HW6_abuitano

Problem 1

a) False: The activation functions involved won't let $h_t = h_{t-1}$. O_t has sigmoid activation while tanh(ct) has tanh activation. f_t, i_t, $\tilde{c}_t$ also have their own non linearities which will affect the value of h_t.

b) False: $C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$, so even if $f_t \approx 0$, error should still get backpropagated through o_t, i_t and $\tilde{c}_t$.

c) True, f_t, i_t and O_t have sigmoid activation, so they'll only have values between 0 and 1.

d) False. with sigmoid activation on f_t, O_t, i_t, all their entries will be non-negative but won't necesarily sum to 1. each entry has sigmoid applied independently.

e) f_t, i_t, O_t all should be the same dimension as h_t

f) $\boxed{h_1 = 0.21741}$ $\boxed{h_2 = -0.18988}$ * Code attached for computation

g) $MSE = \frac{1}{2}\left[(y_1 - h_1)^2 + (y_2 - h_2)^2\right] = \frac{1}{2}\left[0.07985 + 0.97986\right) = \boxed{0.5298}$

In [17]:
```
import numpy as np

wf = [1, 2]
wi = [-1, 0]
wc = [1, 2]
wo = [3, 0]

uf = [0.5]
ui = [2]
uc = [1.5]
uo = [-1]

bf = [0.2]
bi = [-0.1]
bc = [0.5]
bo = [0.8]

x1 = [[1],[0]]
x2 = [[0.5], [-1]]

y1 = 0.5
y2 = 0.8

h0 = 0
c0 = 0
```

In [18]:
```
def sigmoid(x):
    return 1/(1+np.exp(-x))
def tanh(x):
    return (np.exp(x)-np.exp(-x))/(np.exp(x)+np.exp(-x))
```

In [19]:
```
def foi(w, u, b, x, h):
    return sigmoid(np.dot(w,x) + np.dot(u,h) + b)
def c_hash(w, u, b, x, h):
    return tanh(np.dot(w,x) + np.dot(u,h) + b)

f1 = foi(wf, uf, bf, x1, h0)
i1 = foi(wi, ui, bi, x1, h0)
o1 = foi(wo, uo, bo, x1, h0)
chash1 = c_hash(wc, uc, bc, x1, h0)

c1 = f1*c0 + i1*chash1

h1 = o1*tanh(c1)
```

In [20]:
```
h1
```

Out[20]:
```
array([0.21741464])
```

In [23]:
```
f2 = foi(wf, uf, bf, x2, h1)
i2 = foi(wi, ui, bi, x2, h1)
o2 = foi(wo, uo, bo, x2, h1)
chash2 = c_hash(wc, uc, bc, x2, h1)

c2 = f2*c1 + i2*chash2

h2 = o2*tanh(c2)
```

In [24]:
```
h2
```
Out[24]:
```
array([-0.18988225])
```
In [25]:
```
f2
```
Out[25]:
```
array([0.23302782])
```
In [26]:
```
i2
```
Out[26]:
```
array([0.45880094])
```
In [27]:
```
o2
```
Out[27]:
```
array([0.88919901])
```
In [28]:
```
chash2
```
Out[28]:
```
array([-0.58752507])
```

In [1]:

```
import torch
import torch.nn as nn
from torch.autograd import Variable

import os
import numpy as np
from torch.optim import Adam
from torch.utils.data import DataLoader

from dataset import FlowDataset
```

In [2]:

```python
class FlowLSTM(nn.Module):
    def __init__(self, input_size, hidden_size, num_layers, dropout):
        super(FlowLSTM, self).__init__()
        # build your model here
        # your input should be of dim (batch_size, seq_len, input_size)
        # your output should be of dim (batch_size, seq_len, input_size) as well
        # since you are predicting velocity of next step given previous one

        # feel free to add functions in the class if needed

        self.input_size = input_size
        self.hidden_size = hidden_size
        self.num_layers = num_layers
        self.dropout = dropout

        if self.num_layers > 1:
            self.dropout = dropout
        else:
            self.dropout = 0.0

        # define the LSTM layer
        self.lstm = nn.LSTM(
            input_size = self.input_size,
            hidden_size = self.hidden_size,
            num_layers = self.num_layers,
            batch_first=True,
            dropout=self.dropout
        )

        # define the output layer
        self.dense = nn.Linear(self.hidden_size,self.input_size)

    # initialize hidden state as
    def initial_hidden_state(self,batch):
        return Variable(torch.zeros(self.num_layers,batch,self.hidden_size))


    # forward pass through LSTM layer
    def forward(self, x):
        '''
        input: x of dim (batch_size, 19, 17)
        '''
        # define your feedforward pass
        batch = x.shape[0]
        h_0 = self.initial_hidden_state(batch)
        h_1 = self.initial_hidden_state(batch)
        out, _ = self.lstm(x, (h_0, h_1))
        out = self.dense(out)
        return out


    # forward pass through LSTM layer for testing
    def test(self, x):
        '''
        input: x of dim (batch_size, 17)
        '''
        # define your feedforward pass
        pred = torch.empty(x.shape[0], 19, x.shape[1])
        pred[:,0,:] = x

        for i in range(1, pred.shape[1]):
            pred[:,i,:] = self.forward(pred[:,:i,:])[:,-1,:]
```

```
            pred[:,i,:] = self.forward(pred[:,:i,:])[:,-1,:]

        return pred
```

In [3]:
```
# from lstm import FlowLSTM

def main():
    # check if cuda available
    device = 'cuda:0' if torch.cuda.is_available() else 'cpu'

    # define dataset and dataloader
    train_dataset = FlowDataset(mode='train')
    test_dataset = FlowDataset(mode='test')
    train_loader = DataLoader(dataset=train_dataset, batch_size=32, shuffle=True, num_wo
    test_loader = DataLoader(dataset=test_dataset, batch_size=16, shuffle=False, num_wor

    # hyper-parameters
    num_epochs = 45
    lr = 0.0008
    input_size = 17 # do not change input size
    hidden_size = 128
    num_layers = 6
    dropout = 0.2

    model = FlowLSTM(
        input_size=input_size,
        hidden_size=hidden_size,
        num_layers=num_layers,
        dropout=dropout
    ).to(device)

    # define your LSTM loss function here
    loss_func = nn.MSELoss()

    # define optimizer for lstm model
    optim = Adam(model.parameters(), lr=lr)
    loss_list = []
    for epoch in range(num_epochs):
        for n_batch, (in_batch, label) in enumerate(train_loader):
            in_batch, label = in_batch.to(device), label.to(device)

            # train LSTM
            out = model(in_batch)
            # calculate LSTM loss
            loss = loss_func(out, label)

            optim.zero_grad()
            loss.backward()
            optim.step()

            # print loss while training
            loss_list.append(loss.item())
            if (n_batch + 1) % 200 == 0:
                print("Epoch: [{}/{}], Batch: {}, Loss: {}".format(
                    epoch+1, num_epochs, n_batch+1, loss.item()))

    # test trained LSTM model
    l1_err, l2_err = 0, 0
    l1_loss = nn.L1Loss()
    l2_loss = nn.MSELoss()


    model.eval()
```

```
    model.eval()
    with torch.no_grad():
        for n_batch, (in_batch, label) in enumerate(test_loader):
            in_batch, label = in_batch.to(device), label.to(device)
            pred = model.test(in_batch)
            l1_err += l1_loss(pred, label).item()
            l2_err += l2_loss(pred, label).item()

    print("Test L1 error:", l1_err)
    print("Test L2 error:", l2_err)

    # visualize the prediction comparing to the ground truth
    if device == 'cpu':
        pred = pred.detach().numpy()[0,:,:]
        label = label.detach().numpy()[0,:,:]
    else:
        pred = pred.detach().cpu().numpy()[0,:,:]
        label = label.detach().cpu().numpy()[0,:,:]

    r = []
    num_points = 17
    interval = 1./num_points
    x = int(num_points/2)
    for j in range(-x,x+1):
        r.append(interval*j)

    from matplotlib import pyplot as plt
    plt.figure()
    for i in range(1, len(pred)):
        c = (i/(num_points+1), 1-i/(num_points+1), 0.5)
        plt.plot(pred[i], r, label='t = %s' %(i), c=c)
    plt.xlabel('velocity [m/s]')
    plt.ylabel('r [m]')
    plt.legend(bbox_to_anchor=(1,1),fontsize='x-small')
    plt.show()

    plt.figure()
    for i in range(1, len(label)):
        c = (i/(num_points+1), 1-i/(num_points+1), 0.5)
        plt.plot(label[i], r, label='t = %s' %(i), c=c)
    plt.xlabel('velocity [m/s]')
    plt.ylabel('r [m]')
    plt.legend(bbox_to_anchor=(1,1),fontsize='x-small')
    plt.show()

    plt.plot(loss_list)
    plt.title("Loss vs iterations")
    plt.xlabel("Iteration")
```

In [4]:

```
if __name__ == "__main__":
    main()
```

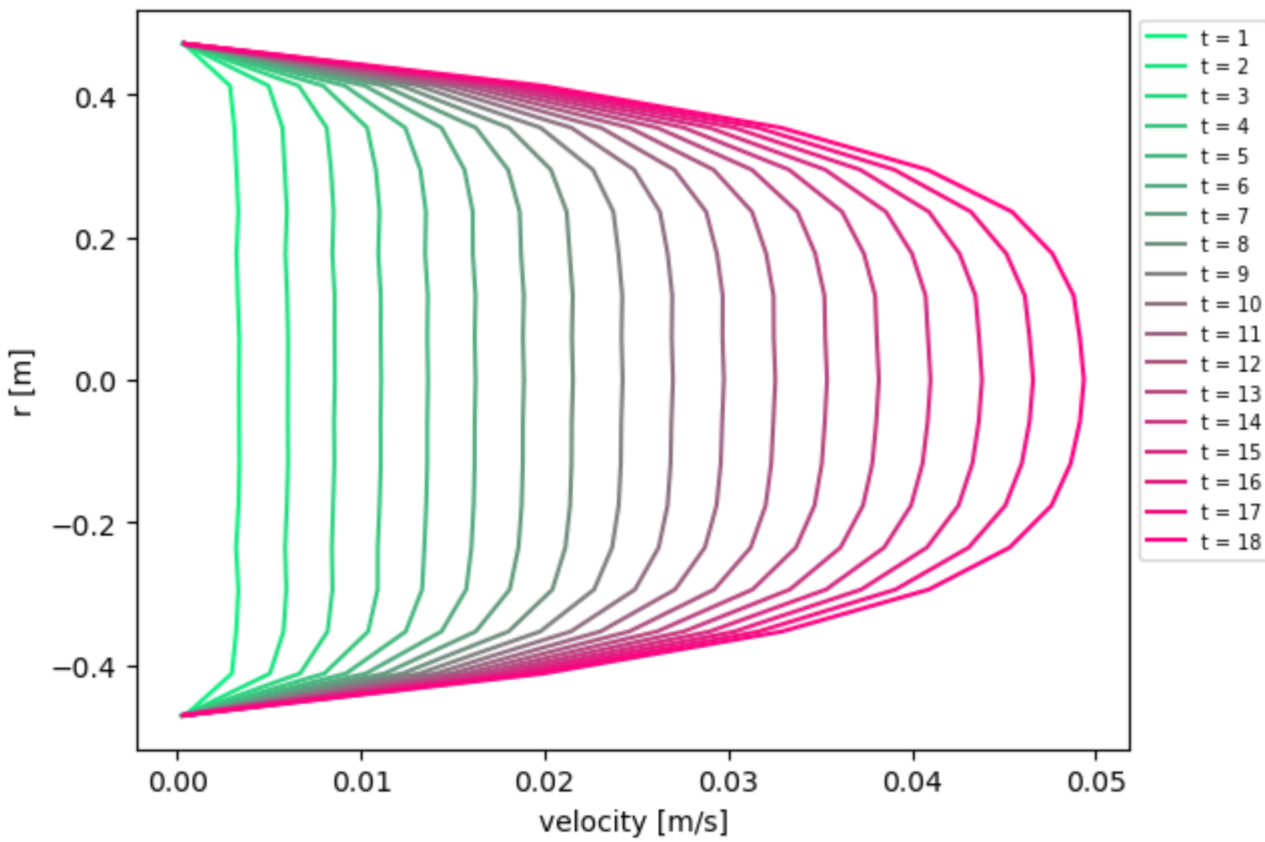
```
Epoch: [1/45], Batch: 200, Loss: 2.3255193809745833e-05
Epoch: [1/45], Batch: 400, Loss: 6.974390089453664e-06
Epoch: [1/45], Batch: 600, Loss: 4.437452389538521e-06
Epoch: [1/45], Batch: 800, Loss: 3.559965762178763e-06
Epoch: [1/45], Batch: 1000, Loss: 3.371770390003803e-06
Epoch: [1/45], Batch: 1200, Loss: 3.0058367883611936e-06
Epoch: [2/45], Batch: 200, Loss: 1.5161417650233489e-06
Epoch: [2/45], Batch: 400, Loss: 2.0050592866027728e-06
Epoch: [2/45], Batch: 600, Loss: 1.497989956078527e-06
Epoch: [2/45], Batch: 800, Loss: 1.392068497807486e-06
Epoch: [2/45], Batch: 1000, Loss: 1.2230227639520308e-06
Epoch: [2/45], Batch: 1200, Loss: 6.150688705020002e-07
Epoch: [3/45], Batch: 200, Loss: 1.0360344049331616e-06
Epoch: [3/45], Batch: 400, Loss: 8.67429491790972e-07
Epoch: [3/45], Batch: 600, Loss: 5.785137204838975e-07
Epoch: [3/45], Batch: 800, Loss: 6.280916409195925e-07
Epoch: [3/45], Batch: 1000, Loss: 6.038130209162773e-07
Epoch: [3/45], Batch: 1200, Loss: 4.367250028280978e-07
Epoch: [4/45], Batch: 200, Loss: 6.784260335734871e-07
Epoch: [4/45], Batch: 400, Loss: 5.26072028605995e-07
Epoch: [4/45], Batch: 600, Loss: 5.45587340639031e-07
Epoch: [4/45], Batch: 800, Loss: 5.432155489870638e-07
Epoch: [4/45], Batch: 1000, Loss: 7.672447281947825e-07
Epoch: [4/45], Batch: 1200, Loss: 5.353983283384878e-07
Epoch: [5/45], Batch: 200, Loss: 5.540914003177022e-07
Epoch: [5/45], Batch: 400, Loss: 3.655332534435729e-07
Epoch: [5/45], Batch: 600, Loss: 5.009224537388945e-07
Epoch: [5/45], Batch: 800, Loss: 2.6971557076649333e-07
Epoch: [5/45], Batch: 1000, Loss: 4.3044002495662426e-07
Epoch: [5/45], Batch: 1200, Loss: 3.1505757647209975e-07
Epoch: [6/45], Batch: 200, Loss: 2.685513322830957e-07
Epoch: [6/45], Batch: 400, Loss: 9.080824270313315e-07
Epoch: [6/45], Batch: 600, Loss: 5.891985779271636e-07
Epoch: [6/45], Batch: 800, Loss: 3.124032446066849e-07
Epoch: [6/45], Batch: 1000, Loss: 2.7858257567459077e-07
Epoch: [6/45], Batch: 1200, Loss: 2.7688523118740704e-07
Epoch: [7/45], Batch: 200, Loss: 2.988740561704617e-07
Epoch: [7/45], Batch: 400, Loss: 2.4208856075347285e-07
Epoch: [7/45], Batch: 600, Loss: 1.8708439597503457e-07
Epoch: [7/45], Batch: 800, Loss: 2.3096536949651636e-07
Epoch: [7/45], Batch: 1000, Loss: 4.3969006924271525e-07
Epoch: [7/45], Batch: 1200, Loss: 2.169093562542912e-07
Epoch: [8/45], Batch: 200, Loss: 3.6086257182432746e-07
Epoch: [8/45], Batch: 400, Loss: 2.2154675605179364e-07
Epoch: [8/45], Batch: 600, Loss: 2.1557430329721683e-07
Epoch: [8/45], Batch: 800, Loss: 1.8375159527295182e-07
Epoch: [8/45], Batch: 1000, Loss: 4.9707205107552e-07
Epoch: [8/45], Batch: 1200, Loss: 1.4699982386900956e-07
Epoch: [9/45], Batch: 200, Loss: 1.5794260832535656e-07
Epoch: [9/45], Batch: 400, Loss: 2.983287004099111e-07
Epoch: [9/45], Batch: 600, Loss: 2.0543218681723374e-07
Epoch: [9/45], Batch: 800, Loss: 4.0722088101574627e-07
Epoch: [9/45], Batch: 1000, Loss: 1.5827691868253169e-07
Epoch: [9/45], Batch: 1200, Loss: 1.7393695372902584e-07
Epoch: [10/45], Batch: 200, Loss: 2.1991566256929218e-07
Epoch: [10/45], Batch: 400, Loss: 1.2490805545439798e-07
Epoch: [10/45], Batch: 600, Loss: 2.38253591078319e-07
Epoch: [10/45], Batch: 800, Loss: 2.2186419812442182e-07
Epoch: [10/45], Batch: 1000, Loss: 3.005629594099446e-07
Epoch: [10/45], Batch: 1200, Loss: 1.2097183343939832e-07
Epoch: [11/45], Batch: 200, Loss: 1.9989639099549095e-07
Epoch: [11/45], Batch: 400, Loss: 2.196893689188073e-07
```

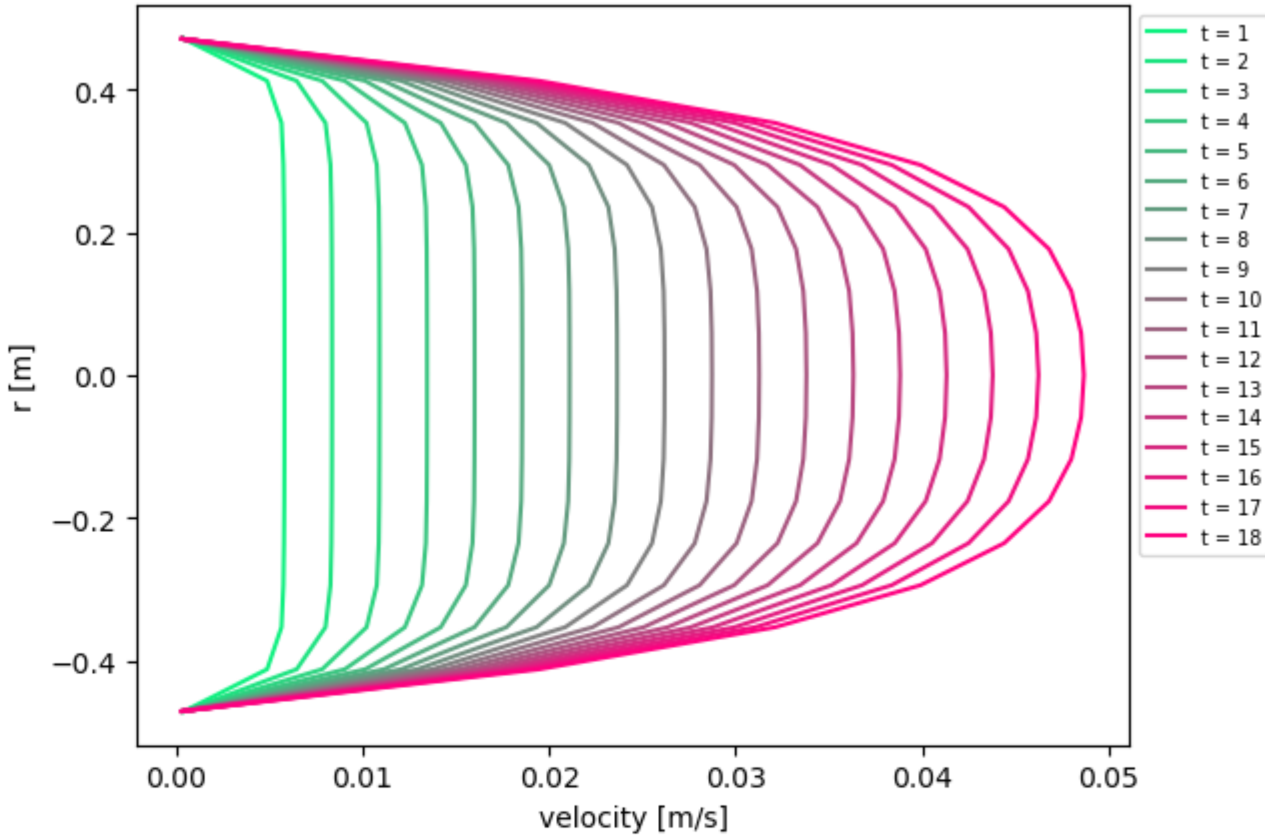
```
Epoch: [11/45], Batch: 600, Loss: 1.1763285101551446e-07
Epoch: [11/45], Batch: 800, Loss: 1.8629104658884899e-07
Epoch: [11/45], Batch: 1000, Loss: 3.171986691086204e-07
Epoch: [11/45], Batch: 1200, Loss: 1.6437925864920544e-07
Epoch: [12/45], Batch: 200, Loss: 3.8816671121821855e-07
Epoch: [12/45], Batch: 400, Loss: 1.2018364259347436e-07
Epoch: [12/45], Batch: 600, Loss: 1.6397129343204142e-07
Epoch: [12/45], Batch: 800, Loss: 1.277092991358586e-07
Epoch: [12/45], Batch: 1000, Loss: 1.9129956285723893e-07
Epoch: [12/45], Batch: 1200, Loss: 1.698319209708643e-07
Epoch: [13/45], Batch: 200, Loss: 1.2886945910395298e-07
Epoch: [13/45], Batch: 400, Loss: 2.1667142391379457e-07
Epoch: [13/45], Batch: 600, Loss: 1.0816107476330217e-07
Epoch: [13/45], Batch: 800, Loss: 2.2923590847767628e-07
Epoch: [13/45], Batch: 1000, Loss: 2.5264790792789427e-07
Epoch: [13/45], Batch: 1200, Loss: 4.1462149624749145e-07
Epoch: [14/45], Batch: 200, Loss: 9.973774695026805e-08
Epoch: [14/45], Batch: 400, Loss: 2.1136044381364627e-07
Epoch: [14/45], Batch: 600, Loss: 1.2987788977625314e-07
Epoch: [14/45], Batch: 800, Loss: 2.352946495420838e-07
Epoch: [14/45], Batch: 1000, Loss: 2.8547083275043406e-07
Epoch: [14/45], Batch: 1200, Loss: 1.7971699151075882e-07
Epoch: [15/45], Batch: 200, Loss: 2.1129685023879574e-07
Epoch: [15/45], Batch: 400, Loss: 1.0344368917003521e-07
Epoch: [15/45], Batch: 600, Loss: 8.938466322661043e-08
Epoch: [15/45], Batch: 800, Loss: 1.571856529380966e-07
Epoch: [15/45], Batch: 1000, Loss: 8.961931285966784e-08
Epoch: [15/45], Batch: 1200, Loss: 1.699431635415749e-07
Epoch: [16/45], Batch: 200, Loss: 2.743237530467013e-07
Epoch: [16/45], Batch: 400, Loss: 1.4707340767472488e-07
Epoch: [16/45], Batch: 600, Loss: 1.1054943627186731e-07
Epoch: [16/45], Batch: 800, Loss: 1.6640282751723134e-07
Epoch: [16/45], Batch: 1000, Loss: 1.4717025464960898e-07
Epoch: [16/45], Batch: 1200, Loss: 1.136075908902967e-07
Epoch: [17/45], Batch: 200, Loss: 1.1405214195292501e-07
Epoch: [17/45], Batch: 400, Loss: 1.7609747260394215e-07
Epoch: [17/45], Batch: 600, Loss: 1.6729809715343436e-07
Epoch: [17/45], Batch: 800, Loss: 9.296714864603928e-08
Epoch: [17/45], Batch: 1000, Loss: 1.7882922520584543e-07
Epoch: [17/45], Batch: 1200, Loss: 3.1136977440837654e-07
Epoch: [18/45], Batch: 200, Loss: 2.618094754325284e-07
Epoch: [18/45], Batch: 400, Loss: 1.2229484980252892e-07
Epoch: [18/45], Batch: 600, Loss: 1.2552956718536734e-07
Epoch: [18/45], Batch: 800, Loss: 2.0194508465465333e-07
Epoch: [18/45], Batch: 1000, Loss: 1.1288961587752055e-07
Epoch: [18/45], Batch: 1200, Loss: 2.7033433980250265e-07
Epoch: [19/45], Batch: 200, Loss: 2.0245109055849753e-07
Epoch: [19/45], Batch: 400, Loss: 1.3586755187589006e-07
Epoch: [19/45], Batch: 600, Loss: 1.1868149840665865e-07
Epoch: [19/45], Batch: 800, Loss: 1.3127942111168522e-07
Epoch: [19/45], Batch: 1000, Loss: 1.3173341528727178e-07
Epoch: [19/45], Batch: 1200, Loss: 1.239546634224098e-07
Epoch: [20/45], Batch: 200, Loss: 1.5700584299338516e-07
Epoch: [20/45], Batch: 400, Loss: 1.1659268039920789e-07
Epoch: [20/45], Batch: 600, Loss: 1.5166634170782345e-07
Epoch: [20/45], Batch: 800, Loss: 1.9209269908060378e-07
Epoch: [20/45], Batch: 1000, Loss: 7.642901067583807e-08
Epoch: [20/45], Batch: 1200, Loss: 7.131343693345116e-08
Epoch: [21/45], Batch: 200, Loss: 2.3295973505810252e-07
Epoch: [21/45], Batch: 400, Loss: 9.31404784410006e-08
Epoch: [21/45], Batch: 600, Loss: 6.86898218305032e-08
Epoch: [21/45], Batch: 800, Loss: 9.058103245251914e-08
```

```
Epoch: [21/45], Batch: 1000, Loss: 1.9313732479986356e-07
Epoch: [21/45], Batch: 1200, Loss: 1.986401514386671e-07
Epoch: [22/45], Batch: 200, Loss: 1.0651014292761829e-07
Epoch: [22/45], Batch: 400, Loss: 2.457639709518844e-07
Epoch: [22/45], Batch: 600, Loss: 1.0657097249122671e-07
Epoch: [22/45], Batch: 800, Loss: 3.1889859997136227e-07
Epoch: [22/45], Batch: 1000, Loss: 1.3653108510425227e-07
Epoch: [22/45], Batch: 1200, Loss: 2.365805755744077e-07
Epoch: [23/45], Batch: 200, Loss: 2.397551384092367e-07
Epoch: [23/45], Batch: 400, Loss: 1.7381493933044112e-07
Epoch: [23/45], Batch: 600, Loss: 8.686259178602995e-08
Epoch: [23/45], Batch: 800, Loss: 1.216220510968924e-07
Epoch: [23/45], Batch: 1000, Loss: 2.0859334881606628e-07
Epoch: [23/45], Batch: 1200, Loss: 9.447047943922371e-08
Epoch: [24/45], Batch: 200, Loss: 1.5537776221208333e-07
Epoch: [24/45], Batch: 400, Loss: 1.74225007754103e-07
Epoch: [24/45], Batch: 600, Loss: 1.2176295172139362e-07
Epoch: [24/45], Batch: 800, Loss: 1.1742218930521631e-07
Epoch: [24/45], Batch: 1000, Loss: 7.827311776509305e-08
Epoch: [24/45], Batch: 1200, Loss: 1.1937953559026937e-07
Epoch: [25/45], Batch: 200, Loss: 9.83620012107167e-08
Epoch: [25/45], Batch: 400, Loss: 1.743040058954648e-07
Epoch: [25/45], Batch: 600, Loss: 6.091658377727072e-08
Epoch: [25/45], Batch: 800, Loss: 7.442552174552475e-08
Epoch: [25/45], Batch: 1000, Loss: 1.3898231543407746e-07
Epoch: [25/45], Batch: 1200, Loss: 7.951707914344297e-08
Epoch: [26/45], Batch: 200, Loss: 1.0837398178864532e-07
Epoch: [26/45], Batch: 400, Loss: 1.2328959542173834e-07
Epoch: [26/45], Batch: 600, Loss: 1.0887515600188635e-07
Epoch: [26/45], Batch: 800, Loss: 2.178762343874041e-07
Epoch: [26/45], Batch: 1000, Loss: 2.213676992823821e-07
Epoch: [26/45], Batch: 1200, Loss: 6.143208253206467e-08
Epoch: [27/45], Batch: 200, Loss: 1.0260980332077452e-07
Epoch: [27/45], Batch: 400, Loss: 1.7373409377796634e-07
Epoch: [27/45], Batch: 600, Loss: 4.228017189689126e-07
Epoch: [27/45], Batch: 800, Loss: 7.109301236596366e-08
Epoch: [27/45], Batch: 1000, Loss: 1.9014181873444613e-07
Epoch: [27/45], Batch: 1200, Loss: 1.954554704752809e-07
Epoch: [28/45], Batch: 200, Loss: 2.9972412107781565e-07
Epoch: [28/45], Batch: 400, Loss: 1.5562707744720683e-07
Epoch: [28/45], Batch: 600, Loss: 6.7767068401281e-08
Epoch: [28/45], Batch: 800, Loss: 2.531582481424266e-07
Epoch: [28/45], Batch: 1000, Loss: 7.649239819329523e-08
Epoch: [28/45], Batch: 1200, Loss: 1.0013773987793684e-07
Epoch: [29/45], Batch: 200, Loss: 1.1414248746177691e-07
Epoch: [29/45], Batch: 400, Loss: 7.864448292593806e-08
Epoch: [29/45], Batch: 600, Loss: 6.471776003991181e-08
Epoch: [29/45], Batch: 800, Loss: 7.43382173595819e-08
Epoch: [29/45], Batch: 1000, Loss: 8.287111086247023e-08
Epoch: [29/45], Batch: 1200, Loss: 1.0964917862565926e-07
Epoch: [30/45], Batch: 200, Loss: 1.1640654662414818e-07
Epoch: [30/45], Batch: 400, Loss: 5.7146181120515394e-08
Epoch: [30/45], Batch: 600, Loss: 8.258064809751886e-08
Epoch: [30/45], Batch: 800, Loss: 5.0429719067324186e-08
Epoch: [30/45], Batch: 1000, Loss: 8.668826723123857e-08
Epoch: [30/45], Batch: 1200, Loss: 1.569896284081551e-07
Epoch: [31/45], Batch: 200, Loss: 7.879584984493704e-08
Epoch: [31/45], Batch: 400, Loss: 1.3536100595956668e-07
Epoch: [31/45], Batch: 600, Loss: 1.0594663990559638e-07
Epoch: [31/45], Batch: 800, Loss: 1.0273726758214252e-07
Epoch: [31/45], Batch: 1000, Loss: 6.915196593126893e-08
Epoch: [31/45], Batch: 1200, Loss: 5.4292780049536304e-08
```

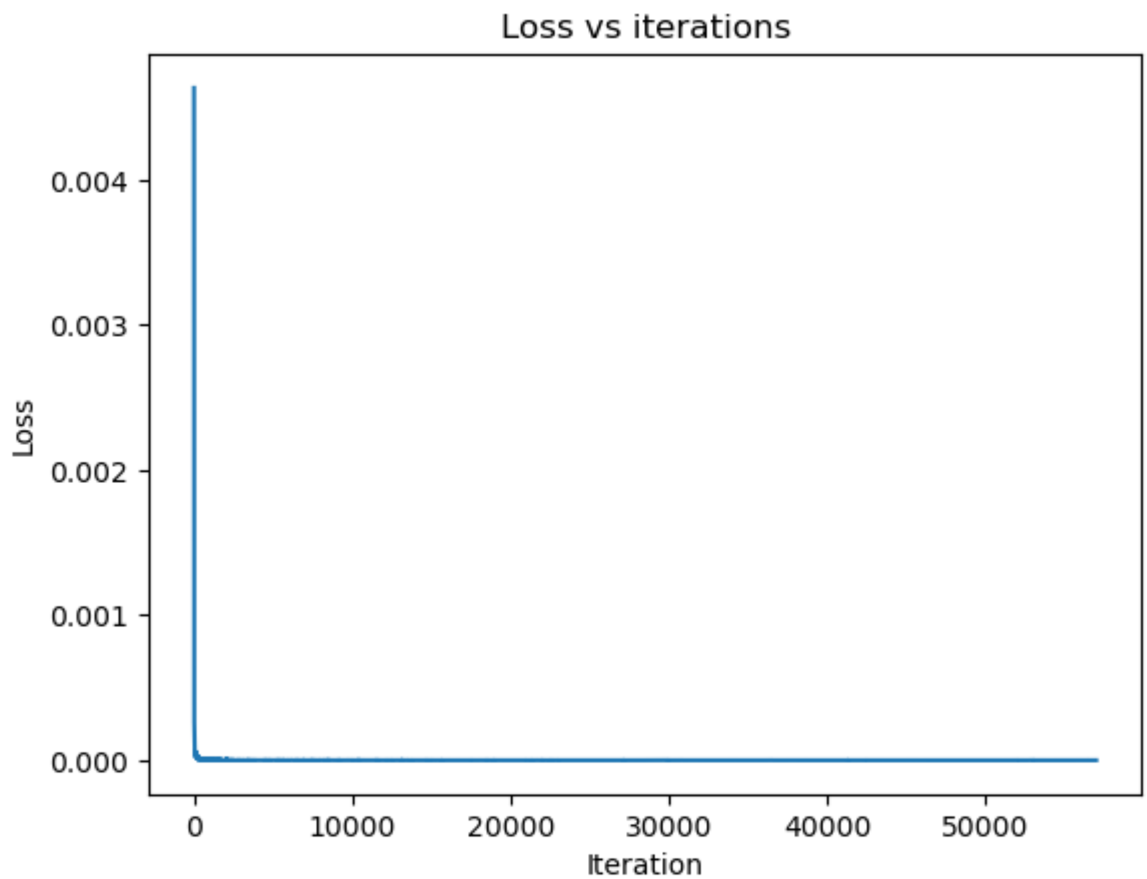
```
Epoch: [32/45], Batch: 200, Loss: 1.4213053134426445e-07
Epoch: [32/45], Batch: 400, Loss: 5.138400638315943e-08
Epoch: [32/45], Batch: 600, Loss: 3.627957028129458e-07
Epoch: [32/45], Batch: 800, Loss: 8.03711373009719e-08
Epoch: [32/45], Batch: 1000, Loss: 1.538057716743424e-07
Epoch: [32/45], Batch: 1200, Loss: 6.41756088270995e-08
Epoch: [33/45], Batch: 200, Loss: 1.7668745044829848e-07
Epoch: [33/45], Batch: 400, Loss: 1.1361547791466364e-07
Epoch: [33/45], Batch: 600, Loss: 9.173912474125245e-08
Epoch: [33/45], Batch: 800, Loss: 3.153822660806327e-07
Epoch: [33/45], Batch: 1000, Loss: 8.973096043973783e-08
Epoch: [33/45], Batch: 1200, Loss: 7.631145138020656e-08
Epoch: [34/45], Batch: 200, Loss: 5.611227393842455e-08
Epoch: [34/45], Batch: 400, Loss: 1.1855816239858541e-07
Epoch: [34/45], Batch: 600, Loss: 1.8868834672503e-07
Epoch: [34/45], Batch: 800, Loss: 8.650679461652544e-08
Epoch: [34/45], Batch: 1000, Loss: 7.016520697789019e-08
Epoch: [34/45], Batch: 1200, Loss: 6.350305170599313e-08
Epoch: [35/45], Batch: 200, Loss: 1.2389023140713107e-07
Epoch: [35/45], Batch: 400, Loss: 2.3439000074176874e-07
Epoch: [35/45], Batch: 600, Loss: 5.539822822697715e-08
Epoch: [35/45], Batch: 800, Loss: 1.8704417925619055e-07
Epoch: [35/45], Batch: 1000, Loss: 1.3577694346622593e-07
Epoch: [35/45], Batch: 1200, Loss: 1.1702405799951521e-07
Epoch: [36/45], Batch: 200, Loss: 1.4525305402912636e-07
Epoch: [36/45], Batch: 400, Loss: 6.998478596642599e-08
Epoch: [36/45], Batch: 600, Loss: 7.748236185989299e-08
Epoch: [36/45], Batch: 800, Loss: 1.288584599024034e-07
Epoch: [36/45], Batch: 1000, Loss: 1.7950348762951762e-07
Epoch: [36/45], Batch: 1200, Loss: 6.573502986384483e-08
Epoch: [37/45], Batch: 200, Loss: 6.964386756180829e-08
Epoch: [37/45], Batch: 400, Loss: 9.050972948898561e-08
Epoch: [37/45], Batch: 600, Loss: 7.402248058951955e-08
Epoch: [37/45], Batch: 800, Loss: 1.0235408609560182e-07
Epoch: [37/45], Batch: 1000, Loss: 1.4620408705923182e-07
Epoch: [37/45], Batch: 1200, Loss: 1.1216319961704357e-07
Epoch: [38/45], Batch: 200, Loss: 7.666930201821742e-08
Epoch: [38/45], Batch: 400, Loss: 7.043615823931759e-08
Epoch: [38/45], Batch: 600, Loss: 8.204606416484239e-08
Epoch: [38/45], Batch: 800, Loss: 9.027765202063165e-08
Epoch: [38/45], Batch: 1000, Loss: 7.183575689850841e-08
Epoch: [38/45], Batch: 1200, Loss: 8.00329829075963e-08
Epoch: [39/45], Batch: 200, Loss: 5.323461849116029e-08
Epoch: [39/45], Batch: 400, Loss: 5.137565040058689e-08
Epoch: [39/45], Batch: 600, Loss: 7.603181728654818e-08
Epoch: [39/45], Batch: 800, Loss: 1.816945456312169e-07
Epoch: [39/45], Batch: 1000, Loss: 8.928530093044174e-08
Epoch: [39/45], Batch: 1200, Loss: 9.657969712861814e-08
Epoch: [40/45], Batch: 200, Loss: 1.005958552013908e-07
Epoch: [40/45], Batch: 400, Loss: 1.6576400696521887e-07
Epoch: [40/45], Batch: 600, Loss: 1.442032271370408e-07
Epoch: [40/45], Batch: 800, Loss: 7.859640049900918e-08
Epoch: [40/45], Batch: 1000, Loss: 9.472785222897073e-08
Epoch: [40/45], Batch: 1200, Loss: 5.7033481937196484e-08
Epoch: [41/45], Batch: 200, Loss: 9.656254462697689e-08
Epoch: [41/45], Batch: 400, Loss: 8.711996457577698e-08
Epoch: [41/45], Batch: 600, Loss: 5.7156455568474485e-08
Epoch: [41/45], Batch: 800, Loss: 7.279953706529341e-08
Epoch: [41/45], Batch: 1000, Loss: 7.954677272437038e-08
Epoch: [41/45], Batch: 1200, Loss: 1.1135262667494317e-07
Epoch: [42/45], Batch: 200, Loss: 5.514426959507546e-08
Epoch: [42/45], Batch: 400, Loss: 4.13181311387234e-08
```

```
Epoch: [42/45], Batch: 600, Loss: 6.670732943803159e-08
Epoch: [42/45], Batch: 800, Loss: 1.1253098364250036e-07
Epoch: [42/45], Batch: 1000, Loss: 1.1357283113966332e-07
Epoch: [42/45], Batch: 1200, Loss: 6.257906903783805e-08
Epoch: [43/45], Batch: 200, Loss: 1.1900504404138701e-07
Epoch: [43/45], Batch: 400, Loss: 7.35422247544193e-08
Epoch: [43/45], Batch: 600, Loss: 2.970030550386582e-07
Epoch: [43/45], Batch: 800, Loss: 8.526841099865123e-08
Epoch: [43/45], Batch: 1000, Loss: 9.009828261241637e-08
Epoch: [43/45], Batch: 1200, Loss: 1.0789925397602929e-07
Epoch: [44/45], Batch: 200, Loss: 4.517924523383954e-08
Epoch: [44/45], Batch: 400, Loss: 8.079925351012207e-08
Epoch: [44/45], Batch: 600, Loss: 1.0159882180005297e-07
Epoch: [44/45], Batch: 800, Loss: 5.457639318251495e-08
Epoch: [44/45], Batch: 1000, Loss: 1.156581248551447e-07
Epoch: [44/45], Batch: 1200, Loss: 4.4154425893339067e-08
Epoch: [45/45], Batch: 200, Loss: 8.871925416542581e-08
Epoch: [45/45], Batch: 400, Loss: 5.5569088885931706e-08
Epoch: [45/45], Batch: 600, Loss: 1.6741935837671917e-07
Epoch: [45/45], Batch: 800, Loss: 6.097327087672966e-08
Epoch: [45/45], Batch: 1000, Loss: 1.7409195152140455e-07
Epoch: [45/45], Batch: 1200, Loss: 8.593719513783071e-08
Test L1 error: 93.60745577514172
Test L2 error: 798.8346153497696
```





t = 1
t = 2
t = 3
t = 4
t = 5
t = 6
t = 7
t = 8
t = 9
t = 10
t = 11
t = 12
t = 13
t = 14
t = 15
t = 16
t = 17
t = 18
r [m]
velocity [m/s]



Loss vs iterations
Loss
Iteration


```
In [ ]:
```