0803-sh_1qm_1

August 3, 2023

[ ]:
```
initial_path = 'peptide-QML/'
initial_path = '../'
```

[ ]:
```
day = "0803"
name_notebook = day + "-sh_1qm.ipynb"
```

[ ]:
```
import numpy as np
```

[ ]:
```
import sys
sys.path.append(initial_path)

%load_ext autoreload
%autoreload 2
from my_code import functions as f
from my_code import pytorch_model as ptm
from my_code import quantum_nodes as qn
from my_code import pytorch_wrappers as pw
```

1 Data

[ ]:
```
file_path = initial_path + 'data/energies/PET/generated/
 ↪bb14_Strings_Energies_10_000_4_aa.txt'  # Replace with the actual path to␣
 ↪your 'data.txt' file
string_list, number_list = f.read_data_file(file_path)
score_list = np.array(number_list)/1000
vector_list = np.array([f.string_to_vector(string) for string in string_list])␣
 ↪# one hot encoding
```

[ ]:
```
X, Y, X_validation, Y_validation = f.create_validating_set(vector_list,␣
 ↪score_list, percentage=0.1)

X = X.reshape(X.shape[0], X.shape[1]*X.shape[2]) # flatten
X_validation = X_validation.reshape(X_validation.shape[0], X_validation.
 ↪shape[1]*X_validation.shape[2]) # flatten
```

[ ]:
```
# Define the dataset
input_data = ptm.torch.tensor(X, dtype=ptm.torch.float64)
```

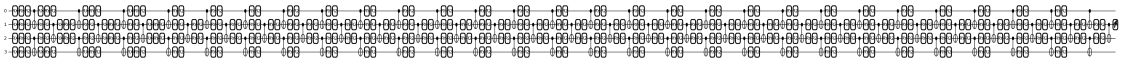
```
target_data = ptm.torch.tensor(Y, dtype=ptm.torch.float64).view(-1, 1)

# Define the validation set
input_validation = ptm.torch.tensor(X_validation, dtype=ptm.torch.float64)
target_validation = ptm.torch.tensor(Y_validation, dtype=ptm.torch.float64).
 ↪view(-1, 1)
```

2 Quantum node

[ ]:
```
n_aminoacids = len(string_list[0])
```

[ ]:
```
quantum_layer = qn.circuit(
    n_qubits = n_aminoacids,
    device = "default.qubit.torch",
    device_options = {'shots': None},
    embedding = qn.parts.AngleEmbedding,
    embedding_ansatz = qn.parts.Ansatz_11,
    block_ansatz = qn.parts.Ansatz_11,
    measurement = qn.parts.Measurement('Z', 1),
    embedding_n_layers = 3,
    block_n_layers = 25,
    wrapper_qlayer = pw.QLayer,
    # wrapper_qlayer = None,
)
```

[ ]:
```
quantum_layer.draw(size=(50,3))
```



3 Hybrid model

[ ]:
```
input_dim = input_data.size(1)

n_pre_classical_layers = 4
layers_dim = np.linspace(n_aminoacids, input_dim, 4).astype(int)
```

[ ]:
```
layers = []
for i in range(1, len(layers_dim)):
    layers += [ptm.nn.Linear(layers_dim[-1*i], layers_dim[-1*(i+1)]), ptm.nn.
 ↪ReLU()]
layers += [ptm.nn.Linear(layers_dim[0], layers_dim[0])]
layers += [quantum_layer()]
```

```
# layers += [nn.Linear(1, 1)]
# layers += [nn.Linear(2, 4), nn.ReLU()]
# layers += [nn.Linear(4, 1)]
```

[ ]:
```
# Create model and set data:
model = ptm.pytorch_model(
    layers,
    save_options = {'initial_path': initial_path, 'name_notebook':␣
 ↪name_notebook},
    # keep_track_params=True,
)
model.set_data(
    data_X=input_data,
    data_Y=target_data,
    data_X_validation=input_validation,
    data_Y_validation=target_validation
)
```

[ ]:
```
print(model(input_data[0]).item())
print(model(input_data[1]).item())
```

```
-0.014470531463892544
-0.0014800423006265384
```

[ ]:
```
model
```

[ ]:
```
Sequential(
  (0): Linear(in_features=76, out_features=52, bias=True)
  (1): ReLU()
  (2): Linear(in_features=52, out_features=28, bias=True)
  (3): ReLU()
  (4): Linear(in_features=28, out_features=4, bias=True)
  (5): ReLU()
  (6): Linear(in_features=4, out_features=4, bias=True)
  (7): QLayer(
    (qlayer): <Quantum Torch Layer: func=circuit>
  )
)
```

[ ]:
```
model.keep_track_params=True,
```

[ ]:
```
import torch.optim as optim

# train the model
model.train(
    optimizer= optim.Adam,
    optimizer_options={'lr': 0.5},
    num_epochs = 1,
```

```
    batch_size = 32,
)
```

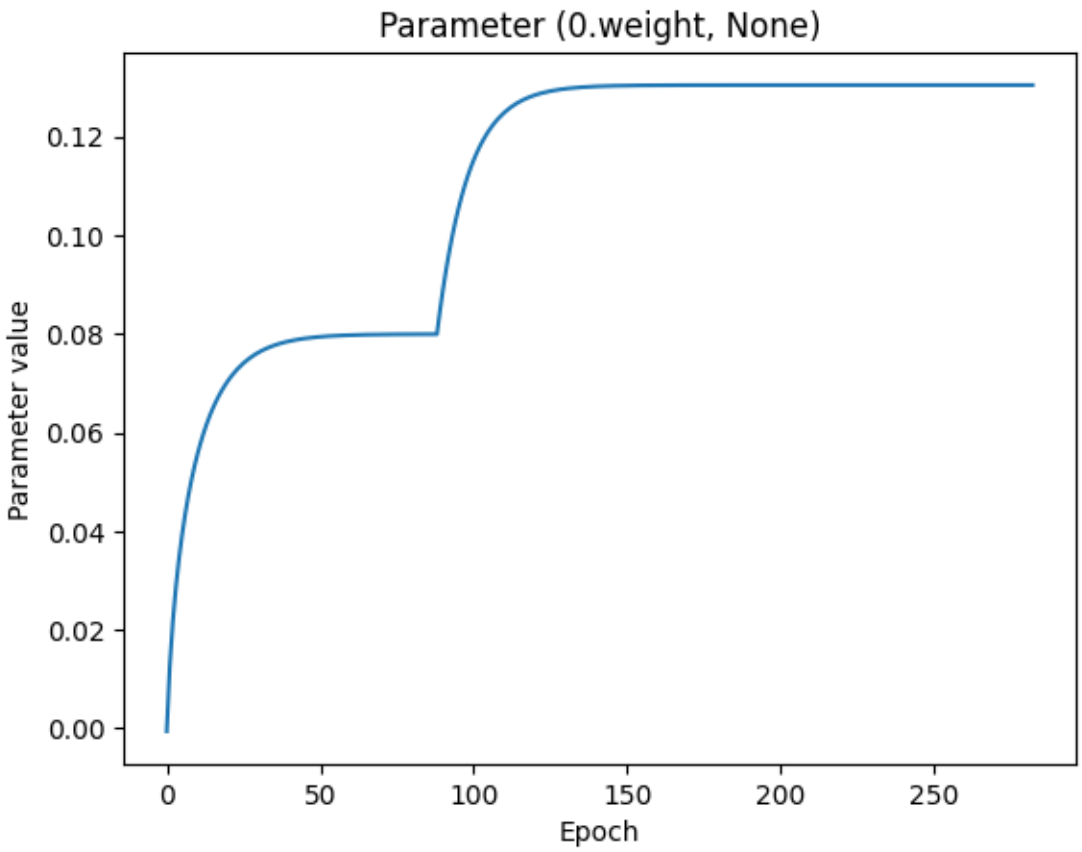
```
Epoch [0/1], Loss: 1.4905, Loss validation: 1.5147
- Epoch [1/1], i: [5888/9000], Loss: 2.9597
```

[ ]:
```
model.save_str()
```

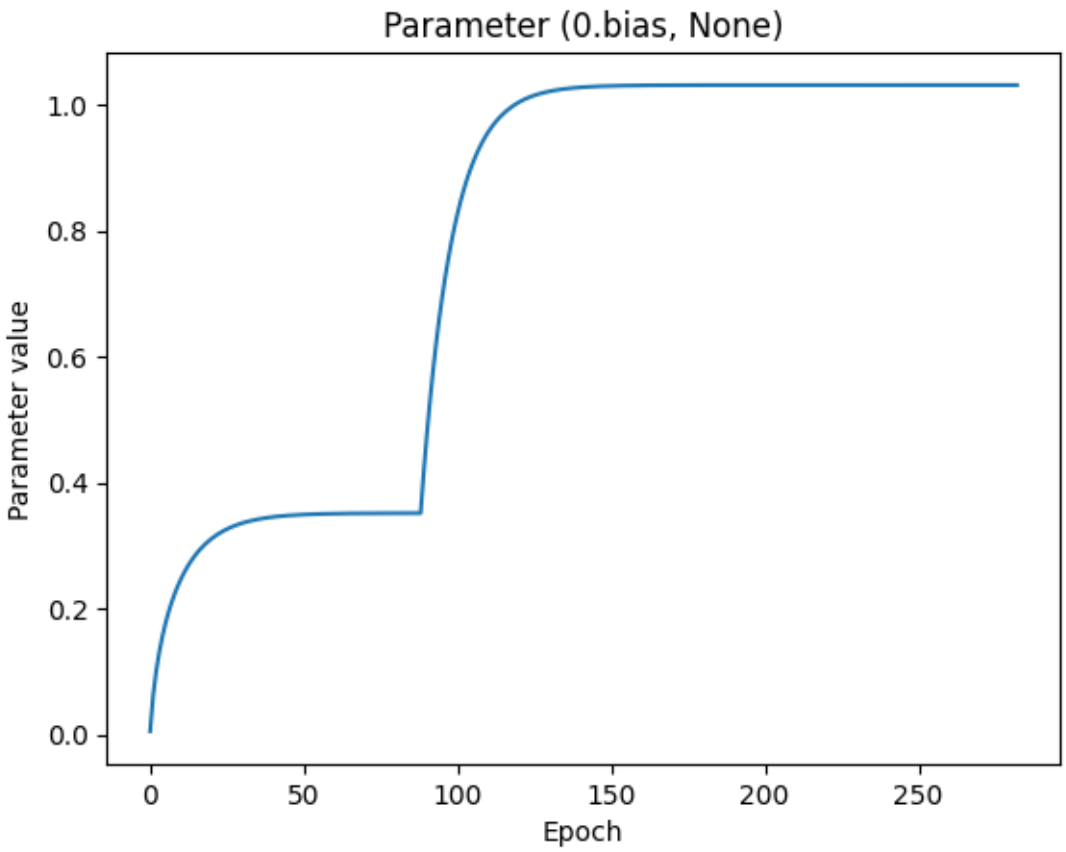
```
Sequential(
  (0): Linear(in_features=76, out_features=52, bias=True)
  (1): ReLU()
  (2): Linear(in_features=52, out_features=28, bias=True)
  (3): ReLU()
  (4): Linear(in_features=28, out_features=4, bias=True)
  (5): ReLU()
  (6): Linear(in_features=4, out_features=4, bias=True)
  (7): QLayer(
    (qlayer): <Quantum Torch Layer: func=circuit>
  )
)
```

[ ]:
```
for layer in model.model.state_dict().keys():
    model.plot_parameter(layer=layer, save=True)
```

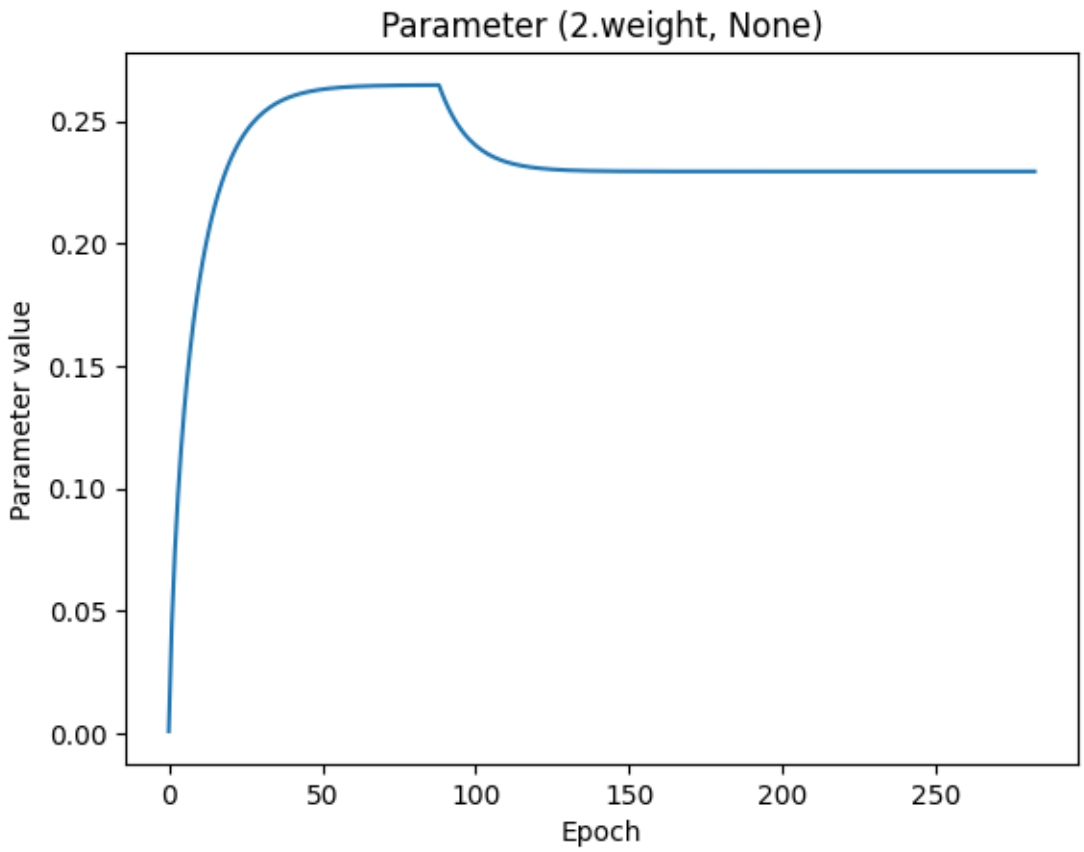
```
Saved in:  ../checkpoints/0803/plots/0803-sh_1qm_parameter_0.weight_None_1.png
```



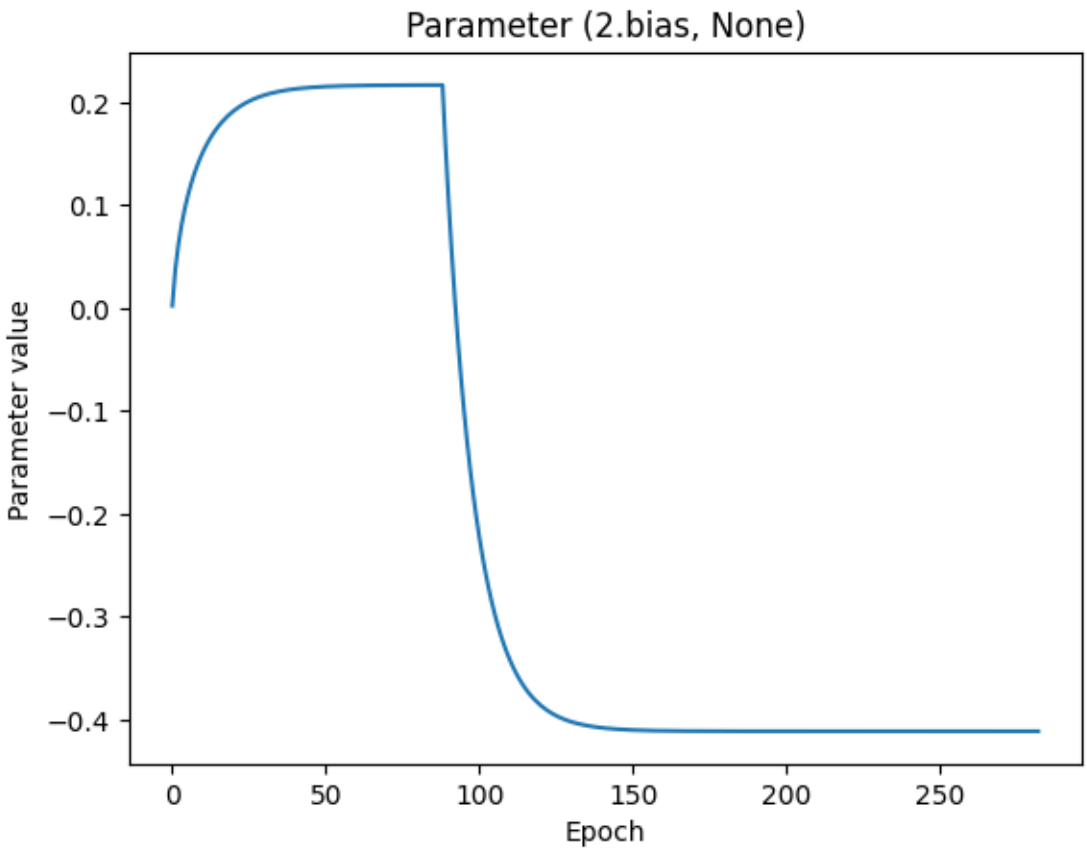
Saved in: ../checkpoints/0803/plots/0803-sh_1qm_parameter_0.bias_None_1.png



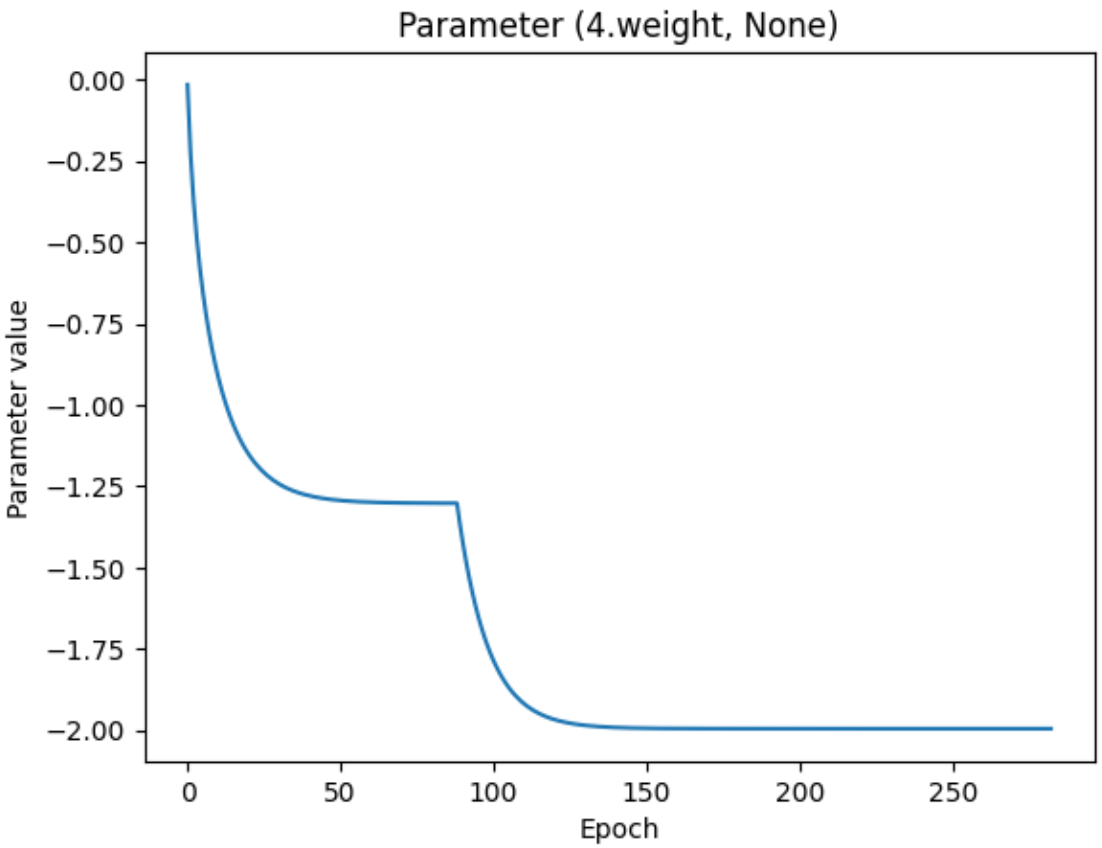
Saved in: ../checkpoints/0803/plots/0803-sh_1qm_parameter_2.weight_None_1.png



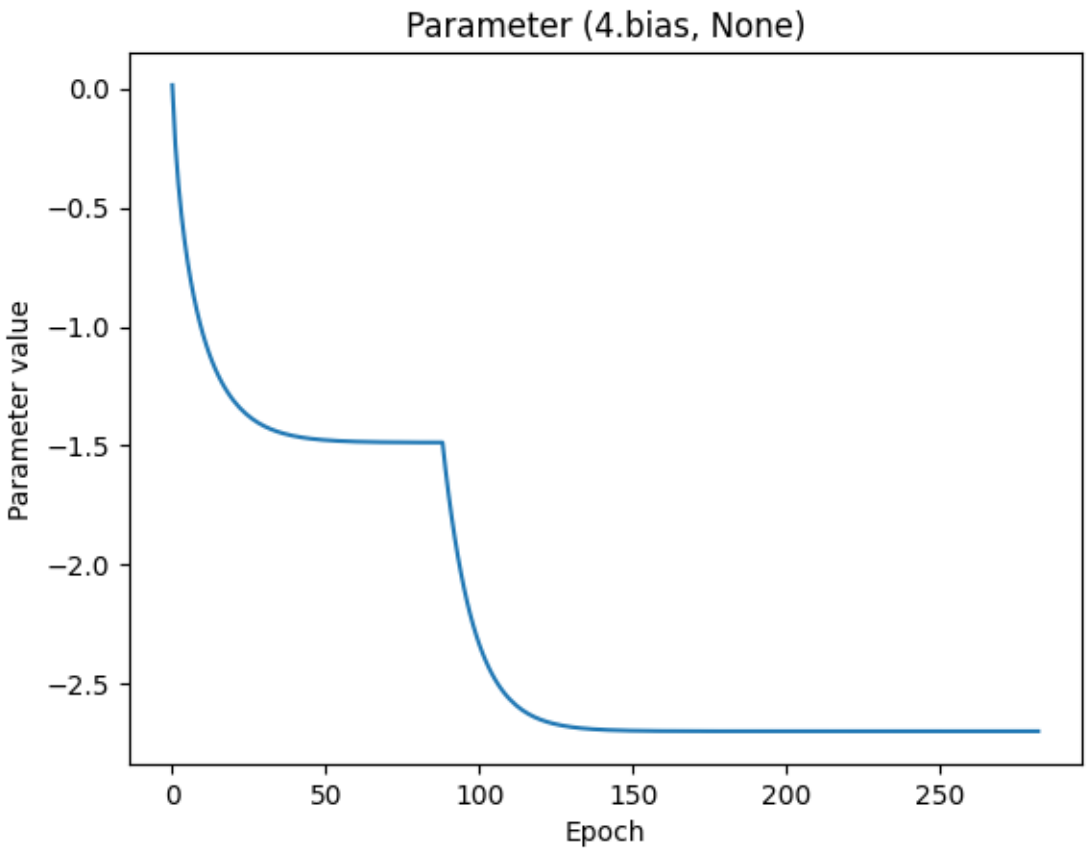

```
Saved in:  ../checkpoints/0803/plots/0803-sh_1qm_parameter_2.bias_None_1.png
```



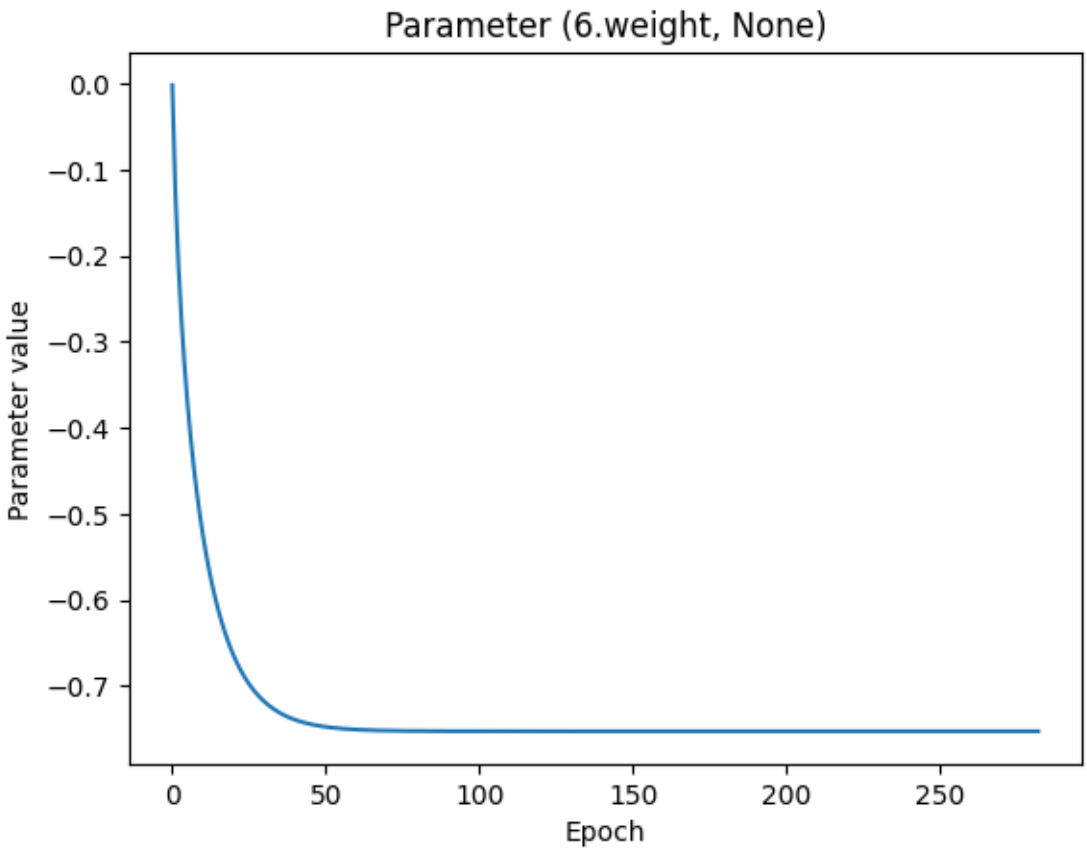
Saved in: ../checkpoints/0803/plots/0803-sh_1qm_parameter_4.weight_None_1.png



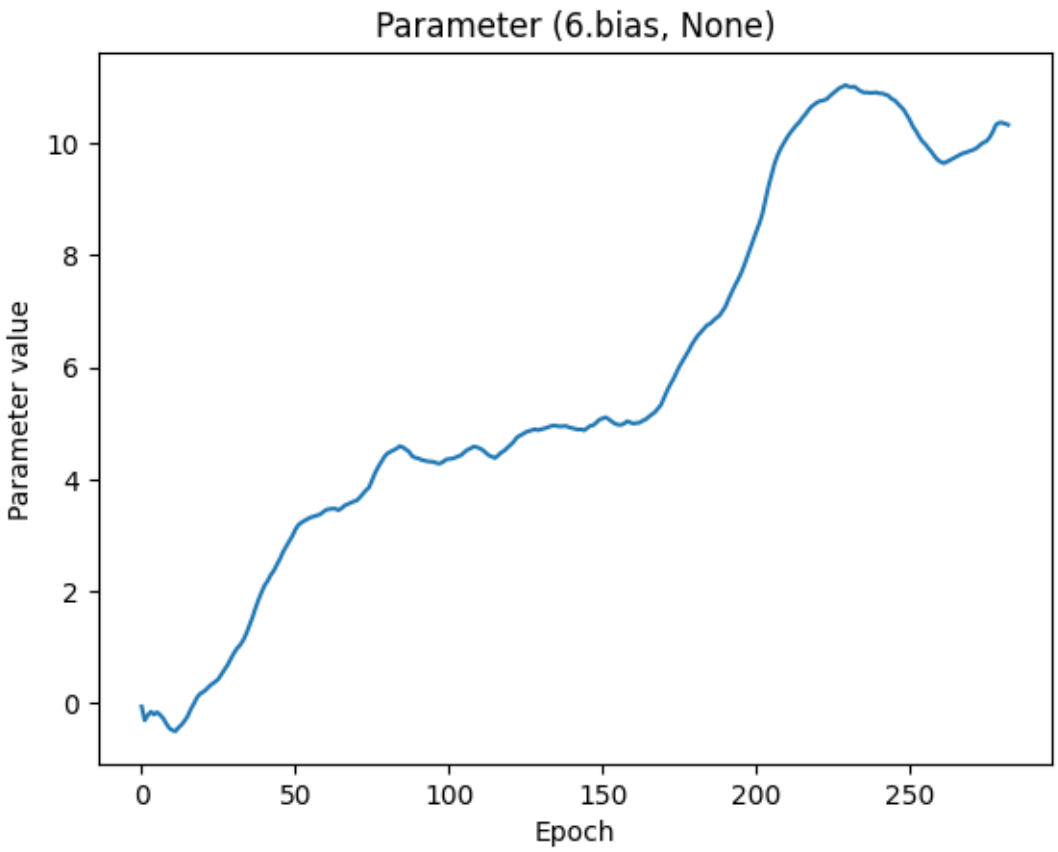
Saved in: ../checkpoints/0803/plots/0803-sh_1qm_parameter_4.bias_None_1.png



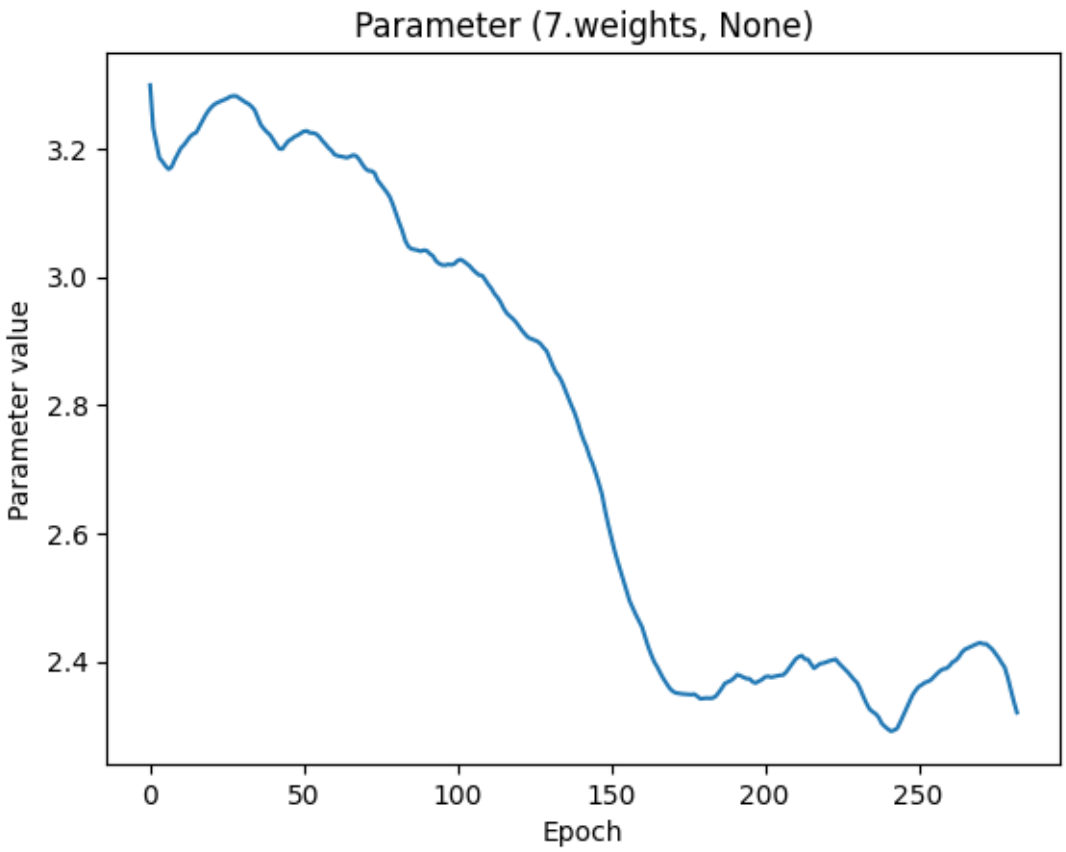
Saved in: ../checkpoints/0803/plots/0803-sh_1qm_parameter_6.weight_None_1.png



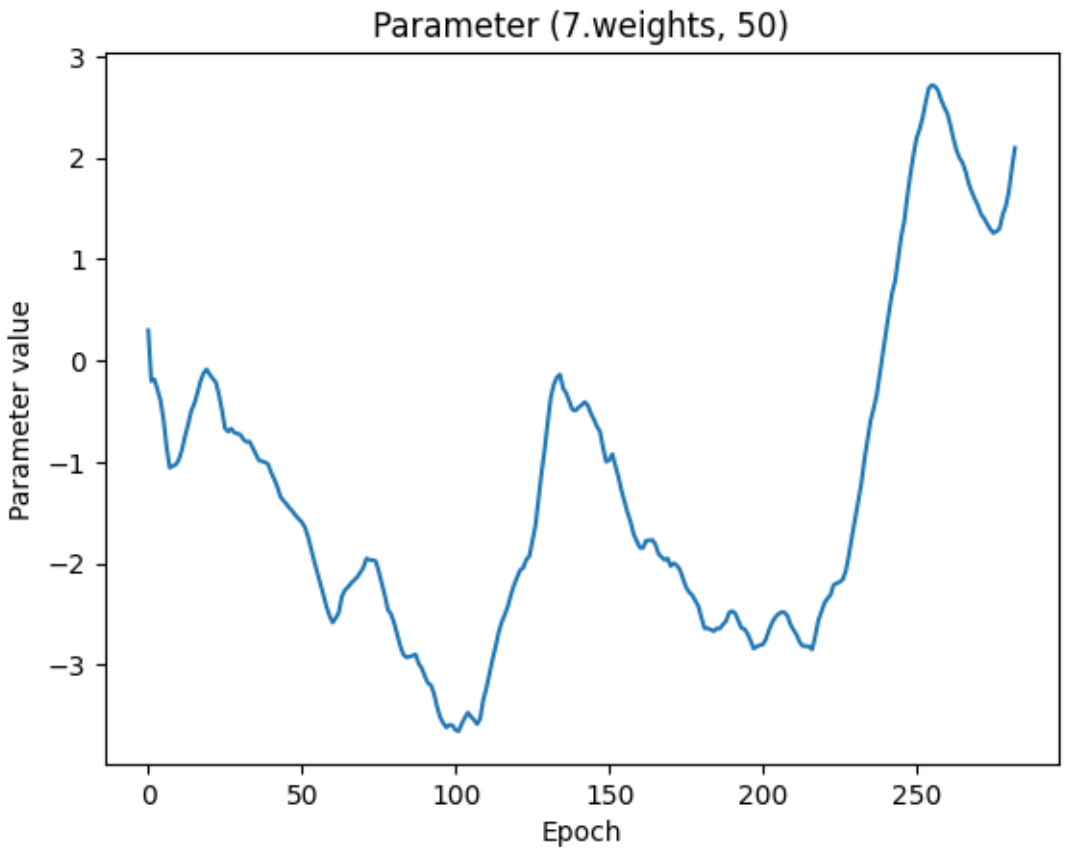

```
Saved in:  ../checkpoints/0803/plots/0803-sh_1qm_parameter_6.bias_None_1.png
```



Saved in: ../checkpoints/0803/plots/0803-sh_1qm_parameter_7.weights_None_1.png



[ ]:
```
model.plot_parameter(layer='7.weights', index=50)
```

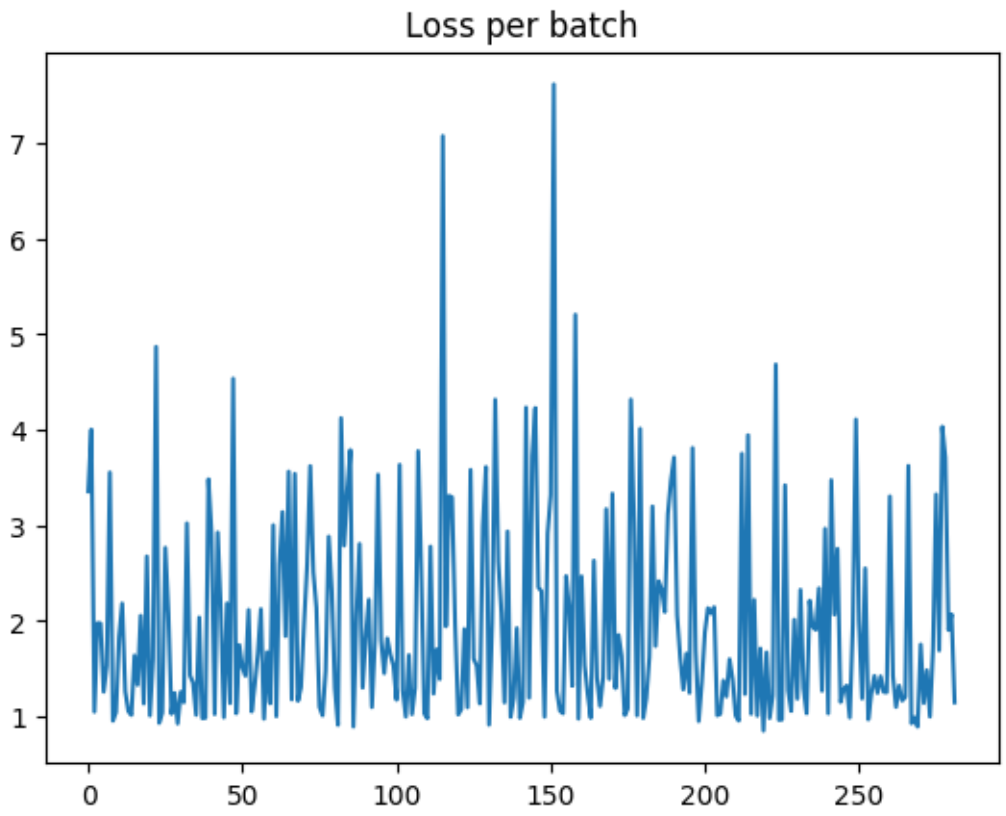


[ ]:
```
model
```

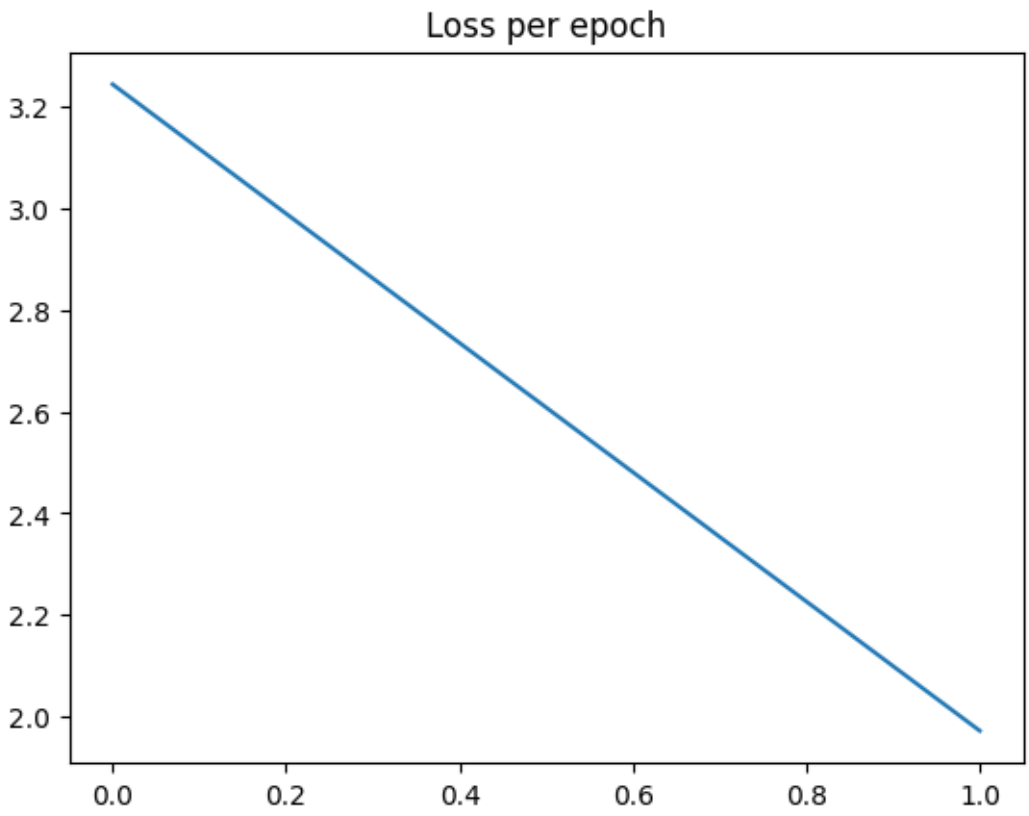
[ ]:
```
Sequential(
  (0): Linear(in_features=76, out_features=52, bias=True)
  (1): ReLU()
  (2): Linear(in_features=52, out_features=28, bias=True)
  (3): ReLU()
  (4): Linear(in_features=28, out_features=4, bias=True)
  (5): ReLU()
  (6): Linear(in_features=4, out_features=4, bias=True)
  (7): <Quantum Torch Layer: func=circuit>
)
```

[ ]:
```
# plot the losses of the trainig loop
model.plot_losses(save=True)
```

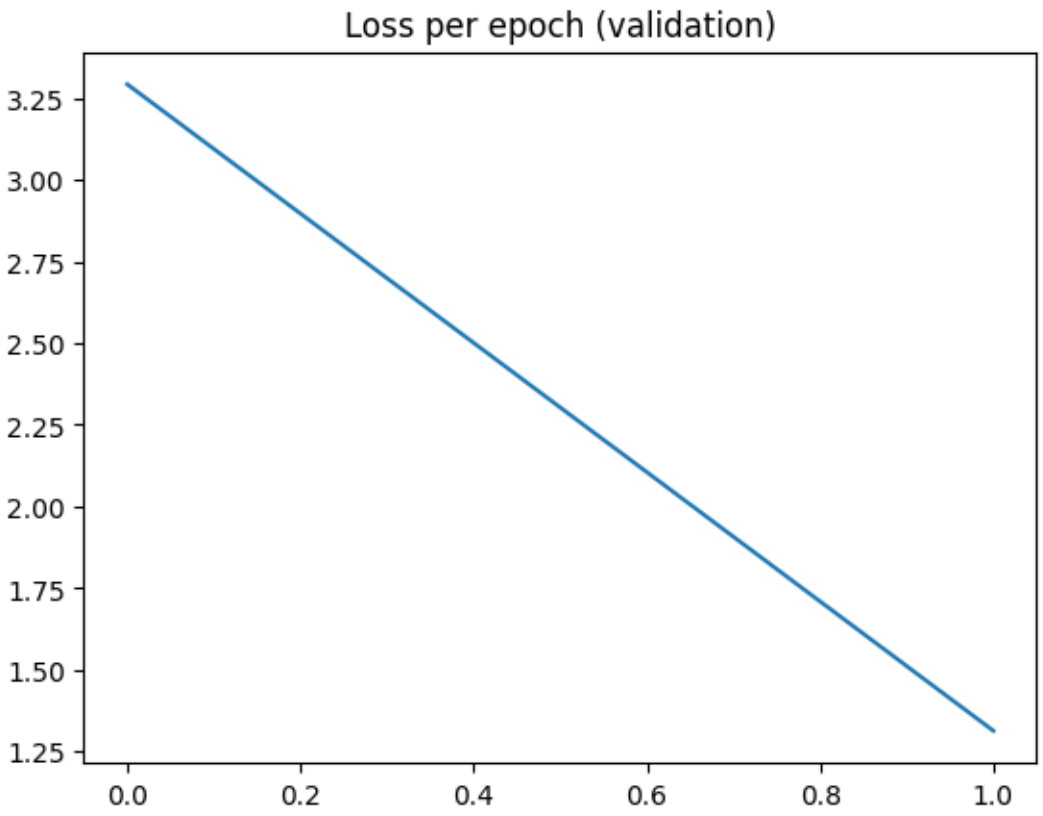
```
Saved in:  ../checkpoints/0803/plots/0803-sh_1qm_losses_batches_1.png
```




```
Saved in:  ../checkpoints/0803/plots/0803-sh_1qm_losses_epoch_1.png
```



Saved in: ../checkpoints/0803/plots/0803-sh_1qm_losses_epoch_validation_1.png



[ ]:
```
model.save_state_dict()
```

```
Model saved as ../checkpoints/0803/models/0803-sh_1qm_1.pth
```

[ ]:
```
# push changes to git
if initial_path != '../':
    !cd peptide-QML && git add . && git commit -m "data trained model" && git␣
 ↪push
```

[ ]:
```
#load model
model.load_state_dict()
```

```
Model loaded from ../checkpoints/0803/models/0803-sh_1qm_1.pth
```

[ ]:
```
# print validation
model.print_validation(save=True, precision=4, percentatge=0.02)
```

```
i: 0,           target: -0.0165,        output: 0.1431,         loss: 9.6800
i: 1,           target: -0.1004,        output: 0.1431,         loss: 2.4251
i: 2,           target: -0.0534,        output: 0.1431,         loss: 3.6773
i: 3,           target: 0.0739,         output: 0.1431,         loss: 0.9353
i: 4,           target: 0.1004,         output: 0.1431,         loss: 0.4254
```

```
i: 5,           target: 0.0332,         output: 0.1431,         loss: 3.3147
i: 6,           target: -0.0569,        output: 0.1431,         loss: 3.5123
i: 7,           target: -0.0926,        output: 0.1431,         loss: 2.5458
i: 8,           target: 0.1444,         output: 0.1431,         loss: 0.0095
i: 9,           target: 0.0269,         output: 0.1431,         loss: 4.3169
i: 10,          target: -0.0756,        output: 0.1431,         loss: 2.8924
i: 11,          target: -0.1129,        output: 0.1431,         loss: 2.2673
i: 12,          target: -0.0641,        output: 0.1431,         loss: 3.2337
i: 13,          target: 0.0306,         output: 0.1431,         loss: 3.6791
i: 14,          target: 0.2133,         output: 0.1431,         loss: 0.3292
i: 15,          target: -0.0793,        output: 0.1431,         loss: 2.8033
i: 16,          target: 0.0933,         output: 0.1431,         loss: 0.5332
i: 17,          target: -0.0293,        output: 0.1431,         loss: 5.8890
i: 18,          target: -0.0732,        output: 0.1431,         loss: 2.9546
i: 19,          target: -0.0956,        output: 0.1431,         loss: 2.4971
Average loss: 0.0579
Saved in:  ../checkpoints/0803/txts/0803-sh_1qm_validation_1.txt
```