In [1]:

```
import numpy as np
import edf
from time import time
import sys
%matplotlib inline
import matplotlib.pyplot as plt
```

In [2]:

```
traindata = './mnist_data/train.npz'
valdata = './mnist_data/test.npz'

data = np.load(traindata)
t_imgs = np.float32(data['imgs'])/255.
t_labels = np.float32(data['labels'])

data = np.load(valdata)
v_imgs = np.float32(data['imgs'])/255.
v_labels = np.float32(data['labels'])
```

In [3]:

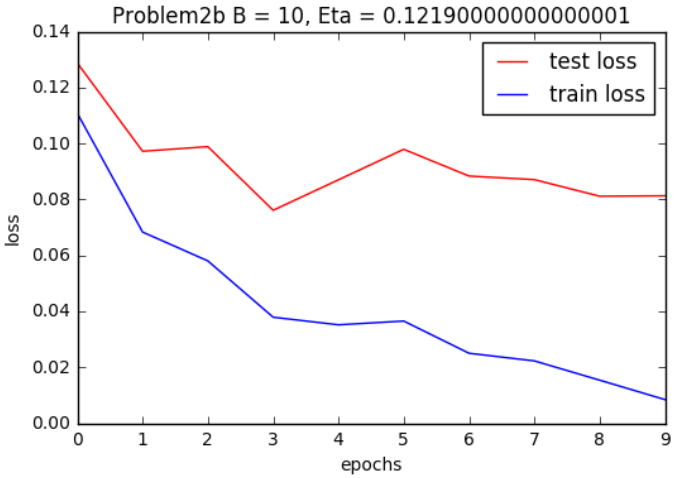
```
# for repeatability
np.random.seed(0)

batch_array = [10,100,50]
eta_array = [10 * 0.0056 + 0.0659, 0.37, 100 * 0.005
6 + 0.0659, 0.37, 50 * 0.0056 + 0.0659, 0.37]
#batch = 10
# learning rate eta, measured by per-batch unit. If
 you change this batch size, you might also change e
ta
# accoridng to the equation given in the homework.
#eta = batch * 0.0056 + 0.0659


for i in range(0,3):
    batch = batch_array[i]
    for j in range(i*2,i*2+2):
        eta = eta_array[j]
        edf.params = []
        edf.components = []


        # Inputs and parameters
        inp = edf.Value()
        lab = edf.Value()

        W1 = edf.Param(edf.xavier((28*28,128)))
        B1 = edf.Param(np.zeros((128)))
        W2 = edf.Param(edf.xavier((128,10)))
        B2 = edf.Param(np.zeros((10)))

        # models
        hidden = edf.RELU(edf.Add(edf.VDot(inp,W1),B
1))
        pred = edf.SoftMax(edf.Add(edf.VDot(hidden,W
2),B2))
        loss = edf.LogLoss(edf.Aref(pred,lab))
        acc = edf.Accuracy(pred,lab)

        # evaluate the random performance
        def eval(imgs, labels):
```

