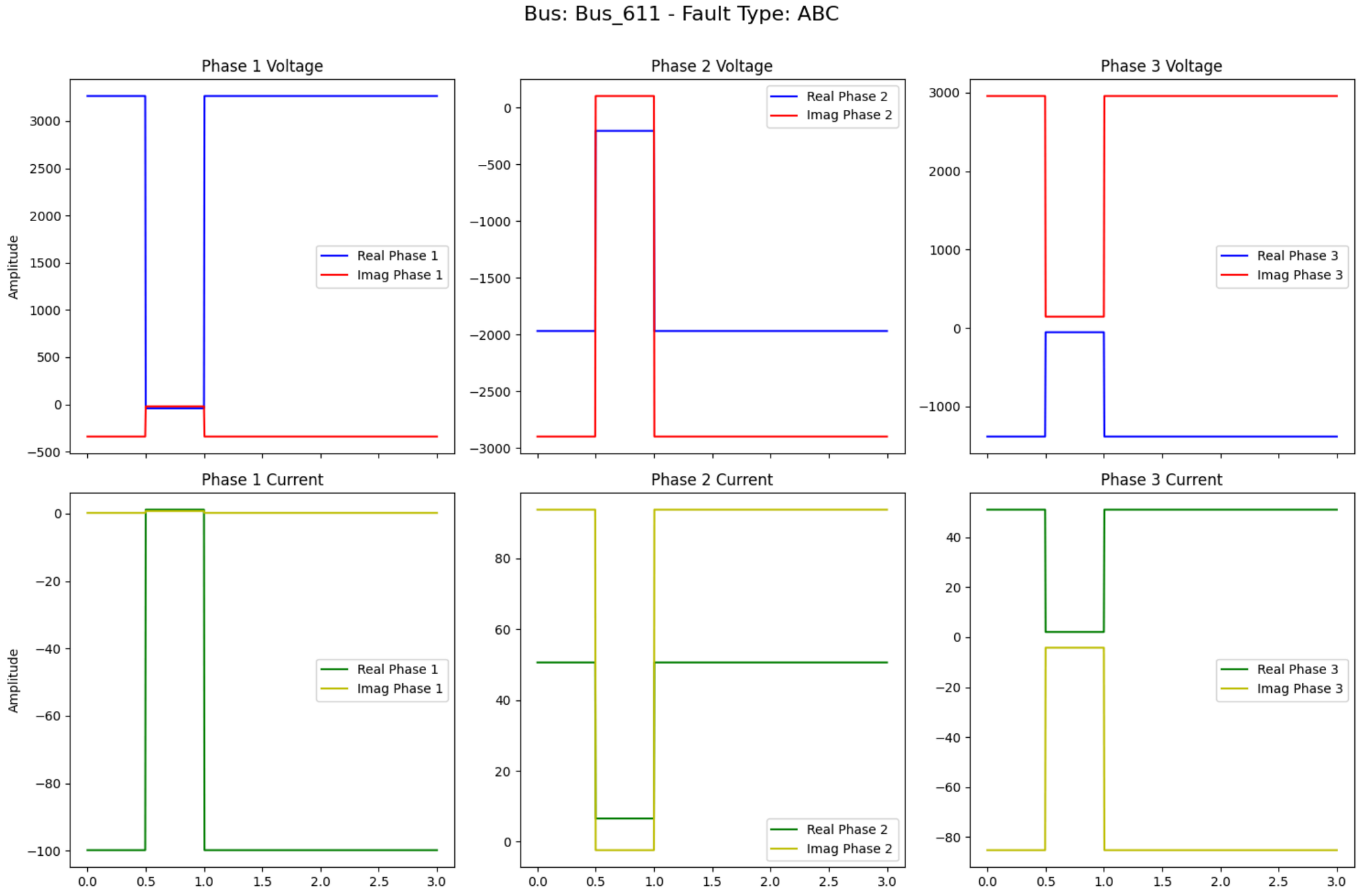
	A	B	C	D	E	F	G	H	I
1	Time	FaultyBusVoltages_1	FaultyBusVoltages_2	FaultyBusVoltages_3	FaultyBusCurrents_1	FaultyBusCurrents_2	FaultyBusCurrents_3	FaultType	FaultLocation
2	0	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
3	0.004	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
4	0.008	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
5	0.012	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
6	0.016	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
7	0.02	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
8	0.024	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
9	0.028	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
10	0.032	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
11	0.036	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
12	0.04	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
13	0.044	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
14	0.048	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
15	0.052	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
16	0.056	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
17	0.06	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
18	0.064	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
19	0.068	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
20	0.072	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
21	0.076	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
22	0.08	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611
23	0.084	3265-340.5i	-1968-2897i	-1384+2957i	-99.86+0.2117i	50.6+93.75i	51.03-85.23i	ABC	611



