Recurrent Neural Networks

CE/CZ4042 – Tutorial 8

1. A recurrent neural network (RNN) receives sequences of 2-dimensional inputs and has three hidden neurons and one output neuron. The weight matrix $\boldsymbol{U}$ connecting the input to the hidden layer, the weight matrix $\boldsymbol{V}$ connecting the hidden output to the output layer, the hidden layer bias $\boldsymbol{b}$ and the output layer bias $\boldsymbol{c}$ are given by

$$\boldsymbol{U} = \begin{pmatrix} -1.0 & 0.5 \\ 0.5 & 0.1 \\ 0.2 & -2.0 \end{pmatrix}, \boldsymbol{V} = \begin{pmatrix} 2.0 \\ -1.5 \end{pmatrix}, \boldsymbol{b} = \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}, \text{ and } \boldsymbol{c} = 0.5.$$

Find the output sequence for an input sequence of $\left(\boldsymbol{x}(t)\right)_{t=1}^{4}$ where

$$\boldsymbol{x}(1) = \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix}, \boldsymbol{x}(2) = \begin{pmatrix} -1 \\ 1 \\ -2 \end{pmatrix}, \boldsymbol{x}(3) = \begin{pmatrix} 0 \\ 3 \\ 1 \end{pmatrix}, \text{ and } \boldsymbol{x}(4) = \begin{pmatrix} 2 \\ -1 \\ 0 \end{pmatrix} \text{if}$$

(a) The RNN is of hidden-recurrence (Elman) type with the recurrence weight matrix $\boldsymbol{W}$ connecting the previous hidden output to the current hidden layer input is given by

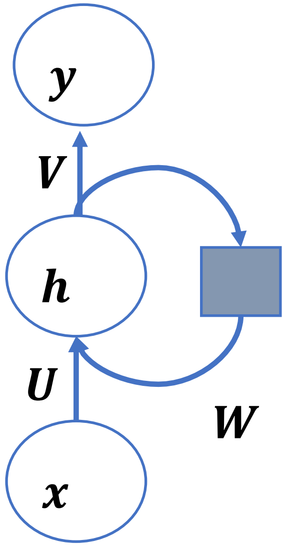
$$\boldsymbol{W} = \begin{pmatrix} 2.0 & 1.3 \\ 1.5 & 0.0 \end{pmatrix}.$$

Assume that the hidden activations are initialized to zero.

Hidden Recurrence:

$$\boldsymbol{U} = \begin{pmatrix} -1.0 & 0.5 \\ 0.5 & 0.1 \\ 0.2 & -2.0 \end{pmatrix}, \boldsymbol{V} = \begin{pmatrix} 2.0 \\ -1.5 \end{pmatrix} \text{ and } \boldsymbol{W} = \begin{pmatrix} 2.0 & 1.3 \\ 1.5 & 0.0 \end{pmatrix}.$$

$$\boldsymbol{b} = \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}, \text{ and } \boldsymbol{c} = 0.5$$



Three input neurons, two hidden neurons, and one output neuron.

$$\boldsymbol{h}(t) = \phi\left(\boldsymbol{U}^T\boldsymbol{x}(t) + \boldsymbol{W}^T\boldsymbol{h}(t-1) + \boldsymbol{b}\right)$$
$$\boldsymbol{y}(t) = \sigma\left(\boldsymbol{V}^T\boldsymbol{h}(t) + \boldsymbol{c}\right)$$

$$\phi(u) = \tanh(u) = \frac{e^u - e^{-u}}{e^u + e^{-u}}$$
$$\sigma(u) = \text{sigmoid}(u) = \frac{1}{1+e^{-u}}.$$

Assume $\boldsymbol{h}(0) = (0 \quad 0)^T$.

At t=1, $\boldsymbol{x}(1) = \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix}$:

$$\boldsymbol{h}(1) = \phi\left(\boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{x}(1) + \boldsymbol{W}^{\boldsymbol{T}}\boldsymbol{h}(0) + \boldsymbol{b}\right)$$

$$= \tanh\left(\begin{pmatrix} -1.0 & 0.5 & 0.2 \\ 0.5 & 0.1 & -2.0 \end{pmatrix}\begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix} + \begin{pmatrix} 2.0 & 1.3 \\ 1.5 & 0.0 \end{pmatrix}\begin{pmatrix} 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}\right) = \begin{pmatrix} 0.0 \\ 0.99 \end{pmatrix}$$

$$\boldsymbol{y}(1) = \sigma\left(\boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{h}(1) + \boldsymbol{c}\right) = \mathrm{sigmoid}\left(\begin{pmatrix} 2.0 & -1.5 \end{pmatrix}\begin{pmatrix} 0.0 \\ 0.99 \end{pmatrix} + (0.5)\right) = (0.27)$$

At t=2, $\boldsymbol{x}(2) = \begin{pmatrix} -1 \\ 1 \\ -2 \end{pmatrix}$:

$$\boldsymbol{h}(2) = \phi\left(\boldsymbol{U}^T\boldsymbol{x}(2) + \boldsymbol{W}^T\boldsymbol{h}(1) + \boldsymbol{b}\right)$$

$$= \tanh\left(\begin{pmatrix} -1.0 & 0.5 & 0.2 \\ 0.5 & 0.1 & -2.0 \end{pmatrix}\begin{pmatrix} -1 \\ 1 \\ -2 \end{pmatrix} + \begin{pmatrix} 2.0 & 1.3 \\ 1.5 & 0.0 \end{pmatrix}\begin{pmatrix} 0.0 \\ 0.99 \end{pmatrix} + \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}\right) = \begin{pmatrix} 0.99 \\ 1.0 \end{pmatrix}$$

$$\boldsymbol{y}(2) = \sigma\left(\boldsymbol{V}^T\boldsymbol{h}(2) + \boldsymbol{c}\right) = \text{sigmoid}\left(\begin{pmatrix} 2.0 & -1.5 \end{pmatrix}\begin{pmatrix} 0.99 \\ 1.0 \end{pmatrix} + (0.5)\right) = (0.73)$$

Similarly,

at t=3, $\boldsymbol{x}(3) = \begin{pmatrix} 0 \\ 3 \\ 1 \end{pmatrix}$; $\boldsymbol{h}(3) = \begin{pmatrix} 1.0 \\ -0.21 \end{pmatrix}$ and $\boldsymbol{y}(3) = (0.94)$

at t=4, $\boldsymbol{x}(4) = \begin{pmatrix} 2 \\ -1 \\ 0 \end{pmatrix}$; $\boldsymbol{h}(4) = \begin{pmatrix} -0.54 \\ 0.98 \end{pmatrix}$ and $\boldsymbol{y}(4) = (0.11)$

The input sequence $\left(\boldsymbol{x}(1), \boldsymbol{x}(2), \boldsymbol{x}(3), \boldsymbol{x}(4)\right)$:

$$\left(\begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix} \quad \begin{pmatrix} -1 \\ 1 \\ -2 \end{pmatrix} \quad \begin{pmatrix} 0 \\ 3 \\ 1 \end{pmatrix} \quad \begin{pmatrix} 2 \\ -1 \\ 0 \end{pmatrix}\right)$$

The output sequence $\left(\boldsymbol{y}(1), \boldsymbol{y}(2), \boldsymbol{y}(3), \boldsymbol{y}(4)\right)$:

$$((0.27) \quad (0.73) \quad (0.94) \quad (0.11))$$

1. A recurrent neural network (RNN) receives sequences of 2-dimensional inputs and has three hidden neurons and one output neuron. The weight matrix $\boldsymbol{U}$ connecting the input to the hidden layer, the weight matrix $\boldsymbol{V}$ connecting the hidden output to the output layer, the hidden layer bias $\boldsymbol{b}$ and the output layer bias $\boldsymbol{c}$ are given by

$$\boldsymbol{U} = \begin{pmatrix} -1.0 & 0.5 \\ 0.5 & 0.1 \\ 0.2 & -2.0 \end{pmatrix}, \boldsymbol{V} = \begin{pmatrix} 2.0 \\ -1.5 \end{pmatrix}, \boldsymbol{b} = \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}, \text{ and } \boldsymbol{c} = 0.5.$$

Find the output sequence for an input sequence of $(\boldsymbol{x}(t))_{t=1}^{4}$ where

$$\boldsymbol{x}(1) = \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix}, \boldsymbol{x}(2) = \begin{pmatrix} -1 \\ 1 \\ -2 \end{pmatrix}, \boldsymbol{x}(3) = \begin{pmatrix} 0 \\ 3 \\ 1 \end{pmatrix}, \text{ and } \boldsymbol{x}(4) = \begin{pmatrix} 2 \\ -1 \\ 0 \end{pmatrix} \text{if}$$

(b) The RNN is of top-down recurrence (Jordan) type with the weight matrix $\boldsymbol{W}$ connecting the previous output to the current state of hidden layer is given by

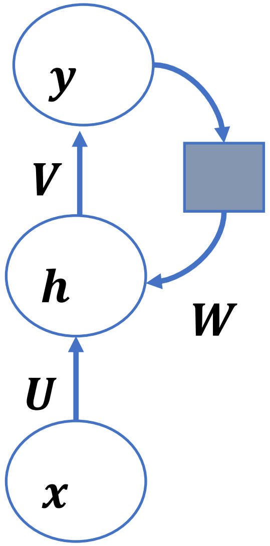
$$\boldsymbol{W} = \begin{pmatrix} 2.0 & 1.3 \end{pmatrix}.$$

Assume that the output activations are initialized to zero.

Top-down Recurrence:

$$\boldsymbol{U} = \begin{pmatrix} -1.0 & 0.5 \\ 0.5 & 0.1 \\ 0.2 & -2.0 \end{pmatrix}, \boldsymbol{V} = \begin{pmatrix} 2.0 \\ -1.5 \end{pmatrix}, \boldsymbol{W} = \begin{pmatrix} 2.0 & 1.3 \end{pmatrix}$$

$$\boldsymbol{b} = \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}, \text{ and } \boldsymbol{c} = 0.5$$



Three input neurons, two hidden neurons, and one output neuron.

$$\boldsymbol{h}(t) = \phi\left(\boldsymbol{U}^T\boldsymbol{x}(t) + \boldsymbol{W}^T\boldsymbol{y}(t-1) + \boldsymbol{b}\right)$$
$$\boldsymbol{y}(t) = \sigma\left(\boldsymbol{V}^T\boldsymbol{h}(t) + \boldsymbol{c}\right)$$

Initially,

$$\boldsymbol{Y}(0) = (0.0)$$

At t=1, $\boldsymbol{x}(1) = \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix}$:

$$\boldsymbol{h}(1) = \phi\left(\boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{x}(1) + \boldsymbol{W}^{\boldsymbol{T}}y(0) + \boldsymbol{b}\right)$$

$$= \tanh\left(\begin{pmatrix} -1.0 & 0.5 & 0.2 \\ 0.5 & 0.1 & -2.0 \end{pmatrix}\begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix} + \begin{pmatrix} 2.0 \\ 1.3 \end{pmatrix}(0) + \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}\right) = \begin{pmatrix} 0.0 \\ 0.99 \end{pmatrix}$$

$$\boldsymbol{y}(1) = \sigma\left(\boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{h}(1) + \boldsymbol{c}\right) = \text{sigmoid}\left(\begin{pmatrix} 2.0 & -1.5 & 0.2 \end{pmatrix}\begin{pmatrix} 0.0 \\ 0.99 \end{pmatrix} + (0.5)\right) = (0.27)$$

At t=2, $\boldsymbol{x}(2) = \begin{pmatrix} -1 \\ 1 \\ -2 \end{pmatrix}$:

$$\boldsymbol{h}(2) = \phi\left(\boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{x}(2) + \boldsymbol{W}^{\boldsymbol{T}}y(1) + \boldsymbol{b}\right)$$

$$= \tanh\left(\begin{pmatrix} -1.0 & 0.5 & 0.2 \\ 0.5 & 0.1 & -2.0 \end{pmatrix}\begin{pmatrix} -1 \\ 1 \\ -2 \end{pmatrix} + \begin{pmatrix} 2.0 \\ 1.3 \end{pmatrix}(0.27) + \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}\right) = \begin{pmatrix} 0.95 \\ 1.0 \end{pmatrix}$$

$$\boldsymbol{y}(2) = \sigma\left(\boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{h}(2) + \boldsymbol{c}\right) = \mathrm{sigmoid}\left(\begin{pmatrix} 2.0 & -1.5 & 0.2 \end{pmatrix}\begin{pmatrix} 0.95 \\ 1.0 \end{pmatrix} + (0.5)\right) = (0.71)$$

Similarly,

at t=3, $\boldsymbol{x}(3) = \begin{pmatrix} 0 \\ 3 \\ 1 \end{pmatrix}$; $\boldsymbol{h}(3) = \begin{pmatrix} 1.0 \\ -0.52 \end{pmatrix}$ and $\boldsymbol{y}(3) = (0.96)$

at t=4, $\boldsymbol{x}(4) = \begin{pmatrix} 2 \\ -1 \\ 0 \end{pmatrix}$; $\boldsymbol{h}(4) = \begin{pmatrix} -0.36 \\ 0.98 \end{pmatrix}$ and $\boldsymbol{y}(4) = (0.16)$

The input sequence $(\boldsymbol{x}(1), \boldsymbol{x}(2), \boldsymbol{x}(3), \boldsymbol{x}(4))$:

$$\left(\begin{pmatrix}1\\2\\-1\end{pmatrix} \begin{pmatrix}-1\\1\\-2\end{pmatrix} \begin{pmatrix}0\\3\\1\end{pmatrix} \begin{pmatrix}2\\-1\\0\end{pmatrix}\right)$$

The output sequence $(\boldsymbol{y}(1), \boldsymbol{y}(2), \boldsymbol{y}(3), \boldsymbol{y}(4))$:

$$((0.27) \quad (0.71) \quad (0.96) \quad (0.16))$$

2. An RNN receives 3-dimensional input sequence and produces 2-dimensional output sequence. It has 5 hidden neurons and receives sequences of 100 time steps.

- Generate 8 training input sequences by drawing values uniformly between 0 and 1.0.
- If the sequence $\left(\boldsymbol{y}(t)\right)_{t=1}^{100}$ where $\boldsymbol{y}(t) = \left(y_k(t)\right)_{k=1}^{2} \in \boldsymbol{R}^2$ denotes the output sequence for an input sequence $\left(\boldsymbol{x}(t)\right)_{t=1}^{100}$ where $\boldsymbol{x}(t) = \left(x_i(t)\right)_{i=1}^{3} \in \boldsymbol{R}^3$, generate the corresponding output sequences for the input training sequences as follows:

$$y_1(t) = 5x_1(t) - 0.2x_3(t-1) + 0.1\varepsilon$$
$$y_2(t) = 25x_2(t-1)x_3(t-3) + 0.1\varepsilon$$

where ε is standard normally distributed random variable.

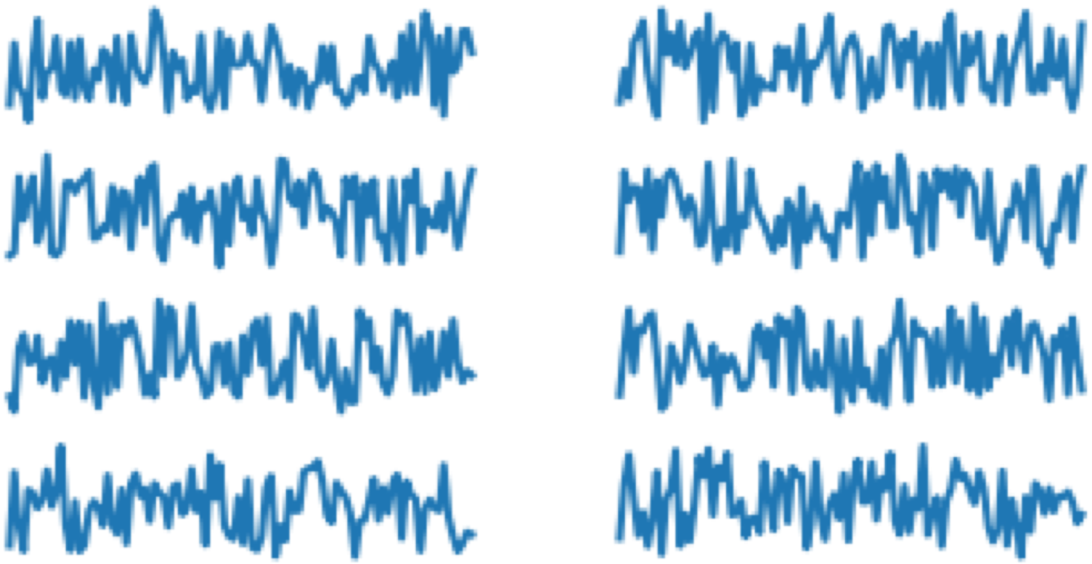
Train an RNN to learn the above sequences by using gradient descent learning. Use a learning factor $\alpha = 0.01$ and Adam optimizer.

```
n_in = 3
n_hidden = 5
n_out = 2

n_steps = 100
n_seqs = 8
```

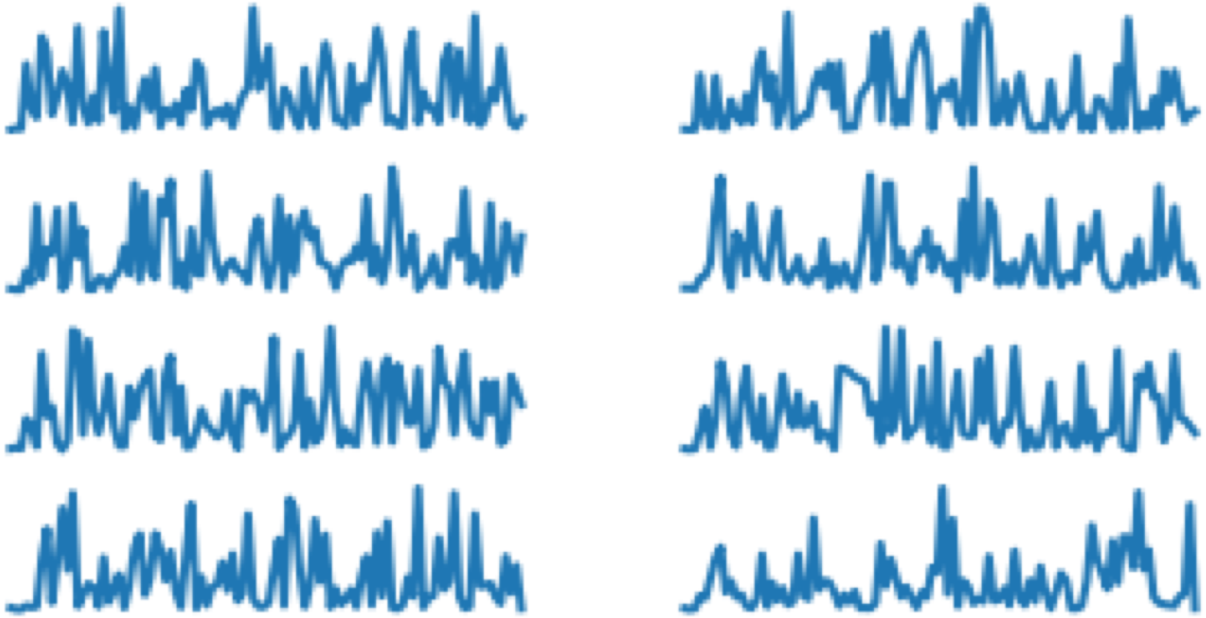
$$y_1(t) = 5x_1(t) - 0.2x_3(t-1) + 0.1\varepsilon$$
$$y_2(t) = 25x_2(t-1)x_3(t-3) + 0.1\varepsilon$$

```
x_train = np.random.rand(n_seqs, n_steps, n_in)
y_train = np.zeros([n_seqs, n_steps, n_out])

y_train[:,1:,0] = 5*x_train[:,1:,0] - x_train[:,:-1,2]
y_train[:,3:,1] = 25*x_train[:,2:-1, 1]*x_train[:,:-3,2]

y_train += 0.1*np.random.randn(n_seqs, n_steps, n_out)
```

$$y_1(t) = 5x_1(t) - 0.2x_3(t-1) + 0.1\varepsilon$$

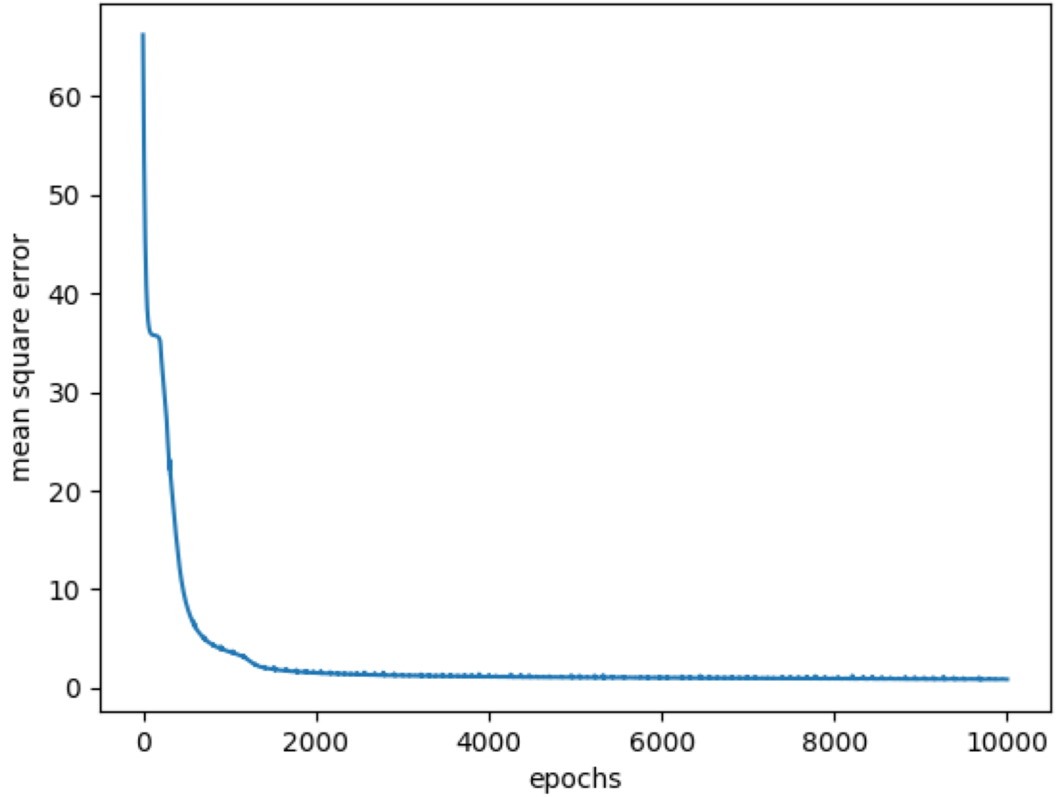


$$y_2(t) = 25x_2(t-1)x_3(t-3) + 0.1\varepsilon$$



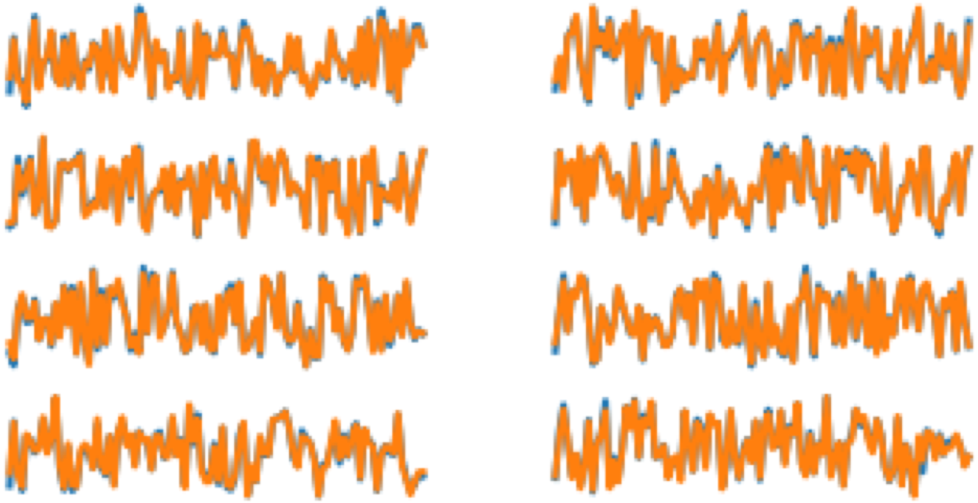
$$\boldsymbol{H}(t) = \phi(\boldsymbol{X}(t)\boldsymbol{U} + \boldsymbol{Y}(t-1)\boldsymbol{W} + \boldsymbol{B})$$
$$\boldsymbol{Y}(t) = \sigma(\boldsymbol{H}(t)\boldsymbol{V} + \boldsymbol{C})$$

```
h = init_state
ys = []
for i, x_ in enumerate(tf.split(x, n_steps, axis = 1)):
    h = tf.tanh(tf.matmul(tf.squeeze(x_), U) + tf.matmul(h, W) + b)
    y_ = tf.matmul(h, V) + c
    ys.append(y_)

ys_ = tf.stack(ys, axis=1)
cost = tf.reduce_mean(tf.reduce_sum(tf.square(y - ys_), axis=2))
train_op = tf.train.AdamOptimizer(lr).minimize(loss=cost)
```



mean square error
60
50
40
30
20
10
0
0
2000
4000
6000
8000
10000
epochs

Predicted $y_1(t)$



Predicted $y_2(t)$