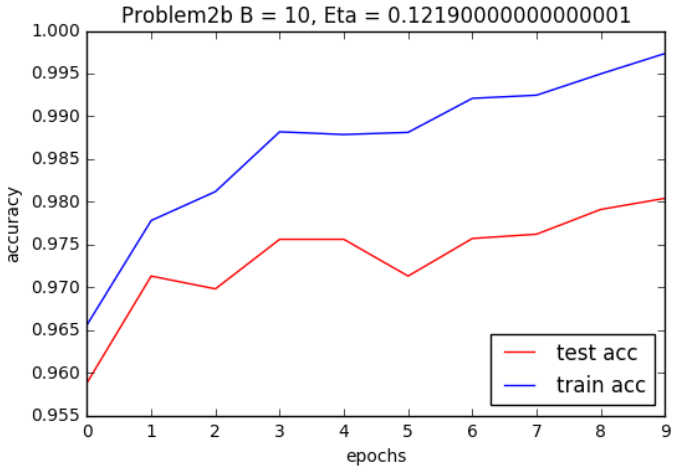
```
            batches = range(0, len(labels), batch)
            objective = 0
            accuracy = 0
            for k in batches:
                inp.set(imgs[k:k+batch])
                lab.set(labels[k:k+batch])
                edf.Forward()
                objective += np.mean(loss.value)
                accuracy += acc.value

            return accuracy/len(batches),
objective/len(batches)


        accuracy, objective = eval(t_imgs, t_labels)
        print("Random accuracy = %.4f" % accuracy)

        # train loop
        train_loss = []
        train_acc = []
        test_loss = []
        test_acc = []
        ep = 0
        stime = time()
        epoch = 10
        batches = range(0, len(t_labels), batch)

        while ep < epoch:

            # randon shuffle the train data in each
 epoch
            perm = np.random.permutation(len(t_label
s))
            for k in batches:
                inp.set(t_imgs[perm[k:k+batch]])
                lab.set(t_labels[perm[k:k+batch]])
                edf.Forward()
                edf.Backward(loss)

                # here, we use Momentum to optimize
 as in problem 2.b
                 # Initialize the "grad_hist" variab
```

```
le to memorize the history of gradient
                mom = 0.55
                if 'grad_hist' not in
edf.params[0].__dict__.keys():
                    for p in edf.params:
                        p.grad_hist = edf.DT(0)

                # please add code here to finish the
 function
                for p in edf.params:
                        p.grad_hist = mom * p.grad_h
ist + (1 - mom) * p.grad
                        p.value -= eta * p.grad_hist
                        p.grad = edf.DT(0)

            # evaluate on trainset
            t_acc, t_loss = eval(t_imgs, t_labels)
            print("Epoch %d: train loss = %.4f [%.3f
secs]" % (ep, t_loss,time()-stime))
            train_loss.append(t_loss)
            train_acc.append(t_acc)

            # evaluate on testset
            v_acc, v_loss = eval(v_imgs, v_labels)
            print("test accuracy=%.4f" % v_acc)
            test_loss.append(v_loss)
            test_acc.append(v_acc)
            stime = time()
            ep += 1

        # plot
        plt.figure(1)
        plt.xlabel("epochs")
        plt.ylabel("loss")
        plt.plot(np.arange(len(test_loss)), test_los
s, color='red')
        plt.plot(np.arange(len(train_loss)), train_l
oss, color='blue')
        plt.legend(['test loss', 'train loss'],
loc='upper right')
        plt.title("Problem2b B = {}, Eta = {}".forma
t(batch,eta))
```

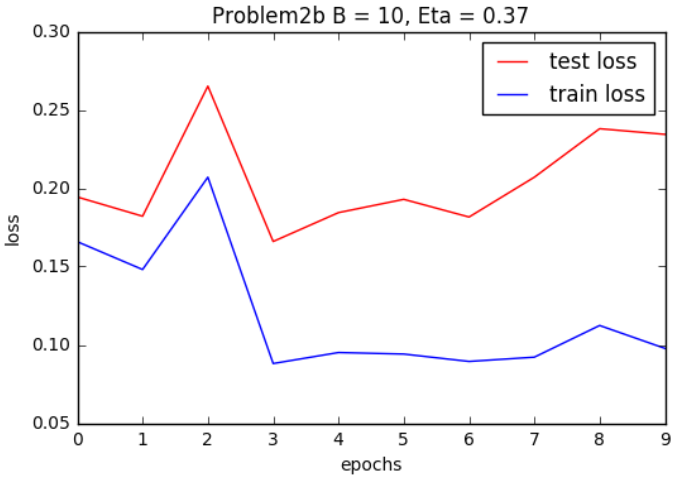
```
        plt.show()

        plt.figure(2)
        plt.xlabel("epochs")
        plt.ylabel("accuracy")
        plt.plot(np.arange(len(test_acc)), test_acc,
 color='red')
        plt.plot(np.arange(len(train_acc)), train_ac
c, color='blue')
        plt.legend(['test acc', 'train acc'], loc='l
ower right')
        plt.title("Problem2b B = {}, Eta = {}".forma
t(batch,eta))
        plt.show()
```

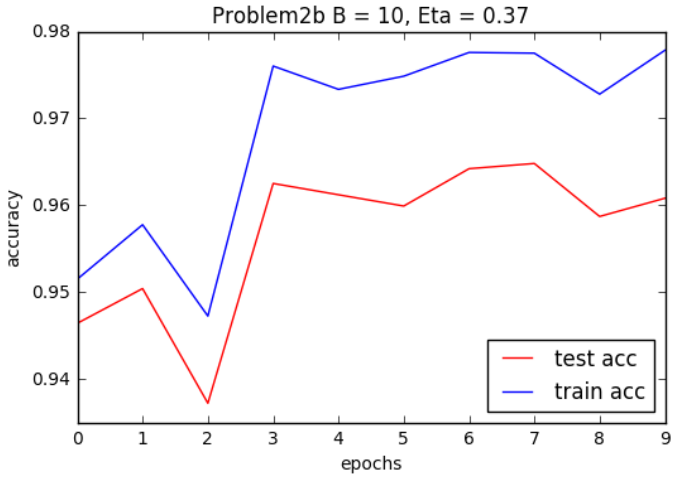
```
Random accuracy = 0.1329
Epoch 0: train loss = 0.1108 [27.840 sec
s]
test accuracy=0.9588
Epoch 1: train loss = 0.0683 [29.300 sec
s]
test accuracy=0.9713
Epoch 2: train loss = 0.0580 [30.065 sec
s]
test accuracy=0.9698
Epoch 3: train loss = 0.0379 [29.577 sec
s]
test accuracy=0.9756
Epoch 4: train loss = 0.0352 [29.459 sec
s]
test accuracy=0.9756
Epoch 5: train loss = 0.0365 [29.087 sec
s]
test accuracy=0.9713
Epoch 6: train loss = 0.0250 [29.718 sec
s]
test accuracy=0.9757
Epoch 7: train loss = 0.0223 [29.458 sec
s]
test accuracy=0.9762
Epoch 8: train loss = 0.0154 [29.805 sec
s]
test accuracy=0.9791
Epoch 9: train loss = 0.0084 [28.867 sec
s]
test accuracy=0.9804
```


Problem2b B = 10, Eta = 0.12190000000000001
test loss
train loss
loss
epochs
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
1
2
3
4
5
6
7
8
9

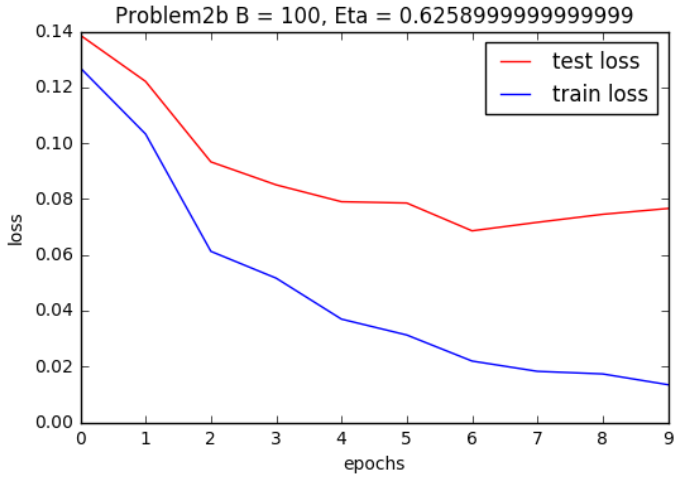

Problem2b B = 10, Eta = 0.12190000000000001
test acc
train acc
accuracy
epochs
1.000
0.995
0.990
0.985
0.980
0.975
0.970
0.965
0.960
0.955
0
1
2
3
4
5
6
7
8
9