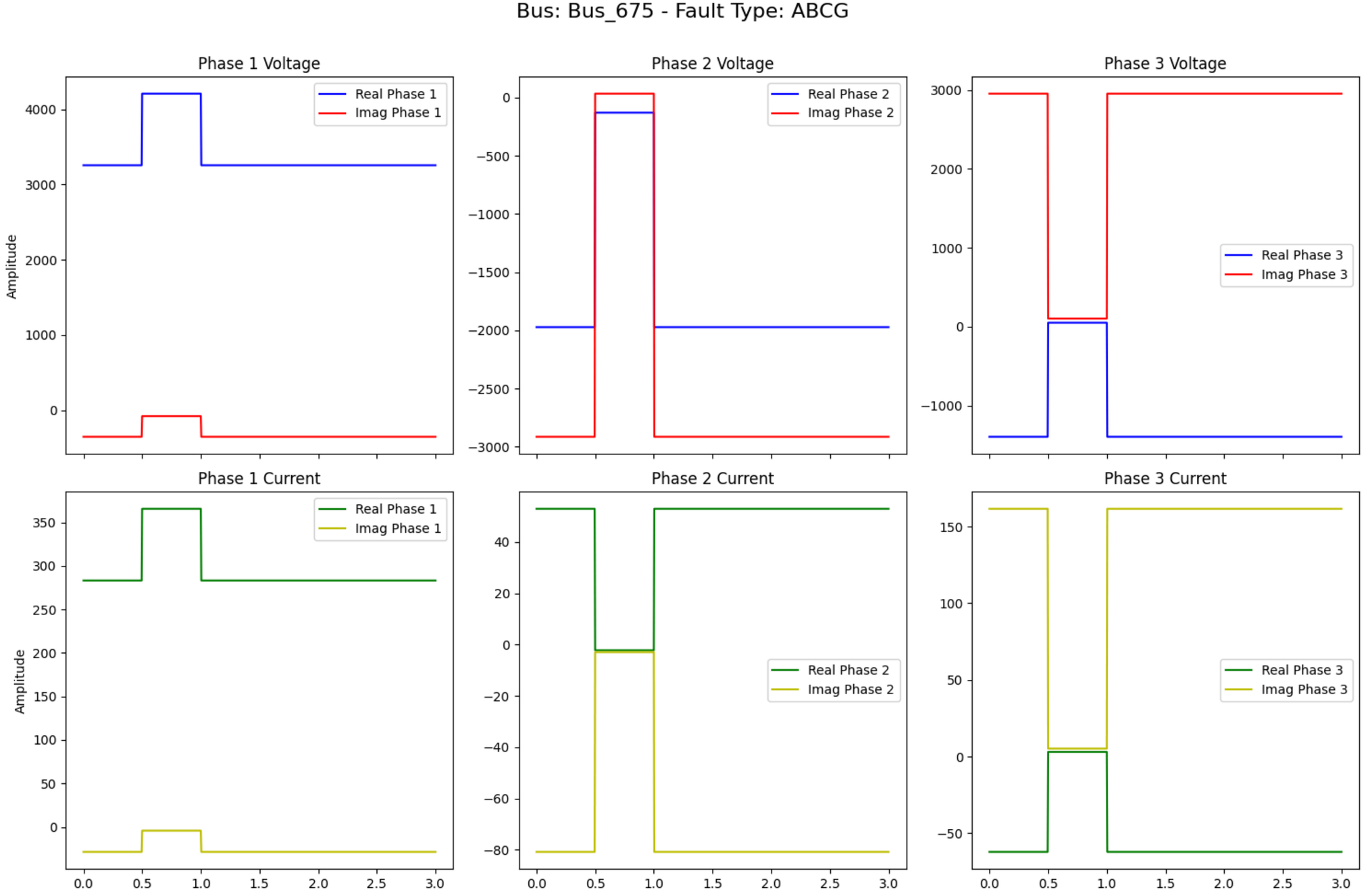
*plotted using python script



*plotted using python script



*plotted using python script



*plotted using python script

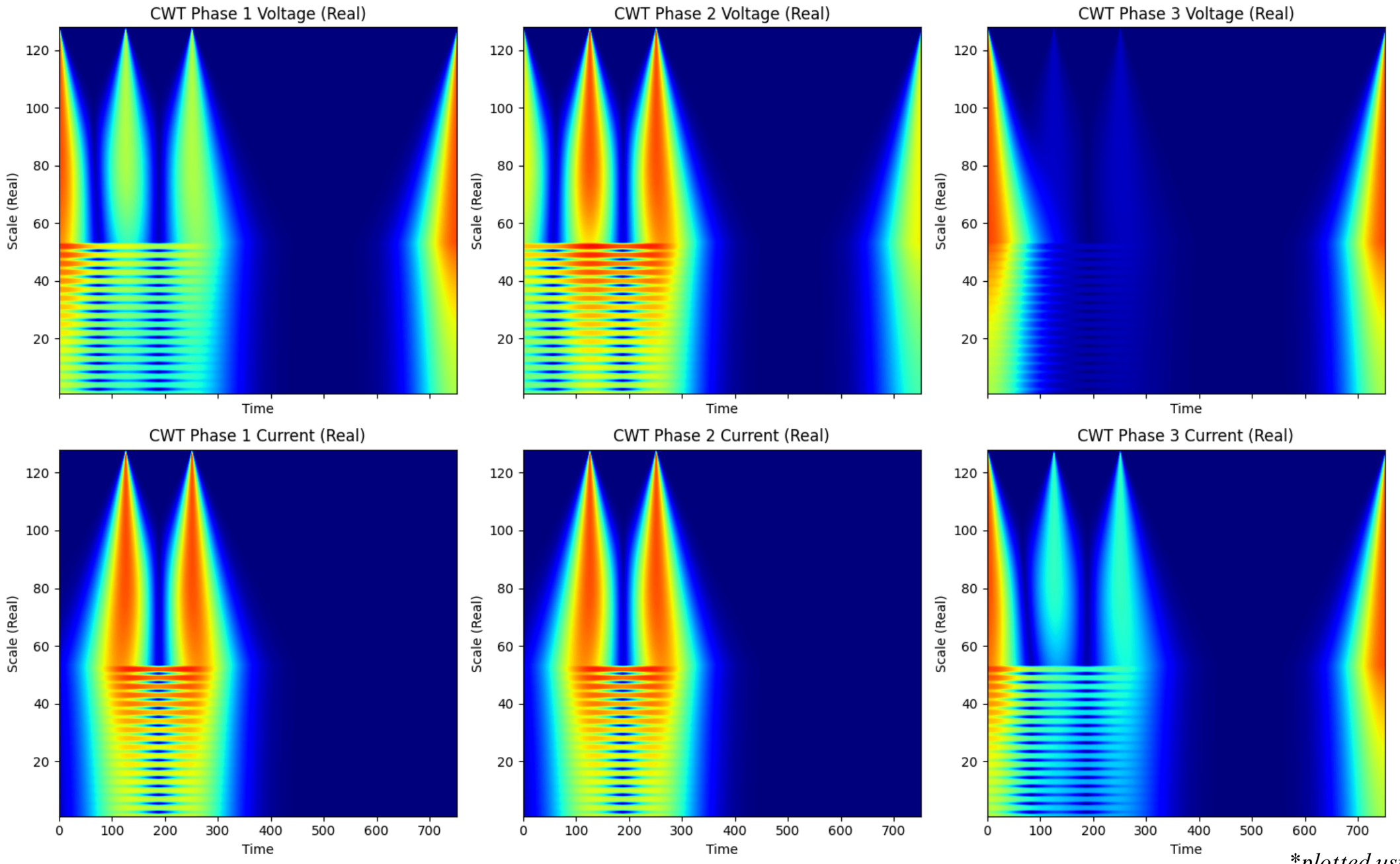
4. Wavelet Transform - Data Loading

Key features of a wavelet transform:

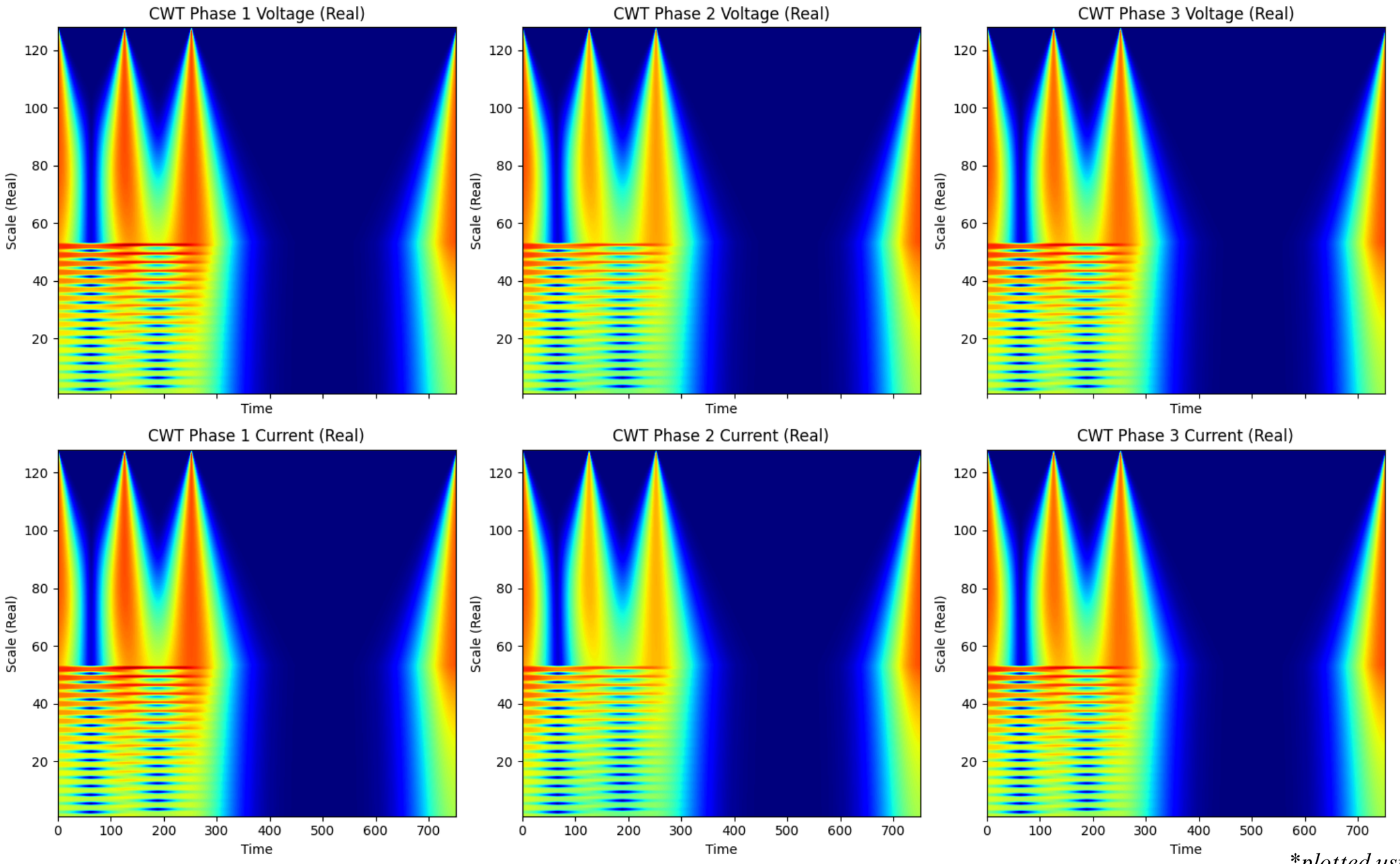
- **Multi-resolution:** The WT provides a representation of a signal with good time and frequency localization.
- **Fast:** The fast wavelet transform is computationally very fast.
- **Separates details:** The WT can separate fine details in a signal.

Idea behind using CWT:

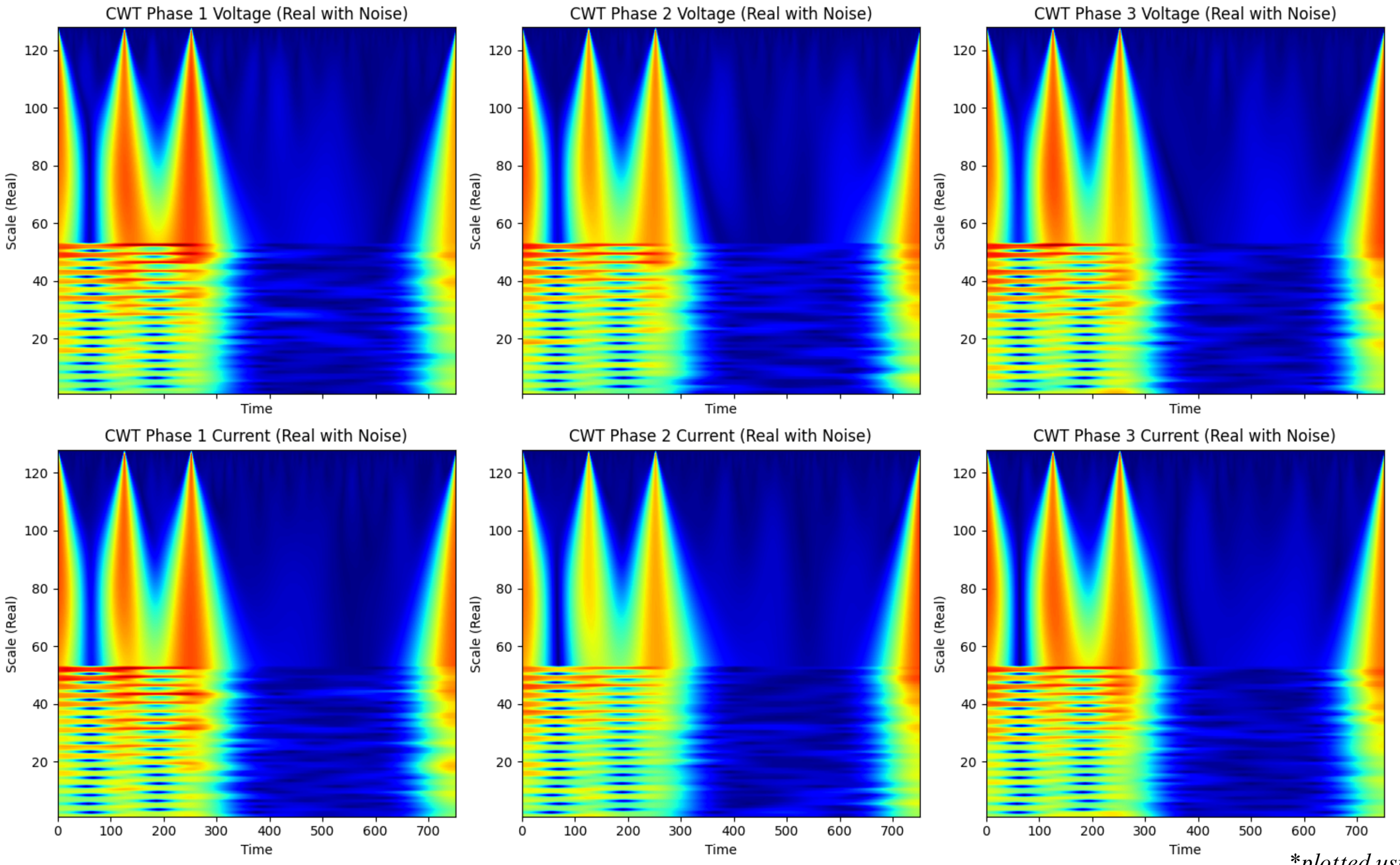
- To transform Time Domain series data into Time-Frequency Domain Image
- This image can make use of both CNN & RNN (Spatial, Spectral & Temporal features can be used)
- Better resolution than STFT.



*plotted using python script



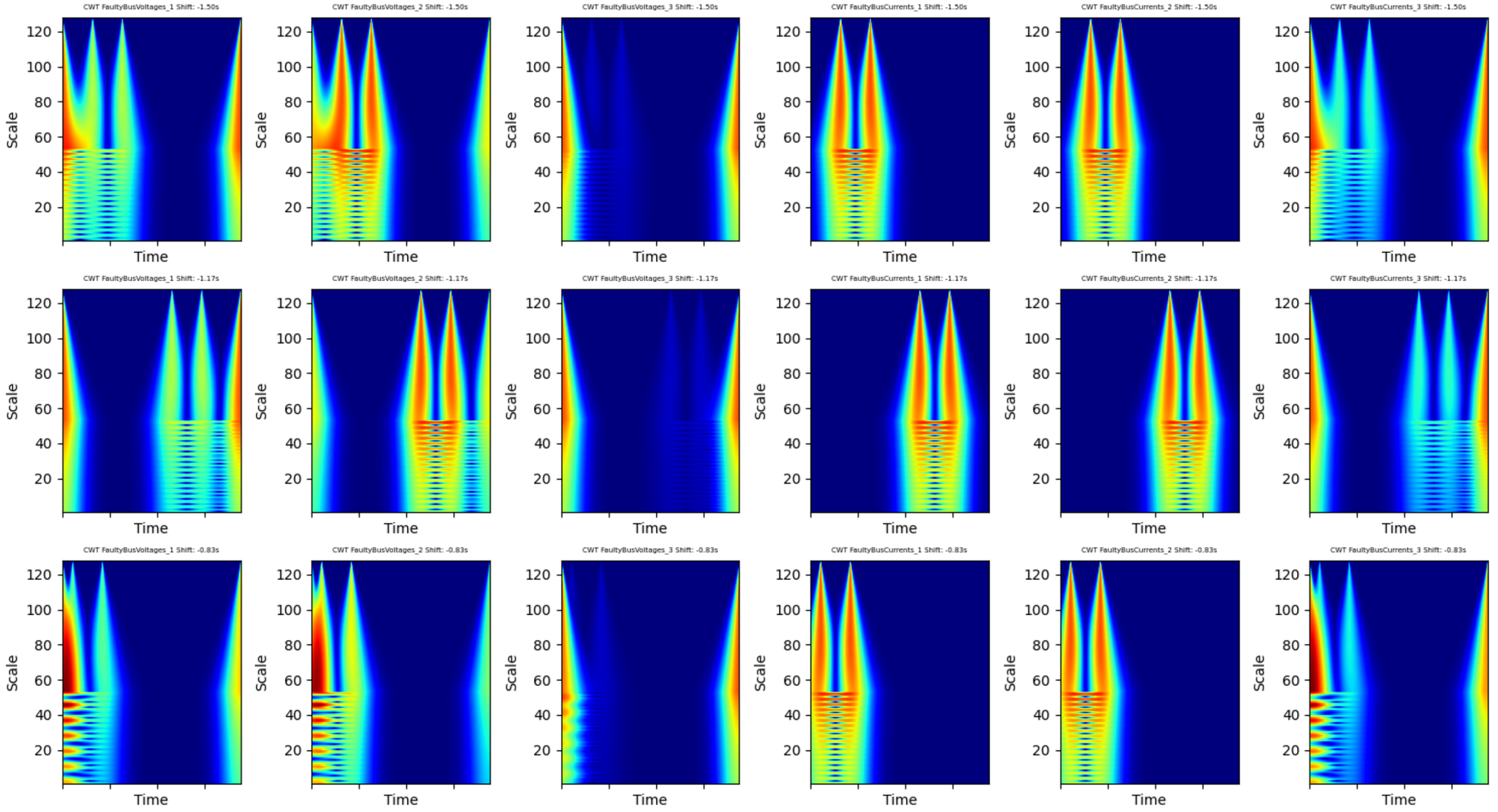
*plotted using python script



*plotted using python script

Bus: Bus_633 - Fault Type: AB - CWT with Time Shifts

Re(Va)	Re(Vb)	Re(Vc)	Re(Ia)	Re(Ib)	Re(Ic)

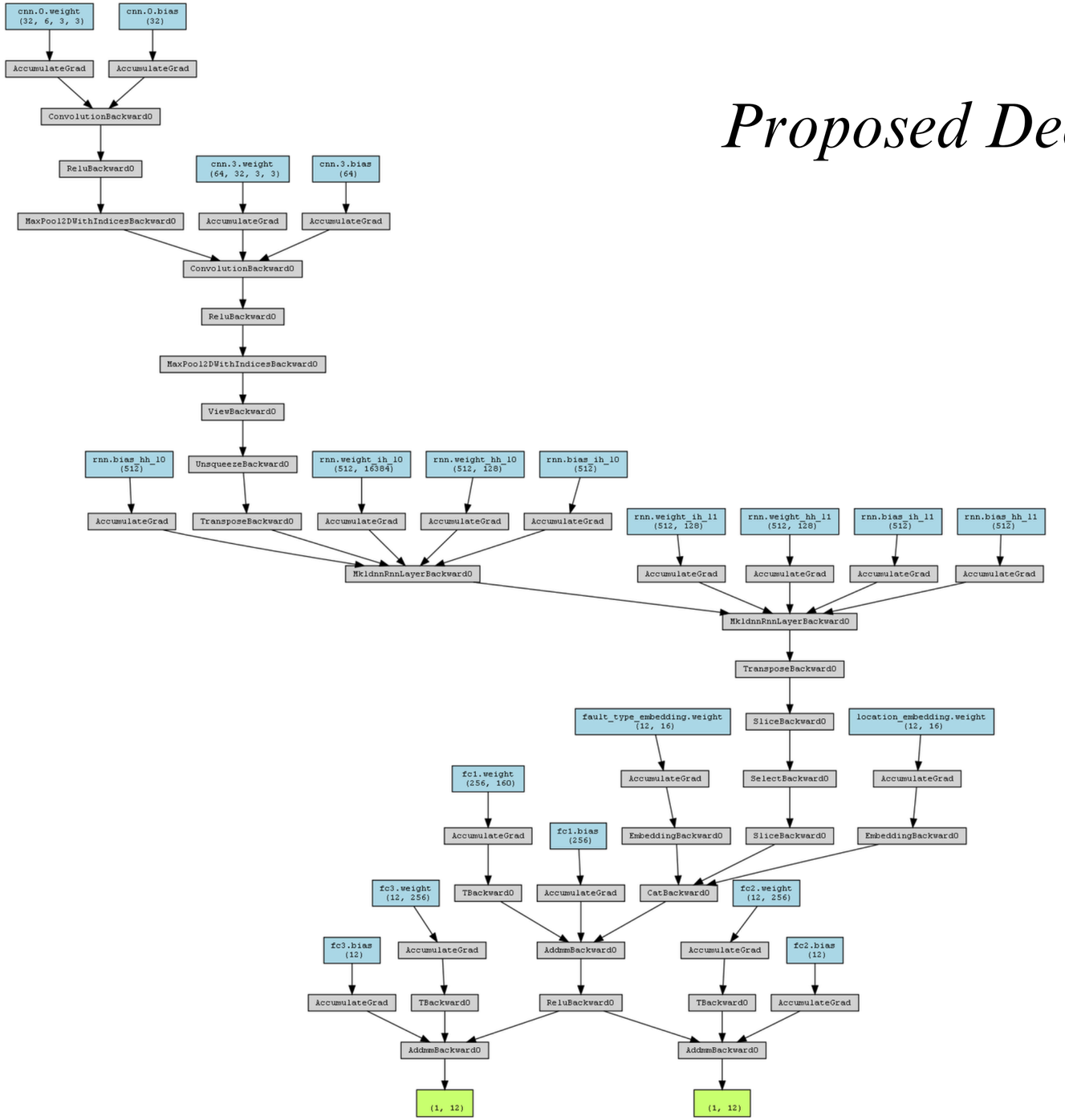


**plotted using python script*

5. Proposed Deep Learning Model

- Based on Time Shifted data augmentation of CWTs of **6 features** i.e **3 phase voltages and 3 phase currents** of each fault at each bus, 6 different tensors are given as input for that particular fault type, bus.
- Fault type and location information are **encoded** into this 6 tensors.
- This **6 set - label encoded inputs** are **stacked** together and **randomized** with other inputs.
- A **hybrid** model using **CNN** and **RNN** is made with **6 inputs** and **2 output groups** for fault categorization and location, each containing **12 outputs**.
- **Two different loss functions** are defined for these two tasks.

Proposed Deep Learning Model



**Visualized using torchviz & IPython.display*

6. Future Scope

- First level of training for this model was done using **bus level measurements.**
- The second level of training on the same best saved model is to be done by **feeder level measurements** so that the model can be flexible to predict with any level of information.
- This model can be used in **real time fault** detection and isolation by sending continuous input in time windows.
- The current model with varied scales of CWT computation will take **0.012 to 0.11 seconds on CPU.**
- With further robust training on a bigger distribution system and inference optimization this time can be significantly reduced and can be used for foreca

7. References

- Goni, M.O.F., Nahiduzzaman, M., Anower, M.S., Rahman, M., Islam, M.R., Ahsan, M., Haider, J., & Shahjalal, M. (2023). Fast and Accurate Fault Detection and Classification in Transmission Lines using Extreme Learning Machine. e-Prime - Advances in Electrical Engineering, Electronics and Energy, 3, 100107.

- KingArthur000. (2020). Electrical Fault Detection and Classification [GitHub Repository]. GitHub. https://github.com/KingArthur000/Electrical-Fault-detection-and-classification

- Kar, S.S., Das, T.K., & Islam, M.T. (2019). Transmission Line Fault Detection and Classification Using Machine Learning Approaches. International Journal of Engineering Research and Technology, 8(12), 450-455.

- Liu, Y., Jiang, H., & Song, W. (2020). Fault Detection in Electrical Transmission Lines Using Convolutional Neural Networks. IEEE Transactions on Power Delivery, 35(4), 1797-1804.

- Wang, H., Lu, L., & Zhou, Z. (2020). Deep Learning Based Transmission Line Fault Detection and Classification Using Wavelet Transform. Electric Power Components and Systems, 48(1-2), 98-110.