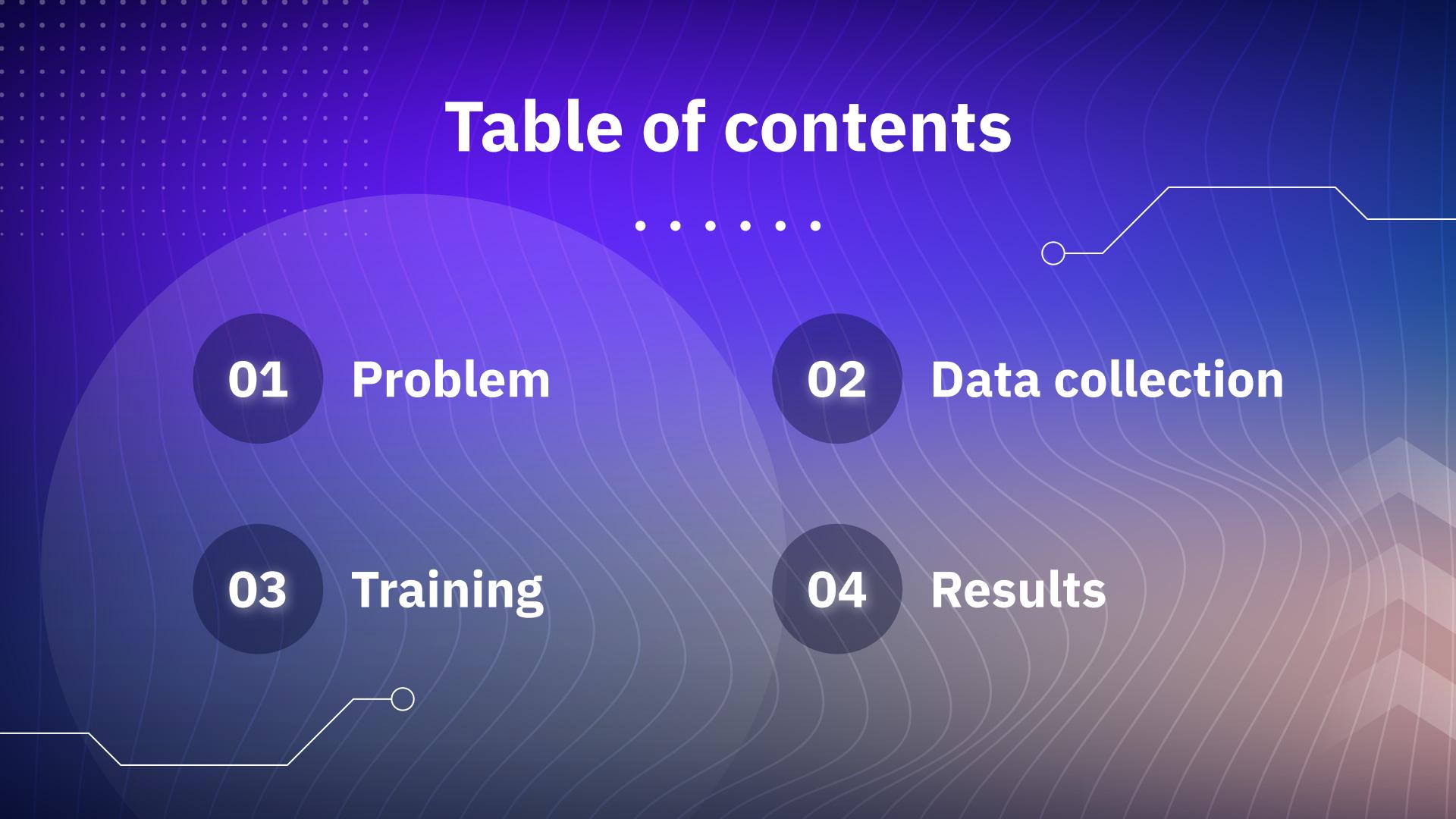




# Application of ML to calculate spin

– Tarek Nabih

# Table of contents



.....

**01 Problem**

**02 Data collection**

**03 Training**

**04 Results**



01

# The Problem



# **Calculate the spin of molecules based on given parameters, plot the relation between a parameter and spin**





# Current method:

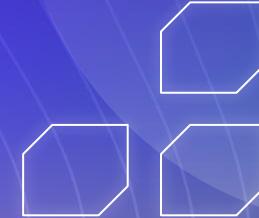




```
[base] tarek_nabih@Tareks-MacBook-Air round 2 % spinev Tarek
SpinEvolution 6.2.5 Double Precision Unlimited Edition
Experiment loaded
Using clust0 mode for splitting the computation
0: scan & ave dimensions point: 1
1: scan & ave dimensions point: 1
2: scan & ave dimensions point: 1
3: scan & ave dimensions point: 1
4: scan & ave dimensions point: 1
5: scan & ave dimensions point: 1
6: scan & ave dimensions point: 1
7: scan & ave dimensions point: 1

clock 40.11
```

- Not efficient
- Would take dozens of minutes to plot



02

# Data collection



To train the machine learning model  
we had to collect big data sets

.....



```
8
9 # Define the headers and their corresponding ranges
10 headers = ['H1_freq', 'spinning_freq' , 'power_1_1_1' , 'cs_beta_2', 'ss_iso_1_2',
11 ranges = [(50,1300),(0.05,80),(0,1500),(0,180),(-600e3,600e3),(-1200e3,1200e3),(0,
12
13 # Generate the input file
14 with open('input.txt', 'w') as f:
15     f.write(' '.join(headers) + '\n')
16 for _ in range(250000):
17     values = [str(np.random.uniform(low, high)) for low, high in ranges]
18     f.write(' '.join(values) + '\n')
19
20
21
```

For every parameter, a random was  
generated and the corresponding  
spin output was calculated via spin  
evolution

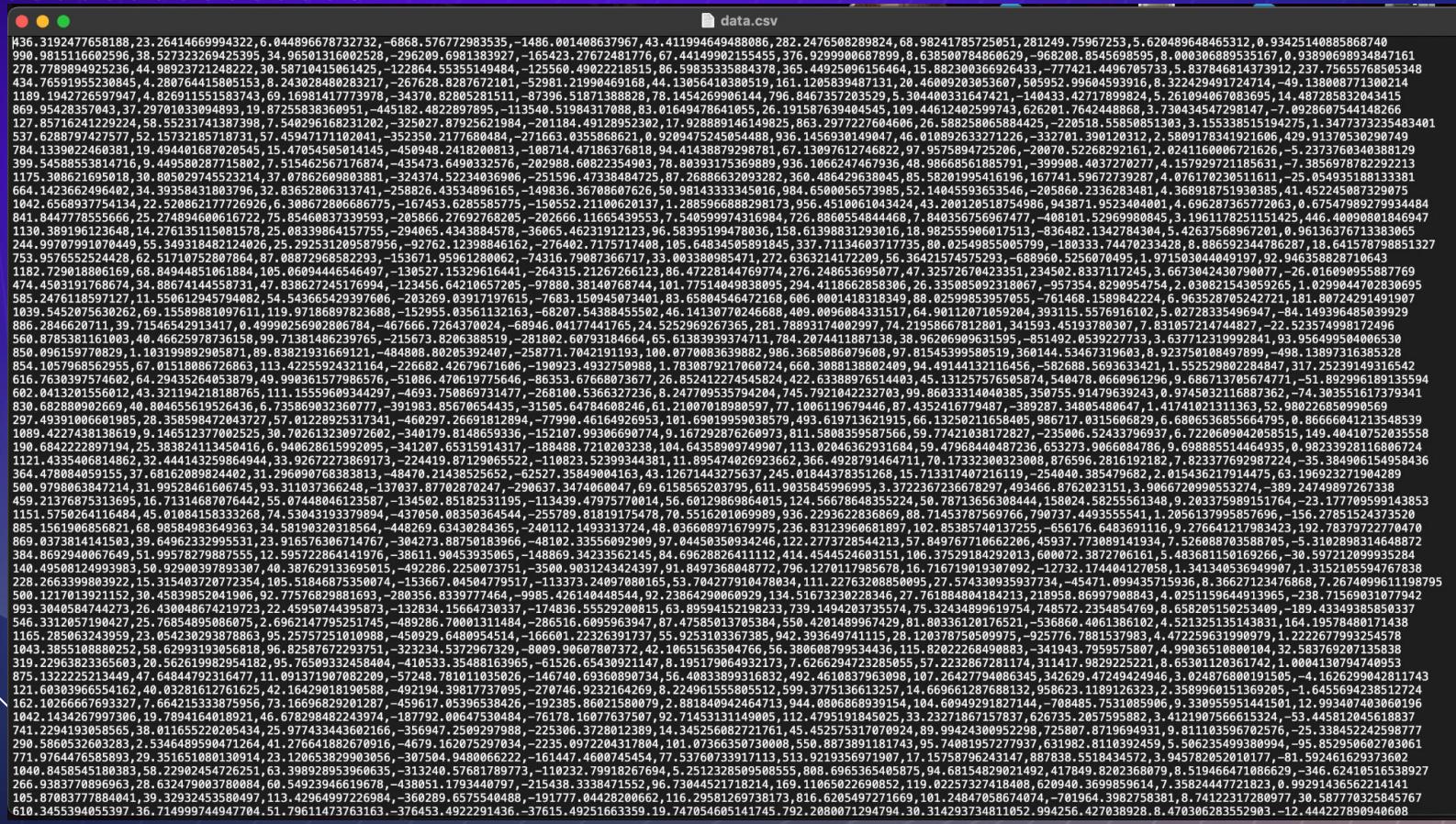


# NYU's supercomputer was used to generate the data

```
tmn8829@login2:/scratch/tmn8829$ squeue
  JOBID PARTITION      NAME      USER ST      TIME  NODES NODELIST(REASON)
1894397    compute      file tmn8829  R  11:51:16      9 cn[119,123-124,134-135,137-140]
1894396    compute      file tmn8829  R  11:51:28     16 cn[051,053-060,071-076,119]
1894395    compute      file tmn8829  R  11:52:18     14 cn[013-014,172-173,175-177,180-181,185-189]
1894392    compute      file tmn8829  R  11:54:13     18 cn[010-012,077-083,085-092]
tmn8829@login2:/scratch/tmn8829$
```

The concept of parallelism was useful in computing the results.

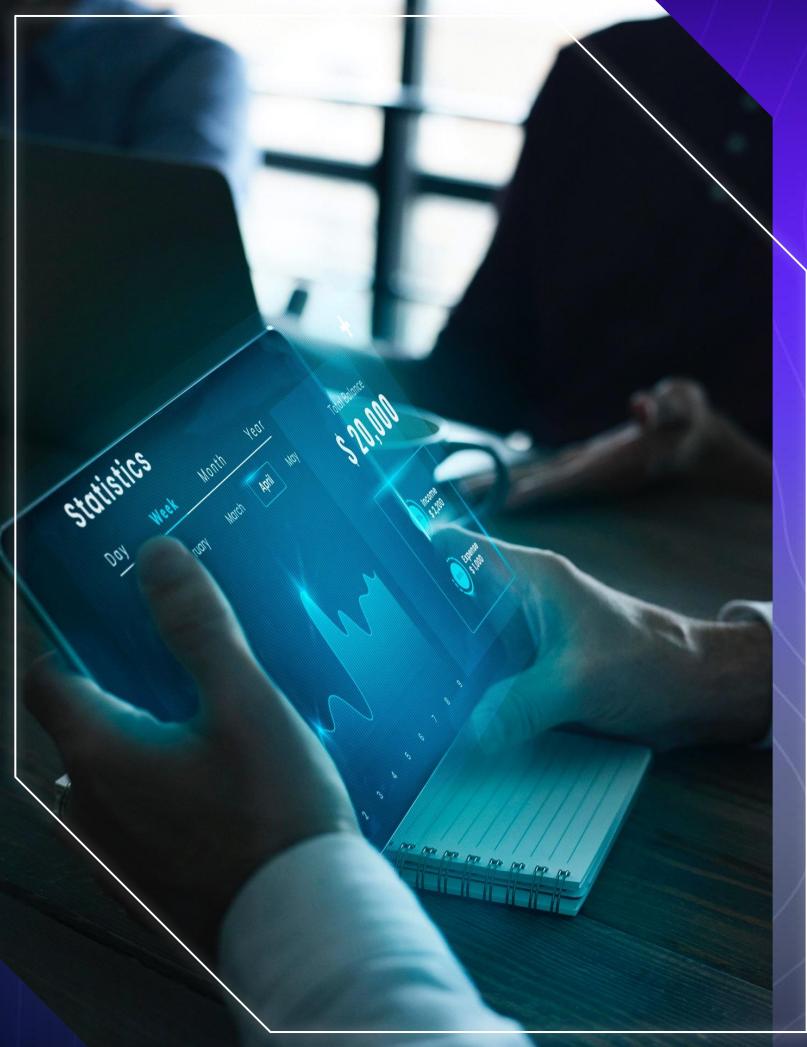




**Around 250,000 lines of data was generated, preparing the first version of the model to be trained.**

03

# Model training



# A picture always reinforces the concept

Images reveal large amounts of data, so remember: use an image instead of a long text. Your audience will appreciate it

# Models tried

```
# Split the data into a training set and a test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Define the models
models = {
    'Linear Regression': LinearRegression(),
    'Decision Tree': DecisionTreeRegressor(),
    'Random Forest': RandomForestRegressor(),
    'Gradient Boosting': GradientBoostingRegressor(),
    'Support Vector Regression': SVR()
}
```

**Initially 5 models were tested on the same number of dataset.**

```
Linear Regression Mean Squared Error: 79624.36022025623
Decision Tree Mean Squared Error: 68327.38474590138
Random Forest Mean Squared Error: 31843.975350144592
Gradient Boosting Mean Squared Error: 42338.93848021941
Support Vector Regression Mean Squared Error: 46424.061255180524
best model is Random Forest
```

# Measuring accuracy

Since it's a regression problem, percentage form couldn't be used to depict accuracy, so instead Mean squared error was used as the loss function



**Random forest was used**



**Model's size created  
problems**



model.pkl



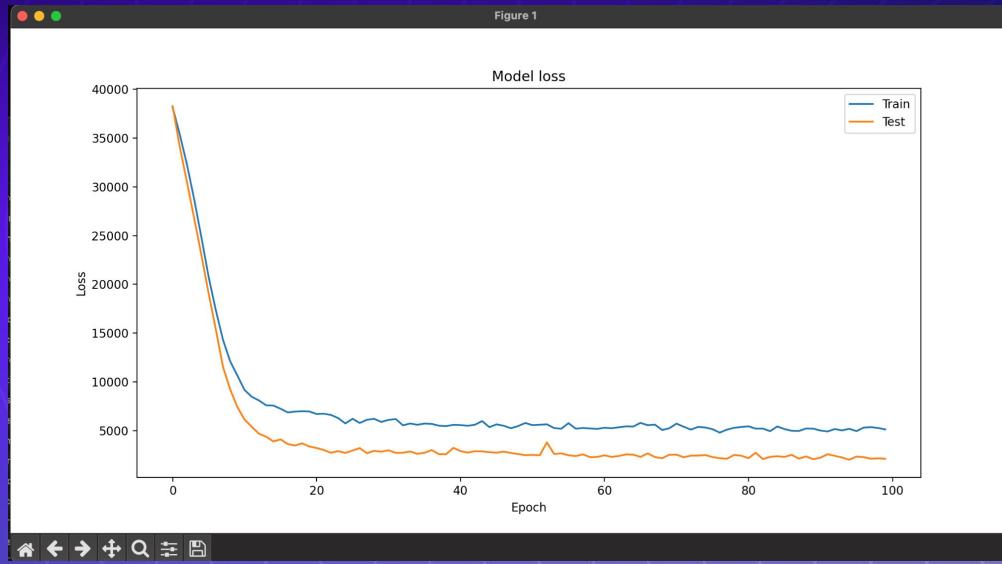
2.05 GB

7/12/2023 9:14:32 PM

```
# Define the neural network model with L2 regularization and increased dropout
model = Sequential([
    Dense(32, activation='relu', input_shape=(10,), kernel_regularizer=l2(1)),
    BatchNormalization(),
    Dropout(0.05),
    Dense(64, activation='relu', kernel_regularizer=l2(1)),
    BatchNormalization(),
    Dropout(0.6),
    Dense(16, activation='relu', kernel_regularizer=l2(1)),
    BatchNormalization(),
    Dropout(0.05),
    Dense(1)
])
```

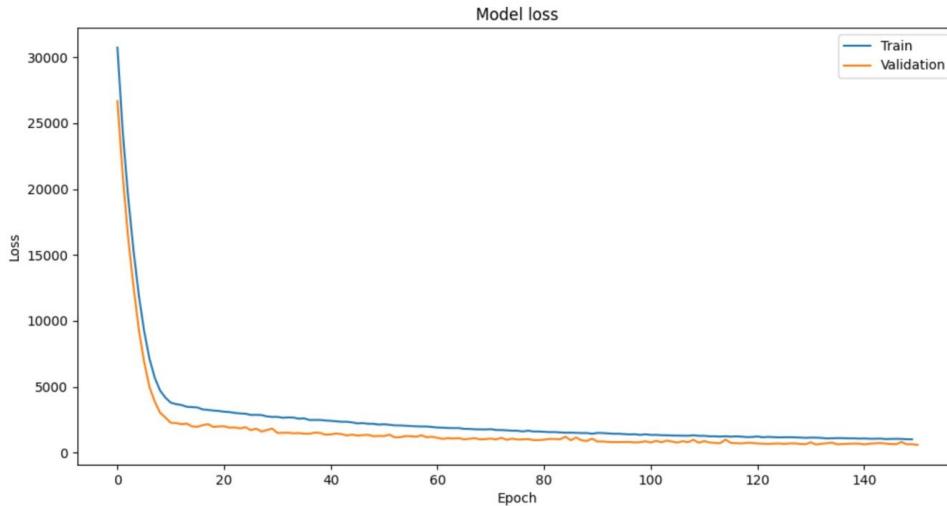
Instead a deep neural  
network was used

- 
- 
- 
- 
- 



Achieved better results than other machine learning algorithms, achieving 2130 MSE

- 
- 
- 
- 
- 



A lot of tweaks were made to achieve MSE of 331

04

# Results

- 
- 
- 
- 
- 

Magnetic field (100-1200):

MAS (0.1-70):

g-orientation (0-120):

e-e-J-coupling (-500000.0--0.0):

e-e-dipolar (-300000.0--0.0):

dipolar orientation (0-120):

hyperfine-coupling (0.0-1000.0):

hyperfine-orientation (0-120):

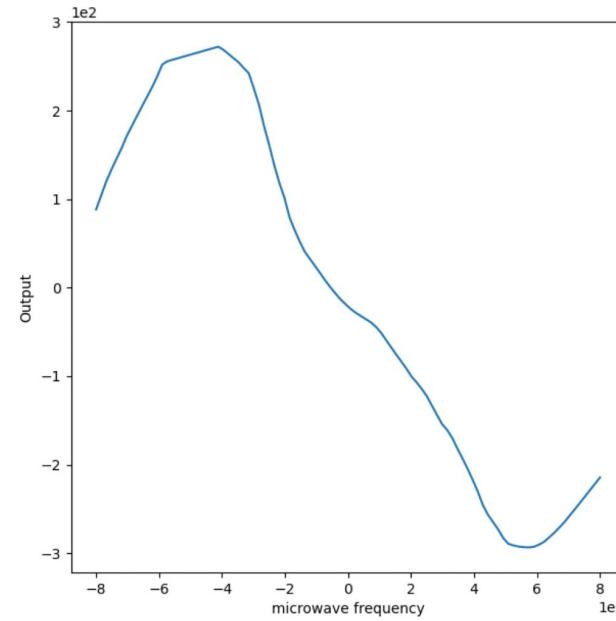
microwave frequency (-800000.0-800000.0):

e-relaxation (0.5-10):

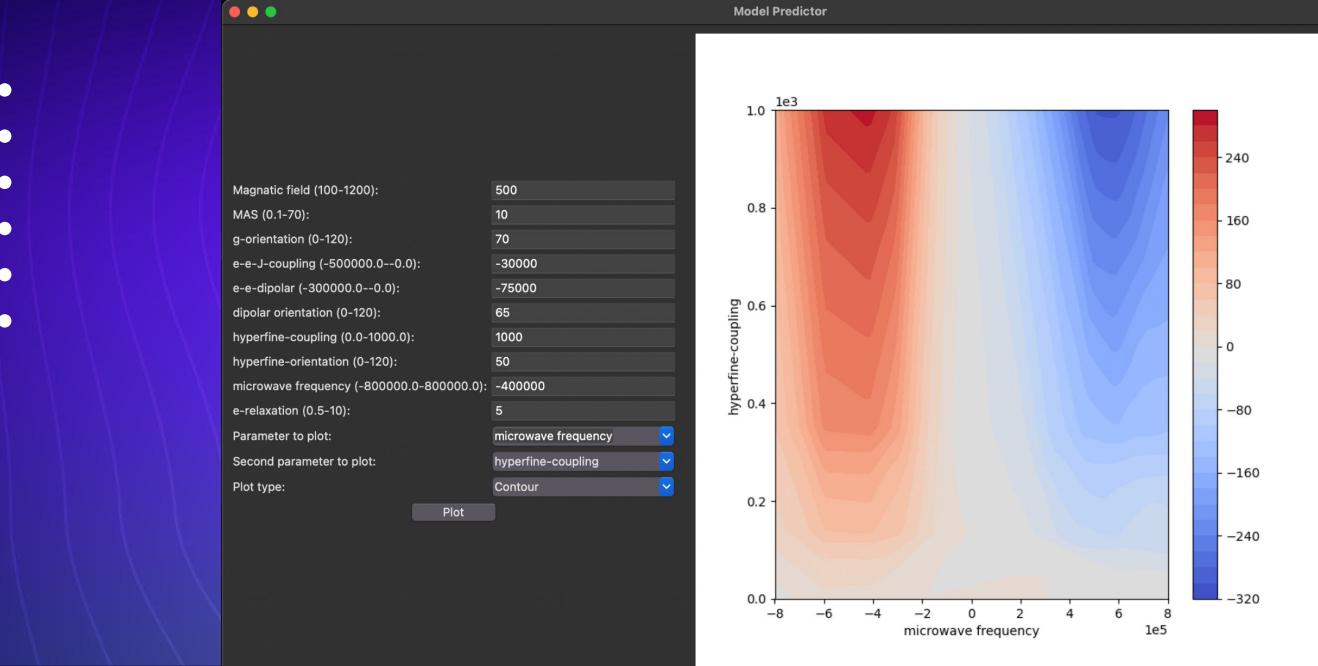
Parameter to plot:

Second parameter to plot:

Plot type:



Using tkinter & matplotlib, UI was developed to plot parameters & spin



And later contour plot was implemented to depict relations

# Current state?

# Thanks!

Do you have any questions?

tmn8829

+971565340617