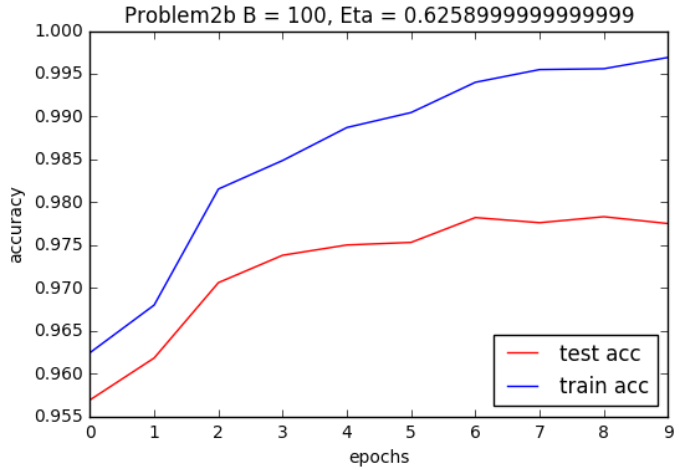
```
Random accuracy = 0.1498
Epoch 0: train loss = 0.1659 [35.164 sec
s]
test accuracy=0.9464
Epoch 1: train loss = 0.1481 [37.466 sec
s]
test accuracy=0.9504
Epoch 2: train loss = 0.2070 [37.141 sec
s]
test accuracy=0.9372
Epoch 3: train loss = 0.0881 [38.064 sec
s]
test accuracy=0.9625
Epoch 4: train loss = 0.0951 [38.239 sec
s]
test accuracy=0.9612
Epoch 5: train loss = 0.0941 [38.700 sec
s]
test accuracy=0.9599
Epoch 6: train loss = 0.0894 [39.435 sec
s]
test accuracy=0.9642
Epoch 7: train loss = 0.0922 [39.906 sec
s]
test accuracy=0.9648
Epoch 8: train loss = 0.1124 [40.595 sec
s]
test accuracy=0.9587
Epoch 9: train loss = 0.0977 [41.058 sec
s]
test accuracy=0.9608
```


Problem2b B = 10, Eta = 0.37
test loss
train loss
loss
epochs
0.30
0.25
0.20
0.15
0.10
0.05
0
1
2
3
4
5
6
7
8
9

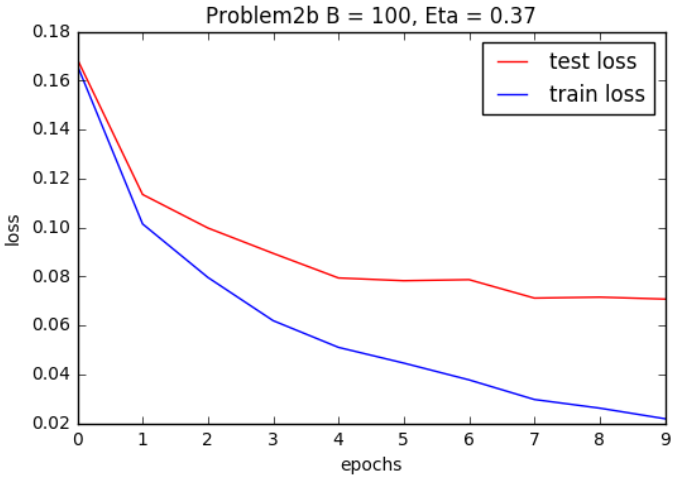

Problem2b B = 10, Eta = 0.37
test acc
train acc
accuracy
epochs
0.98
0.97
0.96
0.95
0.94
0
1
2
3
4
5
6
7
8
9

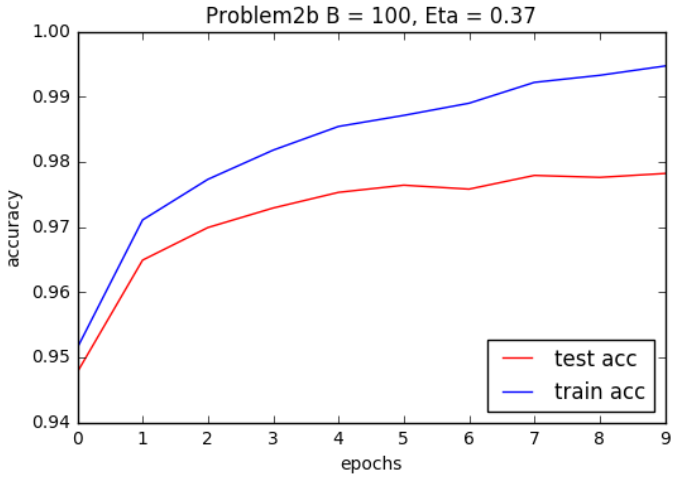
```
Random accuracy = 0.1110
Epoch 0: train loss = 0.1269 [6.135 secs]
test accuracy=0.9569
Epoch 1: train loss = 0.1033 [6.244 secs]
test accuracy=0.9618
Epoch 2: train loss = 0.0612 [6.713 secs]
test accuracy=0.9706
Epoch 3: train loss = 0.0516 [6.790 secs]
test accuracy=0.9738
Epoch 4: train loss = 0.0370 [6.729 secs]
test accuracy=0.9750
Epoch 5: train loss = 0.0313 [6.664 secs]
test accuracy=0.9753
Epoch 6: train loss = 0.0219 [6.616 secs]
test accuracy=0.9782
Epoch 7: train loss = 0.0183 [6.486 secs]
test accuracy=0.9776
Epoch 8: train loss = 0.0173 [6.456 secs]
test accuracy=0.9783
Epoch 9: train loss = 0.0135 [6.730 secs]
test accuracy=0.9775
```



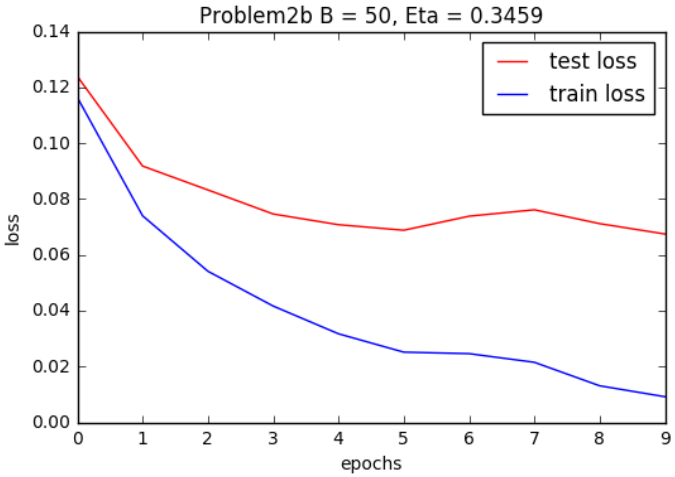


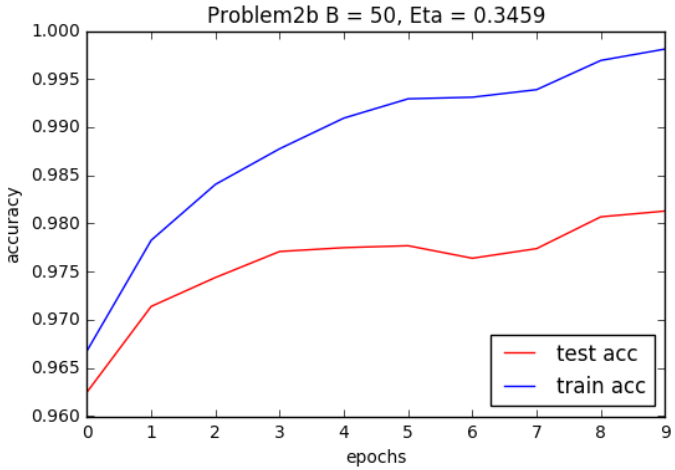

```
Random accuracy = 0.1019
Epoch 0: train loss = 0.1662 [5.855 secs]
test accuracy=0.9477
Epoch 1: train loss = 0.1014 [6.352 secs]
test accuracy=0.9649
Epoch 2: train loss = 0.0796 [6.343 secs]
test accuracy=0.9699
Epoch 3: train loss = 0.0619 [6.443 secs]
test accuracy=0.9729
Epoch 4: train loss = 0.0510 [6.025 secs]
test accuracy=0.9753
Epoch 5: train loss = 0.0446 [6.293 secs]
test accuracy=0.9764
Epoch 6: train loss = 0.0377 [6.505 secs]
test accuracy=0.9758
Epoch 7: train loss = 0.0297 [6.777 secs]
test accuracy=0.9779
Epoch 8: train loss = 0.0262 [6.676 secs]
test accuracy=0.9776
Epoch 9: train loss = 0.0218 [6.716 secs]
test accuracy=0.9782
```


Problem2b B = 100, Eta = 0.37
test loss
train loss
loss
epochs
0.18
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0
1
2
3
4
5
6
7
8
9

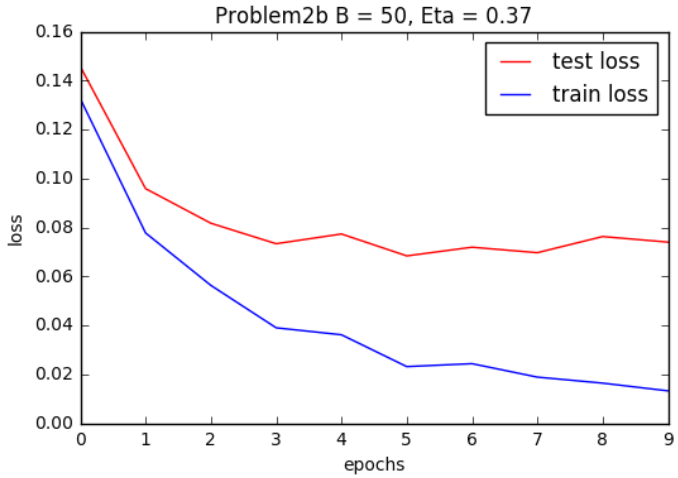

Problem2b B = 100, Eta = 0.37
test acc
train acc
accuracy
epochs
1.00
0.99
0.98
0.97
0.96
0.95
0.94
0
1
2
3
4
5
6
7
8
9

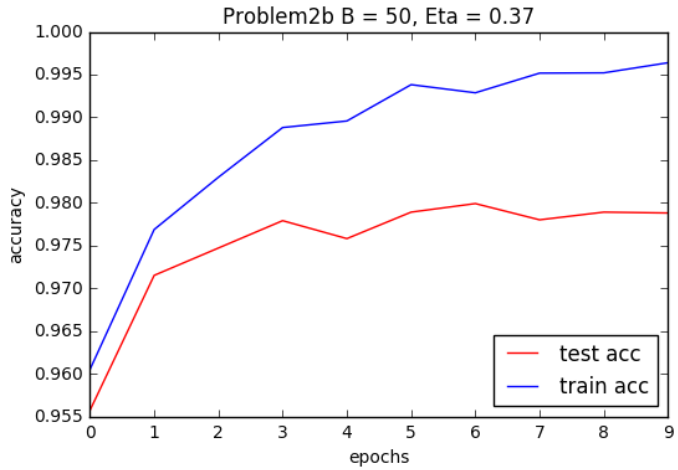
```
Random accuracy = 0.1153
Epoch 0: train loss = 0.1164 [7.651 secs]
test accuracy=0.9625
Epoch 1: train loss = 0.0740 [8.240 secs]
test accuracy=0.9714
Epoch 2: train loss = 0.0541 [8.383 secs]
test accuracy=0.9744
Epoch 3: train loss = 0.0416 [8.237 secs]
test accuracy=0.9771
Epoch 4: train loss = 0.0317 [8.242 secs]
test accuracy=0.9775
Epoch 5: train loss = 0.0251 [8.434 secs]
test accuracy=0.9777
Epoch 6: train loss = 0.0246 [8.154 secs]
test accuracy=0.9764
Epoch 7: train loss = 0.0215 [8.326 secs]
test accuracy=0.9774
Epoch 8: train loss = 0.0131 [8.304 secs]
test accuracy=0.9807
Epoch 9: train loss = 0.0092 [8.141 secs]
test accuracy=0.9813
```






```
Random accuracy = 0.1734
Epoch 0: train loss = 0.1326 [7.643 secs]
test accuracy=0.9557
Epoch 1: train loss = 0.0777 [8.277 secs]
test accuracy=0.9715
Epoch 2: train loss = 0.0563 [8.591 secs]
test accuracy=0.9747
Epoch 3: train loss = 0.0390 [8.663 secs]
test accuracy=0.9779
Epoch 4: train loss = 0.0361 [8.517 secs]
test accuracy=0.9758
Epoch 5: train loss = 0.0231 [8.358 secs]
test accuracy=0.9789
Epoch 6: train loss = 0.0243 [8.151 secs]
test accuracy=0.9799
Epoch 7: train loss = 0.0188 [8.357 secs]
test accuracy=0.9780
Epoch 8: train loss = 0.0164 [8.211 secs]
test accuracy=0.9789
Epoch 9: train loss = 0.0132 [8.200 secs]
test accuracy=0.9788
```


Problem2b B = 50, Eta = 0.37
test loss
train loss
loss
epochs
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
1
2
3
4
5
6
7
8
9


Problem2b B = 50, Eta = 0.37
test acc
train acc
accuracy
epochs
1.000
0.995
0.990
0.985
0.980
0.975
0.970
0.965
0.960
0.955
0
1
2
3
4
5
6
7
8
9

In [ ]: