②LSTM

=====

目次:

1.LSTMによるsin波の再現

【要約】

・RNNの課題　　→時系列を遡れば遡るほど、勾配が消失していくため、長い時系列の学習が困難。

→勾配消失の解決方法とは、別で、構造自体を変えて解決したものがLSTM。

・勾配消失および勾配爆発の解決方法として、勾配が、1であれば解決できる。

→1にするためにCECを導入。CECは、それまでの入力値及び出力値を記憶するための機能部。

・入力ゲート:入力されたデータの記憶の仕方をCECに指示。

・出力ゲート:どんな風にCECの記憶した情報を使うかを学習。

→入力ゲートと出力ゲート:今回の入力と前回の出力をもとに学習。

・忘却ゲート:CECには、過去の情報が全て保管されているが、過去の情報が要らなくなった場合、削除することはできず、保管され続ける。

→過去の情報が要らなくなった場合、そのタイミングで情報を忘却する機能が必要。

→忘却ゲートを利用。

=====

1.LSTMによるsin波の再現

→学習データにsin波を与え、再現させる

→Kerasで実装

①ライブラリのimport

In [1]:

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Activation
from tensorflow.keras.layers import LSTM
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping
import numpy as np
import matplotlib.pyplot as plt
```

②データの作成(sin波の生成)

In [2]:

```
def sin(x, T=100):
    return np.sin(2.0 * np.pi * x / T)

# sin波にノイズを付与する
def toy_problem(T=100, ampl=0.05):
    x = np.arange(0, 2 * T + 1)
    noise = ampl * np.random.uniform(low=-1.0, high=1.0, size=len(x))
    return sin(x) + noise

f = toy_problem()
```

In [3]:

```
def make_dataset(low_data, n_prev=100):

    data, target = [], []
    maxlen = 25

    for i in range(len(low_data)-maxlen):
        data.append(low_data[i:i + maxlen])
        target.append(low_data[i + maxlen])

    re_data = np.array(data).reshape(len(data), maxlen, 1)
    re_target = np.array(target).reshape(len(data), 1)

    return re_data, re_target


#g -> 学習データ, h -> 学習ラベル
g, h = make_dataset(f)
```

③モデル生成

In [5]:

```
# モデル構築

# 1つの学習データのStep数(今回は25)
length_of_sequence = g.shape[1]
in_out_neurons = 1
n_hidden = 300

model = Sequential()
model.add(LSTM(n_hidden, batch_input_shape=(None, length_of_sequence, in_out_neurons), return_s
equences=False))
model.add(Dense(in_out_neurons))
model.add(Activation("linear"))
optimizer = Adam(lr=0.001)
model.compile(loss="mean_squared_error", optimizer=optimizer)
```

```
/usr/local/lib/python3.7/dist-packages/tensorflow/python/keras/optimizer_v2/optimi
zer_v2.py:375: UserWarning: The `lr` argument is deprecated, use `learning_rate` i
nstead.
  "The `lr` argument is deprecated, use `learning_rate` instead.")
```

④学習

In [6]:

```
early_stopping = EarlyStopping(monitor='val_loss', mode='auto', patience=20)
model.fit(g, h,
          batch_size=300,
          epochs=100,
          validation_split=0.1,
          callbacks=[early_stopping]
          )
```

```
Epoch 1/100
1/1 [==============================] - 3s 3s/step - loss: 0.4617 - val_loss: 0.084
1
Epoch 2/100
1/1 [==============================] - 0s 294ms/step - loss: 0.2695 - val_loss: 0.
0377
Epoch 3/100
1/1 [==============================] - 0s 286ms/step - loss: 0.1462 - val_loss: 0.
2113
Epoch 4/100
1/1 [==============================] - 0s 308ms/step - loss: 0.1381 - val_loss: 0.
2233
Epoch 5/100
1/1 [==============================] - 0s 286ms/step - loss: 0.1141 - val_loss: 0.
1171
Epoch 6/100
1/1 [==============================] - 0s 290ms/step - loss: 0.0683 - val_loss: 0.
0481
Epoch 7/100
1/1 [==============================] - 0s 290ms/step - loss: 0.0501 - val_loss: 0.
0217
Epoch 8/100
1/1 [==============================] - 0s 293ms/step - loss: 0.0489 - val_loss: 0.
0142
Epoch 9/100
1/1 [==============================] - 0s 284ms/step - loss: 0.0494 - val_loss: 0.
0122
Epoch 10/100
1/1 [==============================] - 0s 292ms/step - loss: 0.0450 - val_loss: 0.
0107
Epoch 11/100
1/1 [==============================] - 0s 299ms/step - loss: 0.0355 - val_loss: 0.
0095
Epoch 12/100
1/1 [==============================] - 0s 291ms/step - loss: 0.0238 - val_loss: 0.
0101
Epoch 13/100
1/1 [==============================] - 0s 290ms/step - loss: 0.0146 - val_loss: 0.
0139
Epoch 14/100
1/1 [==============================] - 0s 282ms/step - loss: 0.0123 - val_loss: 0.
0190
Epoch 15/100
1/1 [==============================] - 0s 299ms/step - loss: 0.0138 - val_loss: 0.
0186
Epoch 16/100
1/1 [==============================] - 0s 295ms/step - loss: 0.0103 - val_loss: 0.
0114
Epoch 17/100
1/1 [==============================] - 0s 293ms/step - loss: 0.0043 - val_loss: 0.
0038
Epoch 18/100
1/1 [==============================] - 0s 296ms/step - loss: 0.0033 - val_loss: 0.
0012
Epoch 19/100
1/1 [==============================] - 0s 282ms/step - loss: 0.0060 - val_loss: 0.
0024
Epoch 20/100
1/1 [==============================] - 0s 286ms/step - loss: 0.0073 - val_loss: 0.
0033
Epoch 21/100
```

```
1/1 [==============================] - 0s 283ms/step - loss: 0.0055 - val_loss: 0.
0027
Epoch 22/100
1/1 [==============================] - 0s 296ms/step - loss: 0.0029 - val_loss: 0.
0018
Epoch 23/100
1/1 [==============================] - 0s 289ms/step - loss: 0.0018 - val_loss: 0.
0019
Epoch 24/100
1/1 [==============================] - 0s 286ms/step - loss: 0.0028 - val_loss: 0.
0023
Epoch 25/100
1/1 [==============================] - 0s 292ms/step - loss: 0.0046 - val_loss: 0.
0023
Epoch 26/100
1/1 [==============================] - 0s 283ms/step - loss: 0.0055 - val_loss: 0.
0020
Epoch 27/100
1/1 [==============================] - 0s 283ms/step - loss: 0.0050 - val_loss: 0.
0022
Epoch 28/100
1/1 [==============================] - 0s 283ms/step - loss: 0.0041 - val_loss: 0.
0031
Epoch 29/100
1/1 [==============================] - 0s 281ms/step - loss: 0.0035 - val_loss: 0.
0040
Epoch 30/100
1/1 [==============================] - 0s 300ms/step - loss: 0.0031 - val_loss: 0.
0043
Epoch 31/100
1/1 [==============================] - 0s 294ms/step - loss: 0.0026 - val_loss: 0.
0040
Epoch 32/100
1/1 [==============================] - 0s 301ms/step - loss: 0.0021 - val_loss: 0.
0030
Epoch 33/100
1/1 [==============================] - 0s 293ms/step - loss: 0.0021 - val_loss: 0.
0020
Epoch 34/100
1/1 [==============================] - 0s 311ms/step - loss: 0.0023 - val_loss: 0.
0016
Epoch 35/100
1/1 [==============================] - 0s 282ms/step - loss: 0.0026 - val_loss: 0.
0017
Epoch 36/100
1/1 [==============================] - 0s 273ms/step - loss: 0.0024 - val_loss: 0.
0020
Epoch 37/100
1/1 [==============================] - 0s 285ms/step - loss: 0.0020 - val_loss: 0.
0021
Epoch 38/100
1/1 [==============================] - 0s 290ms/step - loss: 0.0017 - val_loss: 0.
0021
```

Out[6]:

```
<tensorflow.python.keras.callbacks.History at 0x7fcfb23ec350>
```

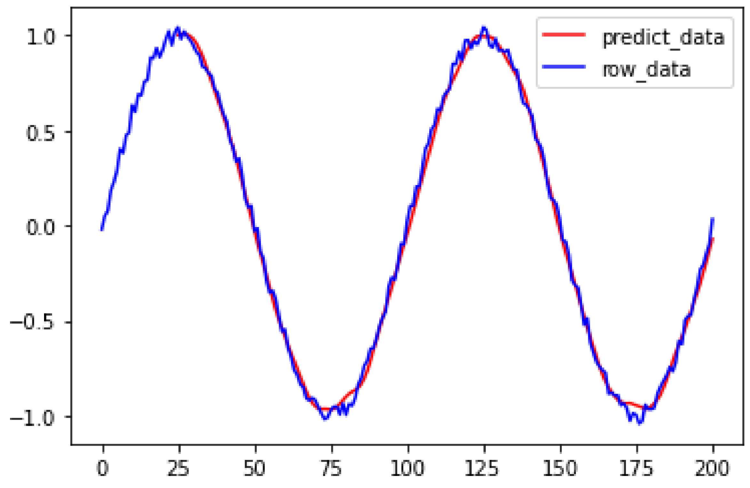
⑤予測

In [7]:

```
# 予測
predicted = model.predict(g)
```

In [8]:

```
plt.figure()
plt.plot(range(25,len(predicted)+25),predicted, color="r", label="predict_data")
plt.plot(range(0, len(f)), f, color="b", label="row_data")
plt.legend()
plt.show()
```



→再現を確認

In [ ]: