

a01-fnn_solution

April 7, 2024

1 Work done by pair of students

- Nico Sharei (nsharei)
- Artem Bisliouk (abisliou)

```
[ ]: #  
# IE 678 Deep Learning, University of Mannheim  
# Author: Rainer Gemulla  
# Edited by:
```

```
[ ]: import math  
import matplotlib.pyplot as plt  
import torch  
import torch.nn as nn  
import torch.nn.functional as F  
  
# import helper functions  
import sys, os  
  
sys.path.append(os.getcwd())  
from a01helper import * # check out the helper functions there, if you like
```

2 Task 1: Implement an MLP

2.1 1a Logistic Regression

```
[ ]: # nn.Module is the superclass of all PyTorch models.  
class LogisticRegression(nn.Module):  
    """A logistic regression model.  
  
    Parameters  
    -----  
    D number of inputs  
    C number of classes  
    """  
  
    # the definition of all parameters the model uses happens here, i.e., during  
    # initialization
```

```

def __init__(self, D, C):
    super(LogisticRegression, self).__init__()

    # Create and initialize model parameters. For (multinomial) logistic
    ↪ regression,
    # we have a  $D \times C$ -dimensional weight matrix  $W$  and a  $C$ -dimensional bias  $b$ .
    self.W = torch.randn(D, C) / math.sqrt(D)
    self.b = torch.randn(C) / math.sqrt(C)

    # Model parameters must be registered to PyTorch as follows. Here we
    ↪ provide
    # a useful name that helps to access/analyze the model later on.
    self.register_parameter("0_weight", nn.Parameter(self.W))
    self.register_parameter("0_bias", nn.Parameter(self.b))

    # the forward function computes the model output for the provided (for this
    # assignment: single) input
    def forward(self, x):
        eta = self.W.t() @ x + self.b
        logprob = F.log_softmax(eta, dim=-1)
        return logprob

```

```

[ ]: # let's test it
logreg = LogisticRegression(3, 2)
x = torch.rand(3) # input
logreg(x) # output (log probabilities)
logreg(x).exp() # output (probabilities)

```

```

[ ]: tensor([0.8802, 0.1198])

```

```

[ ]: # you can access individual parameters as follows
logreg.get_parameter("0_bias")

```

```

[ ]: Parameter containing:
tensor([-0.5327, -1.3804], requires_grad=True)

```

```

[ ]: # or all of them at once
list(logreg.named_parameters())

```

```

[ ]: [('0_weight',
Parameter containing:
tensor([[ 0.6674,  0.1922],
        [ 0.1847,  0.3390],
        [-1.1552, -0.6920]], requires_grad=True)),
('0_bias',
Parameter containing:
tensor([0.2366, 0.1139], requires_grad=True))]

```

```
[ ]: # or directly the tensors stored in the parameters
for par, value in logreg.state_dict().items():
    print(f"{par:<15}= {value}")
```

```
0_weight      = tensor([[ 0.6674,  0.1922],
                        [ 0.1847,  0.3390],
                        [-1.1552, -0.6920]])
0_bias        = tensor([0.2366, 0.1139])
```

2.2 1b MLP

```
[ ]: class MLP(nn.Module):
    """A fully-connected MLP.

    Parameters
    -----

    sizes Contains the layer sizes. The first entry is the number of inputs,
    ↪the last
    entry the number of outputs. All entries in between correspond to the
    ↪number of
    units in the respective hidden layer. E.g., [2,5,7,1] means: 2 inputs -> 5D
    ↪hidden
    layer -> 7D hidden layer -> 1 output.

    phi Activation function used in every hidden layer (the output layer is
    ↪linear).

    """

    def __init__(self, sizes: list[int], phi=F.sigmoid):
        super().__init__()

        # let's remember the specification in this model
        self.sizes = sizes
        self.phi = phi

        for i in range(len(sizes) - 1):
            W = torch.randn(sizes[i], sizes[i+1]) / math.sqrt(sizes[i])
            self.register_parameter(f"{i}_weight", nn.Parameter(W))

            b = torch.randn(sizes[i+1])
            self.register_parameter(f"{i}_bias", nn.Parameter(b))

    def num_layers(self):
        """Number of layers (excluding input layer)"""
```

```

        return len(self.sizes) - 1

    def forward(self, x):
        out = x
        # iterate through each layer
        for i in range(self.num_layers()):
            # extract weight matrix and bias from current layer
            W = getattr(self, f"{i}_weight")
            b = getattr(self, f"{i}_bias")

            # apply matrix multiplication
            out = torch.matmul(out, W) + b

            # apply activation function to every layer besides the output layer
            if i < self.num_layers() - 1:
                out = self.phi(out)

        return out

```

```

[ ]: # here you should see the correct parameter sizes
mlp = MLP([2, 3, 4, 2], torch.relu)
list(mlp.named_parameters())

```

```

[ ]: [('0_weight',
      Parameter containing:
      tensor([[ 0.0835,  0.3357, -0.1311],
              [ 0.2920,  0.3099, -0.4164]], requires_grad=True)),
      ('0_bias',
      Parameter containing:
      tensor([ 0.7342, -0.8138,  0.7079], requires_grad=True)),
      ('1_weight',
      Parameter containing:
      tensor([[ -0.5385,  0.4991,  0.0204, -0.0382],
              [-0.3959,  0.0521,  0.1780,  0.6273],
              [ 0.4448,  0.4533, -0.1692,  0.7084]], requires_grad=True)),
      ('1_bias',
      Parameter containing:
      tensor([-0.1616, -1.4976, -1.5695, -0.4077], requires_grad=True)),
      ('2_weight',
      Parameter containing:
      tensor([[ 0.0626, -0.9391],
              [ 0.2034, -0.2423],
              [-0.0426, -0.5341],
              [ 0.5613, -0.3643]], requires_grad=True)),
      ('2_bias',
      Parameter containing:
      tensor([-0.8349, -0.2435], requires_grad=True))]

```

```
[ ]: # Test your code; we fix the parameters and check the result
with torch.no_grad():
    torch.manual_seed(0)
    for l in range(mlp.num_layers()):
        W, b = mlp.get_parameter(f"{l}_weight"), mlp.get_parameter(f"{l}_bias")
        W[:] = torch.randn(W.shape)
        b[:] = torch.randn(b.shape)

mlp(torch.tensor([-1.0, 2.0])) # must give: [ 0.8315, -3.6792]
```

```
[ ]: tensor([ 0.8315, -3.6792], grad_fn=<AddBackward0>)
```

```
[ ]: # You can also evaluate your model on multiple inputs at once. Here "torch.func.
    ↪vmap"
# produces a function that applies the provided function (mlp#forward) to each
    ↪row of
# its argument (torch.tensor...).
#
# [[ 0.8315, -3.6792],
# [ 4.8448, -6.8813]]
torch.func.vmap(mlp)(torch.tensor([-1.0, 2.0], [1.0, -2.0]))
```

```
[ ]: tensor([[ 0.8315, -3.6792],
            [ 4.8448, -6.8813]], grad_fn=<AddBackward0>)
```

2.3 1c Batching

```
[ ]: # After you adapted the MLP class, you should get the same results as above.
mlp(torch.tensor([-1.0, 2.0])) # must give: [ 0.8315, -3.6792]
```

```
[ ]: tensor([ 0.8315, -3.6792], grad_fn=<AddBackward0>)
```

```
[ ]: # Now without vmap. Only proceed to task 2 once this works correctly.
#
# [[ 0.8315, -3.6792],
# [ 4.8448, -6.8813]]
mlp(torch.tensor([-1.0, 2.0], [1.0, -2.0]))
```

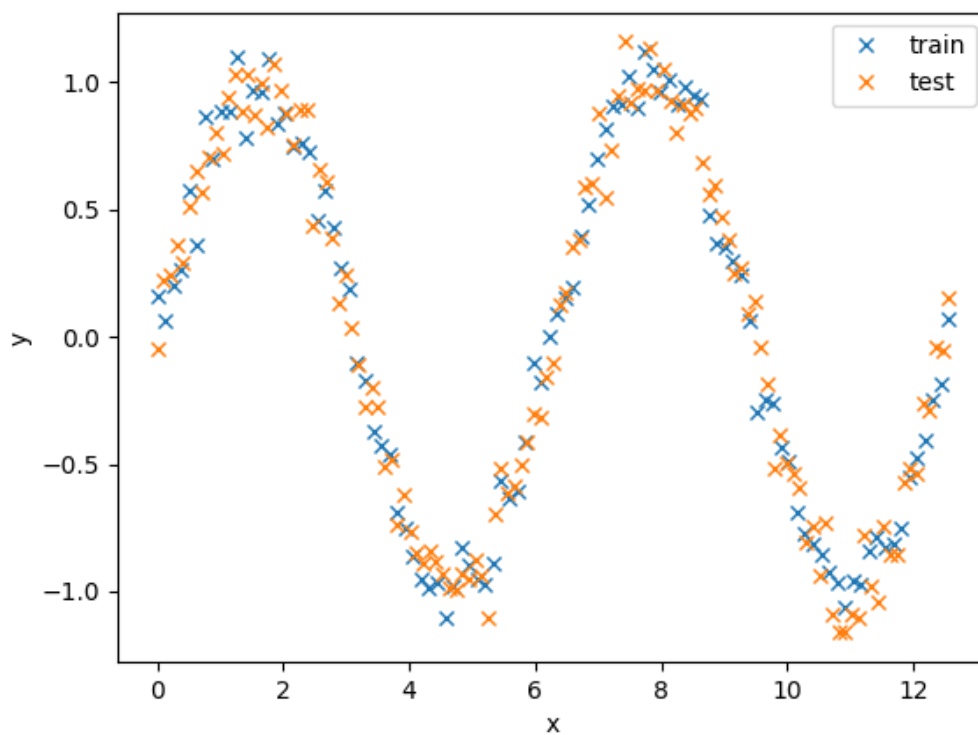
```
[ ]: tensor([[ 0.8315, -3.6792],
            [ 4.8448, -6.8813]], grad_fn=<AddBackward0>)
```

3 2 Multi-Layer Feed-Forward Neural Networks

3.1 2a Conjecture how an FNN fit will look like

```
[ ]: # here is the one-dimensional dataset that we will use
nextplot()
plot1(X1, y1, label="train")
plot1(X1test, y1test, label="test")
plt.legend()
```

```
[ ]: <matplotlib.legend.Legend at 0x13f8c9b10>
```



- zero: essentially, a linear regression function. Line Would go through the middle.
- one:
- two:
- three:

3.2 2b Train with 2 hidden units

```
[ ]: # Training code. You do not need to modify this code.
train_bfgs = lambda model, **kwargs: train_scipy(X1, y1, model, **kwargs)

def train1(hidden_sizes, nreps=10, phi=F.sigmoid, train=train_bfgs, **kwargs):
    """Train an FNN.

    hidden_sizes is a (possibly empty) list containing the sizes of the hidden_
    ↪layer(s).
    nreps refers to the number of repetitions.

    """
    best_model = None
    best_cost = math.inf
    for rep in range(nreps):
        model = MLP([1] + hidden_sizes + [1], phi) # that's your model!
        print(f"X1 shape: {X1.shape}")
        print(f"Repetition {rep: 2d}: ", end="")
        model = train(model, **kwargs)
        mse = F.mse_loss(y1, model(X1)).item()
        if mse < best_cost:
            best_model = model
            best_cost = mse
        print(f"best_cost={best_cost:.3f}")

    return best_model
```

```
[ ]: # Let's fit the model with one hidden layer consisting of 2 units.
model = train1([2], nreps=10)
print("Training error:", F.mse_loss(y1, model(X1)).item())
print("Test error      :", F.mse_loss(y1test, model(X1test)).item())
```

```
X1 shape: torch.Size([100, 1])
Repetition 0:          Current function value: 0.079573
      Iterations: 355
      Function evaluations: 515
      Gradient evaluations: 499
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 1:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
  res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
  res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```

        Current function value: 0.286909
        Iterations: 350
        Function evaluations: 470
        Gradient evaluations: 461
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 2: Optimization terminated successfully.
    Current function value: 0.301865
    Iterations: 156
    Function evaluations: 196
    Gradient evaluations: 196
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 3:          Current function value: 0.079573
    Iterations: 340
    Function evaluations: 474
    Gradient evaluations: 462
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 4:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.286909
        Iterations: 353
        Function evaluations: 582
        Gradient evaluations: 571
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 5: Optimization terminated successfully.
    Current function value: 0.357250
    Iterations: 83
    Function evaluations: 89
    Gradient evaluations: 89
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 6: Optimization terminated successfully.
    Current function value: 0.357250
    Iterations: 70
    Function evaluations: 78
    Gradient evaluations: 78
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 7:          Current function value: 0.079572

```



```

        Iterations: 432
        Function evaluations: 625
        Gradient evaluations: 609
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 8:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

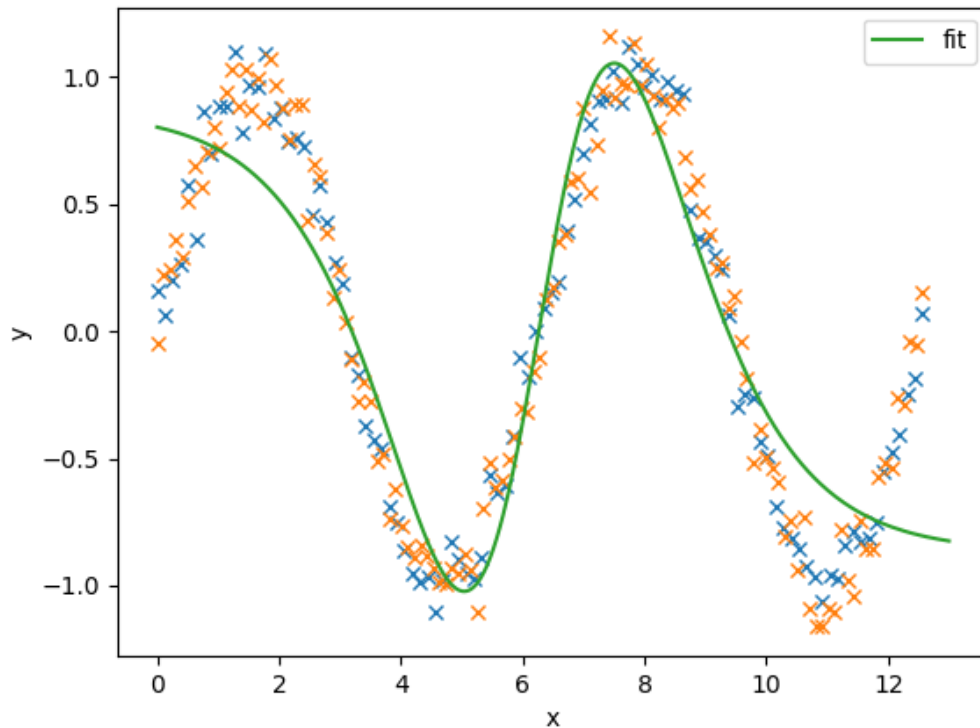
        Current function value: 0.079573
        Iterations: 550
        Function evaluations: 867
        Gradient evaluations: 853
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 9:          Current function value: 0.286909
        Iterations: 315
        Function evaluations: 458
        Gradient evaluations: 449
best_cost=0.080
Training error: 0.0795721486210823
Test error      : 0.08671437948942184

```

```

[ ]: # plot the data and the fit
nextplot()
plot1(X1, y1, label="train")
plot1(X1test, y1test, label="test")
plot1fit(torch.linspace(0, 13, 500).unsqueeze(1), model)

```



```
[ ]: # The weight matrices and bias vectors can be read out as follows. If you want, ↵
      ↵use
      # these parameters to compute the output of the network (on X1) directly and ↵
      ↵compare to
      # vmap(model)(X1).
      for par, value in model.state_dict().items():
          print(f"{par:<15}= {value}")
```

```
0_weight      = tensor([[1.0680, 1.0574]])
0_bias        = tensor([-6.7078, -6.6409])
1_weight      = tensor([[ 696.4487],
                        [-698.1782]])
1_bias        = tensor([0.8637])
```

```
[ ]: # now repeat this multiple times
models = {
    # "model_1_1" : train1([0], nreps=1),
    # "model_1_1" : train1([0], nreps=10),
    # "model_1_1" : train1([1], nreps=1),
    # "model_1_10" : train1([1], nreps=10),
    # "model_2_1" : train1([2], nreps=1),
```

```
# "model_2_10" : train1([2], nreps=10),
"model_3_1" : train1([3], nreps=1),
"model_3_10" : train1([3], nreps=10)
}
```

```
X1 shape: torch.Size([100, 1])
Repetition 0:          Current function value: 0.049879
      Iterations: 445
      Function evaluations: 663
      Gradient evaluations: 652
best_cost=0.050
X1 shape: torch.Size([100, 1])
Repetition 0:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
  res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
  res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

      Current function value: 0.049878
      Iterations: 527
      Function evaluations: 751
      Gradient evaluations: 740
best_cost=0.050
X1 shape: torch.Size([100, 1])
Repetition 1:          Current function value: 0.049877
      Iterations: 481
      Function evaluations: 650
      Gradient evaluations: 641
best_cost=0.050
X1 shape: torch.Size([100, 1])
Repetition 2:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
  res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
  res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
  res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

      Current function value: 0.116200
      Iterations: 480
      Function evaluations: 603
      Gradient evaluations: 592
best_cost=0.050
```

```

X1 shape: torch.Size([100, 1])
Repetition 3:          Current function value: 0.007324
                    Iterations: 338
                    Function evaluations: 470
                    Gradient evaluations: 458
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition 4:          Current function value: 0.275464
                    Iterations: 453
                    Function evaluations: 604
                    Gradient evaluations: 593
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition 5:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

                    Current function value: 0.049878
                    Iterations: 465
                    Function evaluations: 614
                    Gradient evaluations: 607
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition 6:          Current function value: 0.049893
                    Iterations: 300
                    Function evaluations: 418
                    Gradient evaluations: 407
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition 7:          Current function value: 0.079572
                    Iterations: 411
                    Function evaluations: 554
                    Gradient evaluations: 548
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition 8:          Current function value: 0.051868
                    Iterations: 269
                    Function evaluations: 518
                    Gradient evaluations: 504
best_cost=0.007
X1 shape: torch.Size([100, 1])

```

Repetition 9: Current function value: 0.079573

Iterations: 355

Function evaluations: 595

Gradient evaluations: 582

best_cost=0.007

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:

OptimizeWarning: Desired error not necessarily achieved due to precision loss.

res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:

OptimizeWarning: Desired error not necessarily achieved due to precision loss.

res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

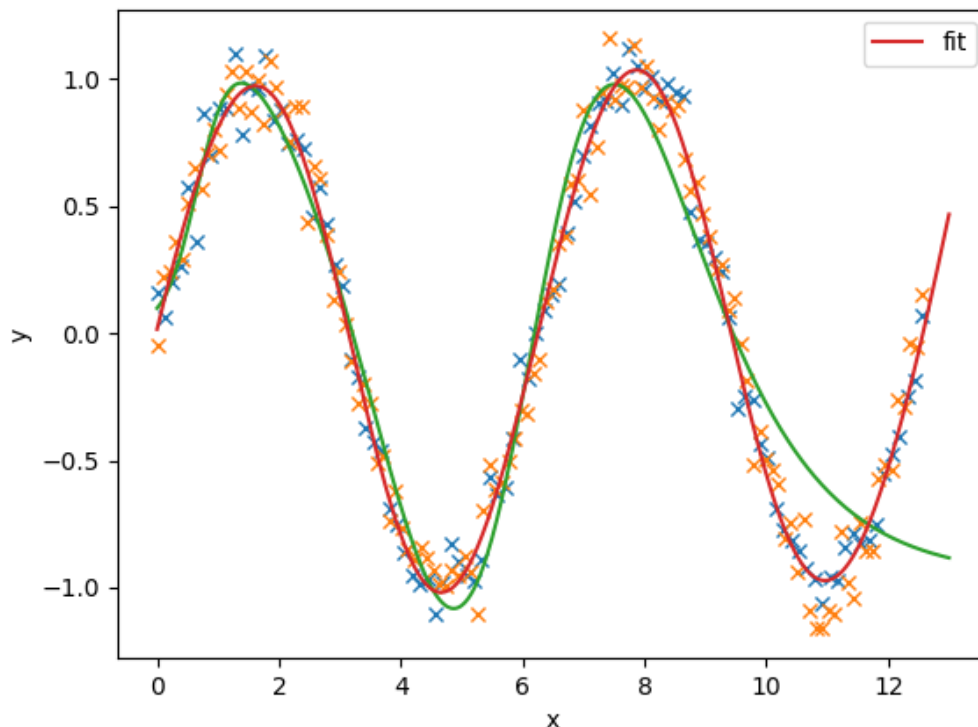
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:

OptimizeWarning: Desired error not necessarily achieved due to precision loss.

res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

```
[ ]: nextplot()
      plot1(X1, y1, label="train")
      plot1(X1test, y1test, label="test")

      for model in models.values():
          plot1fit(torch.linspace(0, 13, 500).unsqueeze(1), model)
```



3.3 2c Width

```
[ ]: # Experiment with different hidden layer sizes. To avoid recomputing
# models, you may want to save your models using torch.save(model, filename) and
# load them again using torch.load(filename).

# Let's fit the model with one hidden layer consisting of 2 units.
model1 = train1([1])
print("Training error:", F.mse_loss(y1, model1(X1)).item())
print("Test error      :", F.mse_loss(y1test, model1(X1test)).item())

torch.save(model1.state_dict(), "./models/model1.pt")
```

```
X1 shape: torch.Size([100, 1])
Repetition 0: Optimization terminated successfully.
    Current function value: 0.372919
    Iterations: 41
    Function evaluations: 44
    Gradient evaluations: 44
best_cost=0.373
X1 shape: torch.Size([100, 1])
Repetition 1: Optimization terminated successfully.
    Current function value: 0.372919
    Iterations: 37
    Function evaluations: 49
    Gradient evaluations: 49
best_cost=0.373
X1 shape: torch.Size([100, 1])
Repetition 2: Optimization terminated successfully.
    Current function value: 0.372919
    Iterations: 40
    Function evaluations: 46
    Gradient evaluations: 46
best_cost=0.373
X1 shape: torch.Size([100, 1])
Repetition 3: Optimization terminated successfully.
    Current function value: 0.372919
    Iterations: 44
    Function evaluations: 45
    Gradient evaluations: 45
best_cost=0.373
X1 shape: torch.Size([100, 1])
Repetition 4: Optimization terminated successfully.
    Current function value: 0.372919
    Iterations: 42
    Function evaluations: 44
```

```

        Gradient evaluations: 44
best_cost=0.373
X1 shape: torch.Size([100, 1])
Repetition 5: Optimization terminated successfully.
    Current function value: 0.372919
    Iterations: 49
    Function evaluations: 53
    Gradient evaluations: 53
best_cost=0.373
X1 shape: torch.Size([100, 1])
Repetition 6: Optimization terminated successfully.
    Current function value: 0.372919
    Iterations: 35
    Function evaluations: 40
    Gradient evaluations: 40
best_cost=0.373
X1 shape: torch.Size([100, 1])
Repetition 7: Optimization terminated successfully.
    Current function value: 0.372919
    Iterations: 44
    Function evaluations: 47
    Gradient evaluations: 47
best_cost=0.373
X1 shape: torch.Size([100, 1])
Repetition 8: Optimization terminated successfully.
    Current function value: 0.372919
    Iterations: 42
    Function evaluations: 47
    Gradient evaluations: 47
best_cost=0.373
X1 shape: torch.Size([100, 1])
Repetition 9: Optimization terminated successfully.
    Current function value: 0.372919
    Iterations: 44
    Function evaluations: 52
    Gradient evaluations: 52
best_cost=0.373
Training error: 0.37291884422302246
Test error      : 0.37431666254997253

```

```

[ ]: model2 = train1([2])
print("Training error:", F.mse_loss(y1, model2(X1)).item())
print("Test error      :", F.mse_loss(y1test, model2(X1test)).item())

torch.save(model2.state_dict(), "./models/model2.pt")

```

```

X1 shape: torch.Size([100, 1])
Repetition 0:      Current function value: 0.079572

```

```

        Iterations: 464
        Function evaluations: 751
        Gradient evaluations: 738
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 1:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.079573
        Iterations: 366
        Function evaluations: 509
        Gradient evaluations: 499
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 2:          Current function value: 0.079573
        Iterations: 322
        Function evaluations: 464
        Gradient evaluations: 456
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 3:          Current function value: 0.079572
        Iterations: 390
        Function evaluations: 601
        Gradient evaluations: 590
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 4:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.079573
        Iterations: 380
        Function evaluations: 501

```



```

        Gradient evaluations: 489
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 5: Optimization terminated successfully.
    Current function value: 0.302737
    Iterations: 130
    Function evaluations: 145
    Gradient evaluations: 145
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 6:      Current function value: 0.079573
    Iterations: 365
    Function evaluations: 492
    Gradient evaluations: 480
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 7:      Current function value: 0.079573
    Iterations: 388
    Function evaluations: 528
    Gradient evaluations: 519
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 8:      Current function value: 0.079573
    Iterations: 356
    Function evaluations: 528
    Gradient evaluations: 516
best_cost=0.080
X1 shape: torch.Size([100, 1])
Repetition 9:      Current function value: 0.079573
    Iterations: 370
    Function evaluations: 494
    Gradient evaluations: 482
best_cost=0.080
Training error: 0.0795714259147644
Test error      : 0.08671518415212631

```

```

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

```

```

[ ]: model3= train1([3])
      print("Training error:", F.mse_loss(y1, model3(X1)).item())

```

```
print("Test error      :", F.mse_loss(y1test, model3(X1test)).item())

torch.save(model3.state_dict(), "./models/model3.pt")
```

```
X1 shape: torch.Size([100, 1])
Repetition 0:      Current function value: 0.049877
      Iterations: 484
      Function evaluations: 642
      Gradient evaluations: 632
best_cost=0.050
X1 shape: torch.Size([100, 1])
Repetition 1:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

      Current function value: 0.049877
      Iterations: 584
      Function evaluations: 784
      Gradient evaluations: 772
best_cost=0.050
X1 shape: torch.Size([100, 1])
Repetition 2:      Current function value: 0.008496
      Iterations: 456
      Function evaluations: 685
      Gradient evaluations: 668
best_cost=0.008
X1 shape: torch.Size([100, 1])
Repetition 3:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

      Current function value: 0.053382
      Iterations: 511
      Function evaluations: 759
      Gradient evaluations: 748
best_cost=0.008
X1 shape: torch.Size([100, 1])
Repetition 4:      Current function value: 0.007324
      Iterations: 595
      Function evaluations: 806
```

```

        Gradient evaluations: 792
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition 5:
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.371771
        Iterations: 327
        Function evaluations: 470
        Gradient evaluations: 458
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition 6:          Current function value: 0.007792
        Iterations: 389
        Function evaluations: 608
        Gradient evaluations: 593
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition 7:          Current function value: 0.051894
        Iterations: 419
        Function evaluations: 539
        Gradient evaluations: 527
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition 8:          Current function value: 0.049877
        Iterations: 502
        Function evaluations: 628
        Gradient evaluations: 617
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition 9:          Current function value: 0.007324
        Iterations: 354
        Function evaluations: 451
        Gradient evaluations: 439
best_cost=0.007
Training error: 0.007324240170419216
Test error      : 0.010335271246731281

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.

```

```

    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

```

```

[ ]: model10= train1([10])
print("Training error:", F.mse_loss(y1, model10(X1)).item())
print("Test error      :", F.mse_loss(y1test, model10(X1test)).item())

torch.save(model10.state_dict(), "./models/model10.pt")

```

X1 shape: torch.Size([100, 1])

Repetition 0:

```

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

```

```

    Current function value: 0.006654
    Iterations: 1923
    Function evaluations: 2408
    Gradient evaluations: 2393

```

best_cost=0.007

X1 shape: torch.Size([100, 1])

Repetition 1:

```

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

```

```

    Current function value: 0.005709
    Iterations: 2753
    Function evaluations: 3149
    Gradient evaluations: 3136

```

best_cost=0.006

X1 shape: torch.Size([100, 1])

Repetition 2:

```

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

```

```

    Current function value: 0.006452
    Iterations: 3022
    Function evaluations: 3425
    Gradient evaluations: 3414

```

best_cost=0.006

X1 shape: torch.Size([100, 1])

Repetition 3:

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:  
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
        Current function value: 0.005936
```

```
        Iterations: 2293
```

```
        Function evaluations: 2601
```

```
        Gradient evaluations: 2590
```

```
best_cost=0.006
```

```
X1 shape: torch.Size([100, 1])
```

Repetition 4:

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:  
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
        Current function value: 0.006493
```

```
        Iterations: 2748
```

```
        Function evaluations: 3173
```

```
        Gradient evaluations: 3157
```

```
best_cost=0.006
```

```
X1 shape: torch.Size([100, 1])
```

Repetition 5:

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:  
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
        Current function value: 0.006827
```

```
        Iterations: 1563
```

```
        Function evaluations: 1819
```

```
        Gradient evaluations: 1807
```

```
best_cost=0.006
```

```
X1 shape: torch.Size([100, 1])
```

Repetition 6:

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:  
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
        Current function value: 0.006237
```

```
        Iterations: 3076
```

```
        Function evaluations: 3478
```

```
        Gradient evaluations: 3467
```

```
best_cost=0.006
```

```
X1 shape: torch.Size([100, 1])
```

Repetition 7:

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:  
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```

res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

    Current function value: 0.006660
    Iterations: 1110
    Function evaluations: 1305
    Gradient evaluations: 1294
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 8:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

    Current function value: 0.006638
    Iterations: 1792
    Function evaluations: 2141
    Gradient evaluations: 2130
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 9:          Current function value: 0.006652
    Iterations: 863
    Function evaluations: 1093
    Gradient evaluations: 1081
best_cost=0.006
Training error: 0.005708649288862944
Test error      : 0.023016968742012978

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

```

```

[ ]: model50= train1([50])
print("Training error:", F.mse_loss(y1, model50(X1)).item())
print("Test error      :", F.mse_loss(y1test, model50(X1test)).item())

torch.save(model50.state_dict(), "./models/model50.pt")

```

```

X1 shape: torch.Size([100, 1])
Repetition 0:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

    Current function value: 0.004760
    Iterations: 3795
    Function evaluations: 4283
    Gradient evaluations: 4272
best_cost=0.005
X1 shape: torch.Size([100, 1])

```

```

Repetition 1:
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.004108
        Iterations: 5131
        Function evaluations: 5687
        Gradient evaluations: 5676
best_cost=0.004
X1 shape: torch.Size([100, 1])
Repetition 2:
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.004358
        Iterations: 4048
        Function evaluations: 4494
        Gradient evaluations: 4486
best_cost=0.004
X1 shape: torch.Size([100, 1])
Repetition 3:
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.002975
        Iterations: 6598
        Function evaluations: 7335
        Gradient evaluations: 7322
best_cost=0.003
X1 shape: torch.Size([100, 1])
Repetition 4:
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.003348
        Iterations: 4949
        Function evaluations: 5502
        Gradient evaluations: 5492
best_cost=0.003
X1 shape: torch.Size([100, 1])
Repetition 5:
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Maximum number of iterations has been exceeded.

```

```

res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

    Current function value: 0.004299
    Iterations: 10000
    Function evaluations: 10844
    Gradient evaluations: 10844
best_cost=0.003
X1 shape: torch.Size([100, 1])
Repetition 6:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

    Current function value: 0.003333
    Iterations: 5263
    Function evaluations: 5736
    Gradient evaluations: 5726
best_cost=0.003
X1 shape: torch.Size([100, 1])
Repetition 7:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

    Current function value: 0.004762
    Iterations: 2570
    Function evaluations: 2906
    Gradient evaluations: 2894
best_cost=0.003
X1 shape: torch.Size([100, 1])
Repetition 8:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

    Current function value: 0.003223
    Iterations: 7202
    Function evaluations: 7949
    Gradient evaluations: 7930
best_cost=0.003
X1 shape: torch.Size([100, 1])
Repetition 9:          Current function value: 0.004972
    Iterations: 3782
    Function evaluations: 4249
    Gradient evaluations: 4238
best_cost=0.003
Training error: 0.002975039416924119
Test error      : 0.068841353058815

```



```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
[ ]: model100= train1([100])
print("Training error:", F.mse_loss(y1, model100(X1)).item())
print("Test error      :", F.mse_loss(y1test, model100(X1test)).item())

torch.save(model100.state_dict(), "./models/model100.pt")
```

```
X1 shape: torch.Size([100, 1])
```

```
Repetition 0:
```

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
Current function value: 0.004656
Iterations: 3504
Function evaluations: 3996
Gradient evaluations: 3984
```

```
best_cost=0.005
```

```
X1 shape: torch.Size([100, 1])
```

```
Repetition 1:
```

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
Current function value: 0.002520
Iterations: 8639
Function evaluations: 9477
Gradient evaluations: 9464
```

```
best_cost=0.003
```

```
X1 shape: torch.Size([100, 1])
```

```
Repetition 2:
```

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
Current function value: 0.003220
Iterations: 5656
Function evaluations: 6289
Gradient evaluations: 6275
```

```
best_cost=0.003
```

```
X1 shape: torch.Size([100, 1])
```

```
Repetition 3:
```

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Maximum number of iterations has been exceeded.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```

        Current function value: 0.001788
        Iterations: 10000
        Function evaluations: 10769
        Gradient evaluations: 10769
best_cost=0.002
X1 shape: torch.Size([100, 1])
Repetition 4:      Current function value: 0.002305
        Iterations: 10000
        Function evaluations: 10653
        Gradient evaluations: 10653
best_cost=0.002
X1 shape: torch.Size([100, 1])
Repetition 5:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.003947
        Iterations: 3706
        Function evaluations: 4161
        Gradient evaluations: 4149
best_cost=0.002
X1 shape: torch.Size([100, 1])
Repetition 6:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.003632
        Iterations: 5133
        Function evaluations: 5907
        Gradient evaluations: 5894
best_cost=0.002
X1 shape: torch.Size([100, 1])
Repetition 7:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.003517
        Iterations: 9020
        Function evaluations: 9972
        Gradient evaluations: 9961
best_cost=0.002
X1 shape: torch.Size([100, 1])
Repetition 8:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:

```

OptimizeWarning: Desired error not necessarily achieved due to precision loss.

```
res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
Current function value: 0.001982
```

```
Iterations: 9906
```

```
Function evaluations: 10761
```

```
Gradient evaluations: 10749
```

```
best_cost=0.002
```

```
X1 shape: torch.Size([100, 1])
```

```
Repetition 9: Current function value: 0.002907
```

```
Iterations: 7061
```

```
Function evaluations: 7670
```

```
Gradient evaluations: 7660
```

```
best_cost=0.002
```

```
Training error: 0.0017877757782116532
```

```
Test error : 0.7997748851776123
```

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:

OptimizeWarning: Desired error not necessarily achieved due to precision loss.

```
res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
[ ]: model1 = (MLP([1] + [1] + [1], F.sigmoid))
model1.load_state_dict(torch.load('./models/model1.pt'))

model2 = (MLP([1] + [2] + [1], F.sigmoid))
model2.load_state_dict(torch.load('./models/model2.pt'))

model3 = (MLP([1] + [3] + [1], F.sigmoid))
model3.load_state_dict(torch.load('./models/model3.pt'))

model10 = (MLP([1] + [10] + [1], F.sigmoid))
model10.load_state_dict(torch.load('./models/model10.pt'))

model50 = (MLP([1] + [50] + [1], F.sigmoid))
model50.load_state_dict(torch.load('./models/model50.pt'))

model100 = (MLP([1] + [100] + [1], F.sigmoid))
model100.load_state_dict(torch.load('./models/model100.pt'))
```

```
[ ]: <All keys matched successfully>
```

```
[ ]: model1
```

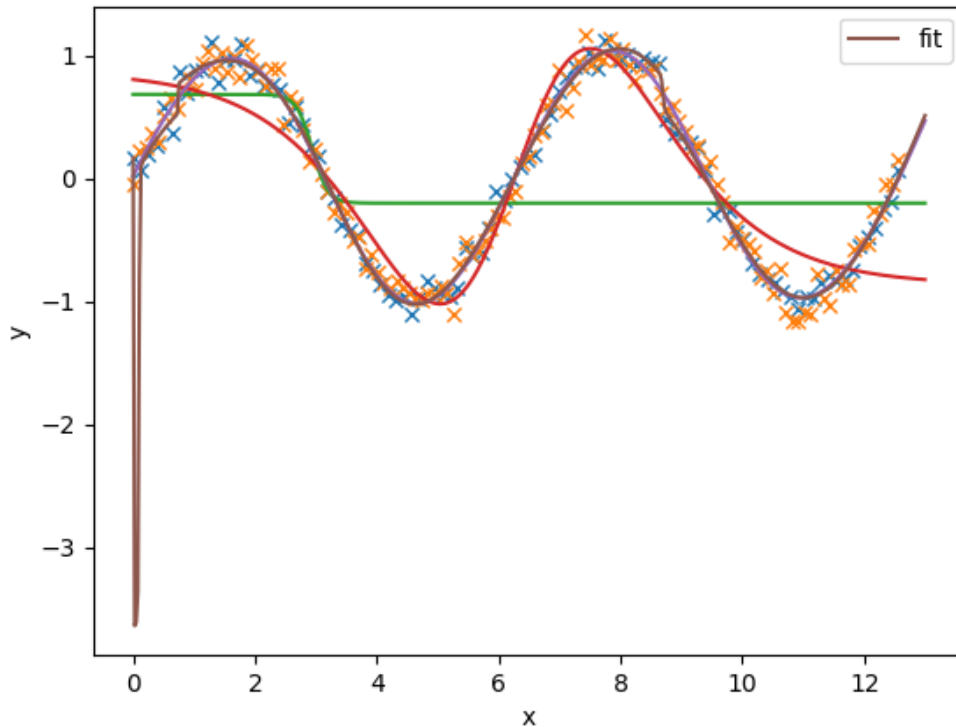
```
[ ]: MLP()
```

```
[ ]: nextplot()
plot1(X1, y1, label="train")
plot1(X1test, y1test, label="test")
```

```

plot1fit(torch.linspace(0, 13, 500).unsqueeze(1), model1)
plot1fit(torch.linspace(0, 13, 500).unsqueeze(1), model2)
plot1fit(torch.linspace(0, 13, 500).unsqueeze(1), model3)
plot1fit(torch.linspace(0, 13, 500).unsqueeze(1), model10)

```



3.4 2d Distributed representations

```

[ ]: # train a model to analyze
model = train1([3])

```

X1 shape: torch.Size([100, 1])

Repetition 0: Current function value: 0.049878

Iterations: 452

Function evaluations: 648

Gradient evaluations: 635

best_cost=0.050

X1 shape: torch.Size([100, 1])

Repetition 1:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:

OptimizeWarning: Desired error not necessarily achieved due to precision loss.

```

    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.051086
        Iterations: 474
        Function evaluations: 637
        Gradient evaluations: 627
best_cost=0.050
X1 shape: torch.Size([100, 1])
Repetition  2:          Current function value: 0.007324
        Iterations: 559
        Function evaluations: 779
        Gradient evaluations: 766
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition  3:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.049882
        Iterations: 283
        Function evaluations: 463
        Gradient evaluations: 446
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition  4:          Current function value: 0.007324
        Iterations: 672
        Function evaluations: 875
        Gradient evaluations: 862
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition  5:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.007324
        Iterations: 554
        Function evaluations: 688

```

```

        Gradient evaluations: 676
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition 6:          Current function value: 0.007324
        Iterations: 313
        Function evaluations: 530
        Gradient evaluations: 516
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition 7:          Current function value: 0.007324
        Iterations: 295
        Function evaluations: 407
        Gradient evaluations: 395
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition 8:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.007324
        Iterations: 394
        Function evaluations: 523
        Gradient evaluations: 510
best_cost=0.007
X1 shape: torch.Size([100, 1])
Repetition 9:          Current function value: 0.079573
        Iterations: 406
        Function evaluations: 555
        Gradient evaluations: 543
best_cost=0.007

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

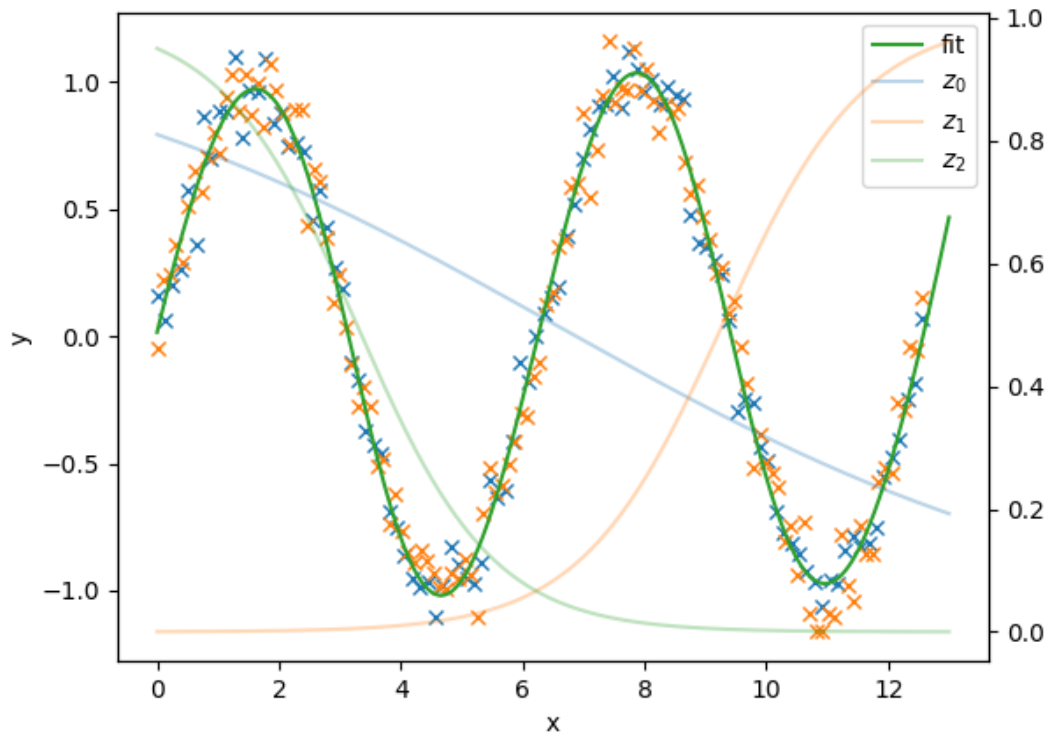
```

```

[ ]: # plot the fit as well as the outputs of each neuron in the hidden
     # layer (scale for the latter is shown on right y-axis)
     nextplot()
     plot1(X1, y1, label="train")
     plot1(X1test, y1test, label="test")

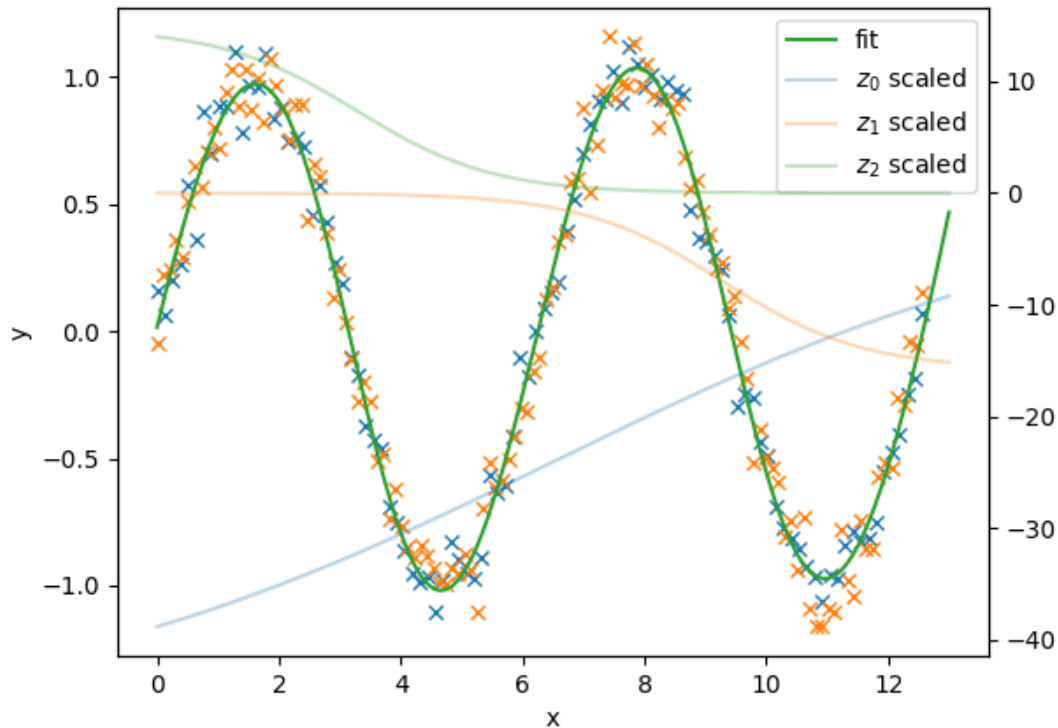
```

```
plot1fit(torch.linspace(0, 13, 500).unsqueeze(1), model, hidden=True,
↪scale=False)
```



```
[ ]: # plot the fit as well as the outputs of each neuron in the hidden layer, scaled
# by its weight for the output neuron (scale for the latter is shown on right
# y-axis)

nextplot()
plot1(X1, y1, label="train")
plot1(X1test, y1test, label="test")
plot1fit(torch.linspace(0, 13, 500).unsqueeze(1), model, hidden=True,
↪scale=True)
```



3.5 2e Experiment with different optimizers (optional)

```
[ ]: # PyTorch provides many gradient-based optimizers; see
# https://pytorch.org/docs/stable/optim.html. You can use a PyTorch optimizer
# as follows.
train_adam = lambda model, **kwargs: fnn_train(
    X1, y1, model, optimizer=torch.optim.Adam(model.parameters(), lr=0.01),
    **kwargs
)
model = train1([50], nreps=1, train=train_adam, max_epochs=5000, tol=1e-8,
    verbose=True)
```

```
X1 shape: torch.Size([100, 1])
Repetition 0: Epoch    0: cost=   1.293
Epoch    1: cost=   0.855
Epoch    2: cost=   0.581
Epoch    3: cost=   0.463
Epoch    4: cost=   0.468
Epoch    5: cost=   0.539
Epoch    6: cost=   0.612
Epoch    7: cost=   0.646
```


Epoch	8:	cost=	0.633
Epoch	9:	cost=	0.589
Epoch	10:	cost=	0.533
Epoch	11:	cost=	0.483
Epoch	12:	cost=	0.451
Epoch	13:	cost=	0.440
Epoch	14:	cost=	0.448
Epoch	15:	cost=	0.467
Epoch	16:	cost=	0.486
Epoch	17:	cost=	0.498
Epoch	18:	cost=	0.499
Epoch	19:	cost=	0.489
Epoch	20:	cost=	0.473
Epoch	21:	cost=	0.456
Epoch	22:	cost=	0.442
Epoch	23:	cost=	0.435
Epoch	24:	cost=	0.435
Epoch	25:	cost=	0.440
Epoch	26:	cost=	0.447
Epoch	27:	cost=	0.452
Epoch	28:	cost=	0.454
Epoch	29:	cost=	0.451
Epoch	30:	cost=	0.445
Epoch	31:	cost=	0.438
Epoch	32:	cost=	0.432
Epoch	33:	cost=	0.429
Epoch	34:	cost=	0.428
Epoch	35:	cost=	0.430
Epoch	36:	cost=	0.432
Epoch	37:	cost=	0.433
Epoch	38:	cost=	0.434
Epoch	39:	cost=	0.432
Epoch	40:	cost=	0.429
Epoch	41:	cost=	0.426
Epoch	42:	cost=	0.424
Epoch	43:	cost=	0.422
Epoch	44:	cost=	0.422
Epoch	45:	cost=	0.423
Epoch	46:	cost=	0.423
Epoch	47:	cost=	0.424
Epoch	48:	cost=	0.423
Epoch	49:	cost=	0.421
Epoch	50:	cost=	0.420
Epoch	51:	cost=	0.418
Epoch	52:	cost=	0.417
Epoch	53:	cost=	0.417
Epoch	54:	cost=	0.417
Epoch	55:	cost=	0.417

Epoch	56:	cost=	0.416
Epoch	57:	cost=	0.416
Epoch	58:	cost=	0.415
Epoch	59:	cost=	0.414
Epoch	60:	cost=	0.413
Epoch	61:	cost=	0.412
Epoch	62:	cost=	0.412
Epoch	63:	cost=	0.411
Epoch	64:	cost=	0.411
Epoch	65:	cost=	0.410
Epoch	66:	cost=	0.410
Epoch	67:	cost=	0.409
Epoch	68:	cost=	0.408
Epoch	69:	cost=	0.408
Epoch	70:	cost=	0.407
Epoch	71:	cost=	0.407
Epoch	72:	cost=	0.406
Epoch	73:	cost=	0.405
Epoch	74:	cost=	0.405
Epoch	75:	cost=	0.404
Epoch	76:	cost=	0.403
Epoch	77:	cost=	0.403
Epoch	78:	cost=	0.402
Epoch	79:	cost=	0.402
Epoch	80:	cost=	0.401
Epoch	81:	cost=	0.401
Epoch	82:	cost=	0.400
Epoch	83:	cost=	0.399
Epoch	84:	cost=	0.399
Epoch	85:	cost=	0.398
Epoch	86:	cost=	0.397
Epoch	87:	cost=	0.397
Epoch	88:	cost=	0.396
Epoch	89:	cost=	0.396
Epoch	90:	cost=	0.395
Epoch	91:	cost=	0.395
Epoch	92:	cost=	0.394
Epoch	93:	cost=	0.393
Epoch	94:	cost=	0.393
Epoch	95:	cost=	0.392
Epoch	96:	cost=	0.392
Epoch	97:	cost=	0.391
Epoch	98:	cost=	0.391
Epoch	99:	cost=	0.390
Epoch	100:	cost=	0.390
Epoch	101:	cost=	0.389
Epoch	102:	cost=	0.389
Epoch	103:	cost=	0.388

Epoch	104:	cost=	0.388
Epoch	105:	cost=	0.387
Epoch	106:	cost=	0.387
Epoch	107:	cost=	0.386
Epoch	108:	cost=	0.386
Epoch	109:	cost=	0.385
Epoch	110:	cost=	0.385
Epoch	111:	cost=	0.385
Epoch	112:	cost=	0.384
Epoch	113:	cost=	0.384
Epoch	114:	cost=	0.383
Epoch	115:	cost=	0.383
Epoch	116:	cost=	0.382
Epoch	117:	cost=	0.382
Epoch	118:	cost=	0.382
Epoch	119:	cost=	0.381
Epoch	120:	cost=	0.381
Epoch	121:	cost=	0.381
Epoch	122:	cost=	0.380
Epoch	123:	cost=	0.380
Epoch	124:	cost=	0.379
Epoch	125:	cost=	0.379
Epoch	126:	cost=	0.379
Epoch	127:	cost=	0.378
Epoch	128:	cost=	0.378
Epoch	129:	cost=	0.378
Epoch	130:	cost=	0.377
Epoch	131:	cost=	0.377
Epoch	132:	cost=	0.377
Epoch	133:	cost=	0.376
Epoch	134:	cost=	0.376
Epoch	135:	cost=	0.376
Epoch	136:	cost=	0.375
Epoch	137:	cost=	0.375
Epoch	138:	cost=	0.375
Epoch	139:	cost=	0.375
Epoch	140:	cost=	0.374
Epoch	141:	cost=	0.374
Epoch	142:	cost=	0.374
Epoch	143:	cost=	0.373
Epoch	144:	cost=	0.373
Epoch	145:	cost=	0.373
Epoch	146:	cost=	0.372
Epoch	147:	cost=	0.372
Epoch	148:	cost=	0.372
Epoch	149:	cost=	0.371
Epoch	150:	cost=	0.371
Epoch	151:	cost=	0.371

Epoch	152:	cost=	0.370
Epoch	153:	cost=	0.370
Epoch	154:	cost=	0.370
Epoch	155:	cost=	0.370
Epoch	156:	cost=	0.369
Epoch	157:	cost=	0.369
Epoch	158:	cost=	0.369
Epoch	159:	cost=	0.368
Epoch	160:	cost=	0.368
Epoch	161:	cost=	0.368
Epoch	162:	cost=	0.367
Epoch	163:	cost=	0.367
Epoch	164:	cost=	0.367
Epoch	165:	cost=	0.367
Epoch	166:	cost=	0.366
Epoch	167:	cost=	0.366
Epoch	168:	cost=	0.366
Epoch	169:	cost=	0.365
Epoch	170:	cost=	0.365
Epoch	171:	cost=	0.365
Epoch	172:	cost=	0.364
Epoch	173:	cost=	0.364
Epoch	174:	cost=	0.364
Epoch	175:	cost=	0.364
Epoch	176:	cost=	0.363
Epoch	177:	cost=	0.363
Epoch	178:	cost=	0.363
Epoch	179:	cost=	0.362
Epoch	180:	cost=	0.362
Epoch	181:	cost=	0.362
Epoch	182:	cost=	0.361
Epoch	183:	cost=	0.361
Epoch	184:	cost=	0.361
Epoch	185:	cost=	0.361
Epoch	186:	cost=	0.360
Epoch	187:	cost=	0.360
Epoch	188:	cost=	0.360
Epoch	189:	cost=	0.359
Epoch	190:	cost=	0.359
Epoch	191:	cost=	0.359
Epoch	192:	cost=	0.358
Epoch	193:	cost=	0.358
Epoch	194:	cost=	0.358
Epoch	195:	cost=	0.357
Epoch	196:	cost=	0.357
Epoch	197:	cost=	0.357
Epoch	198:	cost=	0.357
Epoch	199:	cost=	0.356

Epoch	200:	cost=	0.356
Epoch	201:	cost=	0.356
Epoch	202:	cost=	0.355
Epoch	203:	cost=	0.355
Epoch	204:	cost=	0.355
Epoch	205:	cost=	0.354
Epoch	206:	cost=	0.354
Epoch	207:	cost=	0.354
Epoch	208:	cost=	0.353
Epoch	209:	cost=	0.353
Epoch	210:	cost=	0.353
Epoch	211:	cost=	0.352
Epoch	212:	cost=	0.352
Epoch	213:	cost=	0.352
Epoch	214:	cost=	0.351
Epoch	215:	cost=	0.351
Epoch	216:	cost=	0.351
Epoch	217:	cost=	0.350
Epoch	218:	cost=	0.350
Epoch	219:	cost=	0.349
Epoch	220:	cost=	0.349
Epoch	221:	cost=	0.349
Epoch	222:	cost=	0.348
Epoch	223:	cost=	0.348
Epoch	224:	cost=	0.348
Epoch	225:	cost=	0.347
Epoch	226:	cost=	0.347
Epoch	227:	cost=	0.347
Epoch	228:	cost=	0.346
Epoch	229:	cost=	0.346
Epoch	230:	cost=	0.345
Epoch	231:	cost=	0.345
Epoch	232:	cost=	0.345
Epoch	233:	cost=	0.344
Epoch	234:	cost=	0.344
Epoch	235:	cost=	0.343
Epoch	236:	cost=	0.343
Epoch	237:	cost=	0.343
Epoch	238:	cost=	0.342
Epoch	239:	cost=	0.342
Epoch	240:	cost=	0.341
Epoch	241:	cost=	0.341
Epoch	242:	cost=	0.341
Epoch	243:	cost=	0.340
Epoch	244:	cost=	0.340
Epoch	245:	cost=	0.339
Epoch	246:	cost=	0.339
Epoch	247:	cost=	0.338

Epoch	248:	cost=	0.338
Epoch	249:	cost=	0.338
Epoch	250:	cost=	0.337
Epoch	251:	cost=	0.337
Epoch	252:	cost=	0.336
Epoch	253:	cost=	0.336
Epoch	254:	cost=	0.335
Epoch	255:	cost=	0.335
Epoch	256:	cost=	0.334
Epoch	257:	cost=	0.334
Epoch	258:	cost=	0.333
Epoch	259:	cost=	0.333
Epoch	260:	cost=	0.333
Epoch	261:	cost=	0.332
Epoch	262:	cost=	0.332
Epoch	263:	cost=	0.331
Epoch	264:	cost=	0.331
Epoch	265:	cost=	0.330
Epoch	266:	cost=	0.330
Epoch	267:	cost=	0.329
Epoch	268:	cost=	0.329
Epoch	269:	cost=	0.328
Epoch	270:	cost=	0.327
Epoch	271:	cost=	0.327
Epoch	272:	cost=	0.326
Epoch	273:	cost=	0.326
Epoch	274:	cost=	0.325
Epoch	275:	cost=	0.325
Epoch	276:	cost=	0.324
Epoch	277:	cost=	0.324
Epoch	278:	cost=	0.323
Epoch	279:	cost=	0.323
Epoch	280:	cost=	0.322
Epoch	281:	cost=	0.321
Epoch	282:	cost=	0.321
Epoch	283:	cost=	0.320
Epoch	284:	cost=	0.320
Epoch	285:	cost=	0.319
Epoch	286:	cost=	0.319
Epoch	287:	cost=	0.318
Epoch	288:	cost=	0.317
Epoch	289:	cost=	0.317
Epoch	290:	cost=	0.316
Epoch	291:	cost=	0.315
Epoch	292:	cost=	0.315
Epoch	293:	cost=	0.314
Epoch	294:	cost=	0.314
Epoch	295:	cost=	0.313

Epoch	296:	cost=	0.312
Epoch	297:	cost=	0.312
Epoch	298:	cost=	0.311
Epoch	299:	cost=	0.310
Epoch	300:	cost=	0.310
Epoch	301:	cost=	0.309
Epoch	302:	cost=	0.308
Epoch	303:	cost=	0.308
Epoch	304:	cost=	0.307
Epoch	305:	cost=	0.306
Epoch	306:	cost=	0.306
Epoch	307:	cost=	0.305
Epoch	308:	cost=	0.304
Epoch	309:	cost=	0.303
Epoch	310:	cost=	0.303
Epoch	311:	cost=	0.302
Epoch	312:	cost=	0.301
Epoch	313:	cost=	0.301
Epoch	314:	cost=	0.300
Epoch	315:	cost=	0.299
Epoch	316:	cost=	0.298
Epoch	317:	cost=	0.298
Epoch	318:	cost=	0.297
Epoch	319:	cost=	0.296
Epoch	320:	cost=	0.295
Epoch	321:	cost=	0.294
Epoch	322:	cost=	0.294
Epoch	323:	cost=	0.293
Epoch	324:	cost=	0.292
Epoch	325:	cost=	0.291
Epoch	326:	cost=	0.290
Epoch	327:	cost=	0.290
Epoch	328:	cost=	0.289
Epoch	329:	cost=	0.288
Epoch	330:	cost=	0.287
Epoch	331:	cost=	0.286
Epoch	332:	cost=	0.285
Epoch	333:	cost=	0.285
Epoch	334:	cost=	0.284
Epoch	335:	cost=	0.283
Epoch	336:	cost=	0.282
Epoch	337:	cost=	0.281
Epoch	338:	cost=	0.280
Epoch	339:	cost=	0.279
Epoch	340:	cost=	0.279
Epoch	341:	cost=	0.278
Epoch	342:	cost=	0.277
Epoch	343:	cost=	0.276

Epoch	344:	cost=	0.275
Epoch	345:	cost=	0.274
Epoch	346:	cost=	0.273
Epoch	347:	cost=	0.272
Epoch	348:	cost=	0.271
Epoch	349:	cost=	0.270
Epoch	350:	cost=	0.269
Epoch	351:	cost=	0.268
Epoch	352:	cost=	0.267
Epoch	353:	cost=	0.266
Epoch	354:	cost=	0.265
Epoch	355:	cost=	0.264
Epoch	356:	cost=	0.263
Epoch	357:	cost=	0.263
Epoch	358:	cost=	0.262
Epoch	359:	cost=	0.261
Epoch	360:	cost=	0.260
Epoch	361:	cost=	0.259
Epoch	362:	cost=	0.258
Epoch	363:	cost=	0.256
Epoch	364:	cost=	0.255
Epoch	365:	cost=	0.254
Epoch	366:	cost=	0.253
Epoch	367:	cost=	0.252
Epoch	368:	cost=	0.251
Epoch	369:	cost=	0.250
Epoch	370:	cost=	0.249
Epoch	371:	cost=	0.248
Epoch	372:	cost=	0.247
Epoch	373:	cost=	0.246
Epoch	374:	cost=	0.245
Epoch	375:	cost=	0.244
Epoch	376:	cost=	0.243
Epoch	377:	cost=	0.242
Epoch	378:	cost=	0.241
Epoch	379:	cost=	0.240
Epoch	380:	cost=	0.239
Epoch	381:	cost=	0.237
Epoch	382:	cost=	0.236
Epoch	383:	cost=	0.235
Epoch	384:	cost=	0.234
Epoch	385:	cost=	0.233
Epoch	386:	cost=	0.232
Epoch	387:	cost=	0.231
Epoch	388:	cost=	0.230
Epoch	389:	cost=	0.229
Epoch	390:	cost=	0.228
Epoch	391:	cost=	0.226

Epoch	392:	cost=	0.225
Epoch	393:	cost=	0.224
Epoch	394:	cost=	0.223
Epoch	395:	cost=	0.222
Epoch	396:	cost=	0.221
Epoch	397:	cost=	0.220
Epoch	398:	cost=	0.219
Epoch	399:	cost=	0.217
Epoch	400:	cost=	0.216
Epoch	401:	cost=	0.215
Epoch	402:	cost=	0.214
Epoch	403:	cost=	0.213
Epoch	404:	cost=	0.212
Epoch	405:	cost=	0.211
Epoch	406:	cost=	0.210
Epoch	407:	cost=	0.208
Epoch	408:	cost=	0.207
Epoch	409:	cost=	0.206
Epoch	410:	cost=	0.205
Epoch	411:	cost=	0.204
Epoch	412:	cost=	0.203
Epoch	413:	cost=	0.202
Epoch	414:	cost=	0.200
Epoch	415:	cost=	0.199
Epoch	416:	cost=	0.198
Epoch	417:	cost=	0.197
Epoch	418:	cost=	0.196
Epoch	419:	cost=	0.195
Epoch	420:	cost=	0.194
Epoch	421:	cost=	0.192
Epoch	422:	cost=	0.191
Epoch	423:	cost=	0.190
Epoch	424:	cost=	0.189
Epoch	425:	cost=	0.188
Epoch	426:	cost=	0.187
Epoch	427:	cost=	0.186
Epoch	428:	cost=	0.185
Epoch	429:	cost=	0.183
Epoch	430:	cost=	0.182
Epoch	431:	cost=	0.181
Epoch	432:	cost=	0.180
Epoch	433:	cost=	0.179
Epoch	434:	cost=	0.178
Epoch	435:	cost=	0.177
Epoch	436:	cost=	0.176
Epoch	437:	cost=	0.175
Epoch	438:	cost=	0.174
Epoch	439:	cost=	0.172

Epoch	440:	cost=	0.171
Epoch	441:	cost=	0.170
Epoch	442:	cost=	0.169
Epoch	443:	cost=	0.168
Epoch	444:	cost=	0.167
Epoch	445:	cost=	0.166
Epoch	446:	cost=	0.165
Epoch	447:	cost=	0.164
Epoch	448:	cost=	0.163
Epoch	449:	cost=	0.162
Epoch	450:	cost=	0.161
Epoch	451:	cost=	0.160
Epoch	452:	cost=	0.159
Epoch	453:	cost=	0.158
Epoch	454:	cost=	0.157
Epoch	455:	cost=	0.156
Epoch	456:	cost=	0.155
Epoch	457:	cost=	0.154
Epoch	458:	cost=	0.153
Epoch	459:	cost=	0.152
Epoch	460:	cost=	0.151
Epoch	461:	cost=	0.150
Epoch	462:	cost=	0.149
Epoch	463:	cost=	0.148
Epoch	464:	cost=	0.147
Epoch	465:	cost=	0.146
Epoch	466:	cost=	0.145
Epoch	467:	cost=	0.144
Epoch	468:	cost=	0.143
Epoch	469:	cost=	0.142
Epoch	470:	cost=	0.141
Epoch	471:	cost=	0.140
Epoch	472:	cost=	0.139
Epoch	473:	cost=	0.138
Epoch	474:	cost=	0.137
Epoch	475:	cost=	0.136
Epoch	476:	cost=	0.135
Epoch	477:	cost=	0.135
Epoch	478:	cost=	0.134
Epoch	479:	cost=	0.133
Epoch	480:	cost=	0.132
Epoch	481:	cost=	0.131
Epoch	482:	cost=	0.130
Epoch	483:	cost=	0.129
Epoch	484:	cost=	0.129
Epoch	485:	cost=	0.128
Epoch	486:	cost=	0.127
Epoch	487:	cost=	0.126

Epoch	488:	cost=	0.125
Epoch	489:	cost=	0.124
Epoch	490:	cost=	0.124
Epoch	491:	cost=	0.123
Epoch	492:	cost=	0.122
Epoch	493:	cost=	0.121
Epoch	494:	cost=	0.120
Epoch	495:	cost=	0.120
Epoch	496:	cost=	0.119
Epoch	497:	cost=	0.118
Epoch	498:	cost=	0.117
Epoch	499:	cost=	0.117
Epoch	500:	cost=	0.116
Epoch	501:	cost=	0.115
Epoch	502:	cost=	0.114
Epoch	503:	cost=	0.114
Epoch	504:	cost=	0.113
Epoch	505:	cost=	0.112
Epoch	506:	cost=	0.112
Epoch	507:	cost=	0.111
Epoch	508:	cost=	0.110
Epoch	509:	cost=	0.109
Epoch	510:	cost=	0.109
Epoch	511:	cost=	0.108
Epoch	512:	cost=	0.107
Epoch	513:	cost=	0.107
Epoch	514:	cost=	0.106
Epoch	515:	cost=	0.106
Epoch	516:	cost=	0.105
Epoch	517:	cost=	0.104
Epoch	518:	cost=	0.104
Epoch	519:	cost=	0.103
Epoch	520:	cost=	0.102
Epoch	521:	cost=	0.102
Epoch	522:	cost=	0.101
Epoch	523:	cost=	0.101
Epoch	524:	cost=	0.100
Epoch	525:	cost=	0.099
Epoch	526:	cost=	0.099
Epoch	527:	cost=	0.098
Epoch	528:	cost=	0.098
Epoch	529:	cost=	0.097
Epoch	530:	cost=	0.097
Epoch	531:	cost=	0.096
Epoch	532:	cost=	0.096
Epoch	533:	cost=	0.095
Epoch	534:	cost=	0.095
Epoch	535:	cost=	0.094

Epoch	536:	cost=	0.094
Epoch	537:	cost=	0.093
Epoch	538:	cost=	0.093
Epoch	539:	cost=	0.092
Epoch	540:	cost=	0.092
Epoch	541:	cost=	0.091
Epoch	542:	cost=	0.091
Epoch	543:	cost=	0.090
Epoch	544:	cost=	0.090
Epoch	545:	cost=	0.089
Epoch	546:	cost=	0.089
Epoch	547:	cost=	0.088
Epoch	548:	cost=	0.088
Epoch	549:	cost=	0.087
Epoch	550:	cost=	0.087
Epoch	551:	cost=	0.087
Epoch	552:	cost=	0.086
Epoch	553:	cost=	0.086
Epoch	554:	cost=	0.085
Epoch	555:	cost=	0.085
Epoch	556:	cost=	0.084
Epoch	557:	cost=	0.084
Epoch	558:	cost=	0.084
Epoch	559:	cost=	0.083
Epoch	560:	cost=	0.083
Epoch	561:	cost=	0.083
Epoch	562:	cost=	0.082
Epoch	563:	cost=	0.082
Epoch	564:	cost=	0.081
Epoch	565:	cost=	0.081
Epoch	566:	cost=	0.081
Epoch	567:	cost=	0.080
Epoch	568:	cost=	0.080
Epoch	569:	cost=	0.080
Epoch	570:	cost=	0.079
Epoch	571:	cost=	0.079
Epoch	572:	cost=	0.079
Epoch	573:	cost=	0.078
Epoch	574:	cost=	0.078
Epoch	575:	cost=	0.078
Epoch	576:	cost=	0.077
Epoch	577:	cost=	0.077
Epoch	578:	cost=	0.077
Epoch	579:	cost=	0.076
Epoch	580:	cost=	0.076
Epoch	581:	cost=	0.076
Epoch	582:	cost=	0.075
Epoch	583:	cost=	0.075

Epoch	584:	cost=	0.075
Epoch	585:	cost=	0.075
Epoch	586:	cost=	0.074
Epoch	587:	cost=	0.074
Epoch	588:	cost=	0.074
Epoch	589:	cost=	0.073
Epoch	590:	cost=	0.073
Epoch	591:	cost=	0.073
Epoch	592:	cost=	0.073
Epoch	593:	cost=	0.072
Epoch	594:	cost=	0.072
Epoch	595:	cost=	0.072
Epoch	596:	cost=	0.071
Epoch	597:	cost=	0.071
Epoch	598:	cost=	0.071
Epoch	599:	cost=	0.071
Epoch	600:	cost=	0.070
Epoch	601:	cost=	0.070
Epoch	602:	cost=	0.070
Epoch	603:	cost=	0.070
Epoch	604:	cost=	0.070
Epoch	605:	cost=	0.069
Epoch	606:	cost=	0.069
Epoch	607:	cost=	0.069
Epoch	608:	cost=	0.069
Epoch	609:	cost=	0.068
Epoch	610:	cost=	0.068
Epoch	611:	cost=	0.068
Epoch	612:	cost=	0.068
Epoch	613:	cost=	0.068
Epoch	614:	cost=	0.067
Epoch	615:	cost=	0.067
Epoch	616:	cost=	0.067
Epoch	617:	cost=	0.067
Epoch	618:	cost=	0.066
Epoch	619:	cost=	0.066
Epoch	620:	cost=	0.066
Epoch	621:	cost=	0.066
Epoch	622:	cost=	0.066
Epoch	623:	cost=	0.065
Epoch	624:	cost=	0.065
Epoch	625:	cost=	0.065
Epoch	626:	cost=	0.065
Epoch	627:	cost=	0.065
Epoch	628:	cost=	0.065
Epoch	629:	cost=	0.064
Epoch	630:	cost=	0.064
Epoch	631:	cost=	0.064

Epoch	632:	cost=	0.064
Epoch	633:	cost=	0.064
Epoch	634:	cost=	0.063
Epoch	635:	cost=	0.063
Epoch	636:	cost=	0.063
Epoch	637:	cost=	0.063
Epoch	638:	cost=	0.063
Epoch	639:	cost=	0.063
Epoch	640:	cost=	0.062
Epoch	641:	cost=	0.062
Epoch	642:	cost=	0.062
Epoch	643:	cost=	0.062
Epoch	644:	cost=	0.062
Epoch	645:	cost=	0.062
Epoch	646:	cost=	0.061
Epoch	647:	cost=	0.061
Epoch	648:	cost=	0.061
Epoch	649:	cost=	0.061
Epoch	650:	cost=	0.061
Epoch	651:	cost=	0.061
Epoch	652:	cost=	0.060
Epoch	653:	cost=	0.060
Epoch	654:	cost=	0.060
Epoch	655:	cost=	0.060
Epoch	656:	cost=	0.060
Epoch	657:	cost=	0.060
Epoch	658:	cost=	0.060
Epoch	659:	cost=	0.059
Epoch	660:	cost=	0.059
Epoch	661:	cost=	0.059
Epoch	662:	cost=	0.059
Epoch	663:	cost=	0.059
Epoch	664:	cost=	0.059
Epoch	665:	cost=	0.059
Epoch	666:	cost=	0.058
Epoch	667:	cost=	0.058
Epoch	668:	cost=	0.058
Epoch	669:	cost=	0.058
Epoch	670:	cost=	0.058
Epoch	671:	cost=	0.058
Epoch	672:	cost=	0.058
Epoch	673:	cost=	0.058
Epoch	674:	cost=	0.057
Epoch	675:	cost=	0.057
Epoch	676:	cost=	0.057
Epoch	677:	cost=	0.057
Epoch	678:	cost=	0.057
Epoch	679:	cost=	0.057

Epoch	680:	cost=	0.057
Epoch	681:	cost=	0.057
Epoch	682:	cost=	0.056
Epoch	683:	cost=	0.056
Epoch	684:	cost=	0.056
Epoch	685:	cost=	0.056
Epoch	686:	cost=	0.056
Epoch	687:	cost=	0.056
Epoch	688:	cost=	0.056
Epoch	689:	cost=	0.056
Epoch	690:	cost=	0.056
Epoch	691:	cost=	0.055
Epoch	692:	cost=	0.055
Epoch	693:	cost=	0.055
Epoch	694:	cost=	0.055
Epoch	695:	cost=	0.055
Epoch	696:	cost=	0.055
Epoch	697:	cost=	0.055
Epoch	698:	cost=	0.055
Epoch	699:	cost=	0.055
Epoch	700:	cost=	0.054
Epoch	701:	cost=	0.054
Epoch	702:	cost=	0.054
Epoch	703:	cost=	0.054
Epoch	704:	cost=	0.054
Epoch	705:	cost=	0.054
Epoch	706:	cost=	0.054
Epoch	707:	cost=	0.054
Epoch	708:	cost=	0.054
Epoch	709:	cost=	0.053
Epoch	710:	cost=	0.053
Epoch	711:	cost=	0.053
Epoch	712:	cost=	0.053
Epoch	713:	cost=	0.053
Epoch	714:	cost=	0.053
Epoch	715:	cost=	0.053
Epoch	716:	cost=	0.053
Epoch	717:	cost=	0.053
Epoch	718:	cost=	0.053
Epoch	719:	cost=	0.052
Epoch	720:	cost=	0.052
Epoch	721:	cost=	0.052
Epoch	722:	cost=	0.052
Epoch	723:	cost=	0.052
Epoch	724:	cost=	0.052
Epoch	725:	cost=	0.052
Epoch	726:	cost=	0.052
Epoch	727:	cost=	0.052

Epoch	728:	cost=	0.052
Epoch	729:	cost=	0.052
Epoch	730:	cost=	0.051
Epoch	731:	cost=	0.051
Epoch	732:	cost=	0.051
Epoch	733:	cost=	0.051
Epoch	734:	cost=	0.051
Epoch	735:	cost=	0.051
Epoch	736:	cost=	0.051
Epoch	737:	cost=	0.051
Epoch	738:	cost=	0.051
Epoch	739:	cost=	0.051
Epoch	740:	cost=	0.051
Epoch	741:	cost=	0.050
Epoch	742:	cost=	0.050
Epoch	743:	cost=	0.050
Epoch	744:	cost=	0.050
Epoch	745:	cost=	0.050
Epoch	746:	cost=	0.050
Epoch	747:	cost=	0.050
Epoch	748:	cost=	0.050
Epoch	749:	cost=	0.050
Epoch	750:	cost=	0.050
Epoch	751:	cost=	0.050
Epoch	752:	cost=	0.050
Epoch	753:	cost=	0.049
Epoch	754:	cost=	0.049
Epoch	755:	cost=	0.049
Epoch	756:	cost=	0.049
Epoch	757:	cost=	0.049
Epoch	758:	cost=	0.049
Epoch	759:	cost=	0.049
Epoch	760:	cost=	0.049
Epoch	761:	cost=	0.049
Epoch	762:	cost=	0.049
Epoch	763:	cost=	0.049
Epoch	764:	cost=	0.049
Epoch	765:	cost=	0.049
Epoch	766:	cost=	0.048
Epoch	767:	cost=	0.048
Epoch	768:	cost=	0.048
Epoch	769:	cost=	0.048
Epoch	770:	cost=	0.048
Epoch	771:	cost=	0.048
Epoch	772:	cost=	0.048
Epoch	773:	cost=	0.048
Epoch	774:	cost=	0.048
Epoch	775:	cost=	0.048

Epoch	776:	cost=	0.048
Epoch	777:	cost=	0.048
Epoch	778:	cost=	0.048
Epoch	779:	cost=	0.047
Epoch	780:	cost=	0.047
Epoch	781:	cost=	0.047
Epoch	782:	cost=	0.047
Epoch	783:	cost=	0.047
Epoch	784:	cost=	0.047
Epoch	785:	cost=	0.047
Epoch	786:	cost=	0.047
Epoch	787:	cost=	0.047
Epoch	788:	cost=	0.047
Epoch	789:	cost=	0.047
Epoch	790:	cost=	0.047
Epoch	791:	cost=	0.047
Epoch	792:	cost=	0.047
Epoch	793:	cost=	0.046
Epoch	794:	cost=	0.046
Epoch	795:	cost=	0.046
Epoch	796:	cost=	0.046
Epoch	797:	cost=	0.046
Epoch	798:	cost=	0.046
Epoch	799:	cost=	0.046
Epoch	800:	cost=	0.046
Epoch	801:	cost=	0.046
Epoch	802:	cost=	0.046
Epoch	803:	cost=	0.046
Epoch	804:	cost=	0.046
Epoch	805:	cost=	0.046
Epoch	806:	cost=	0.046
Epoch	807:	cost=	0.046
Epoch	808:	cost=	0.045
Epoch	809:	cost=	0.045
Epoch	810:	cost=	0.045
Epoch	811:	cost=	0.045
Epoch	812:	cost=	0.045
Epoch	813:	cost=	0.045
Epoch	814:	cost=	0.045
Epoch	815:	cost=	0.045
Epoch	816:	cost=	0.045
Epoch	817:	cost=	0.045
Epoch	818:	cost=	0.045
Epoch	819:	cost=	0.045
Epoch	820:	cost=	0.045
Epoch	821:	cost=	0.045
Epoch	822:	cost=	0.045
Epoch	823:	cost=	0.045

Epoch	824:	cost=	0.044
Epoch	825:	cost=	0.044
Epoch	826:	cost=	0.044
Epoch	827:	cost=	0.044
Epoch	828:	cost=	0.044
Epoch	829:	cost=	0.044
Epoch	830:	cost=	0.044
Epoch	831:	cost=	0.044
Epoch	832:	cost=	0.044
Epoch	833:	cost=	0.044
Epoch	834:	cost=	0.044
Epoch	835:	cost=	0.044
Epoch	836:	cost=	0.044
Epoch	837:	cost=	0.044
Epoch	838:	cost=	0.044
Epoch	839:	cost=	0.044
Epoch	840:	cost=	0.044
Epoch	841:	cost=	0.044
Epoch	842:	cost=	0.043
Epoch	843:	cost=	0.043
Epoch	844:	cost=	0.043
Epoch	845:	cost=	0.043
Epoch	846:	cost=	0.043
Epoch	847:	cost=	0.043
Epoch	848:	cost=	0.043
Epoch	849:	cost=	0.043
Epoch	850:	cost=	0.043
Epoch	851:	cost=	0.043
Epoch	852:	cost=	0.043
Epoch	853:	cost=	0.043
Epoch	854:	cost=	0.043
Epoch	855:	cost=	0.043
Epoch	856:	cost=	0.043
Epoch	857:	cost=	0.043
Epoch	858:	cost=	0.043
Epoch	859:	cost=	0.043
Epoch	860:	cost=	0.042
Epoch	861:	cost=	0.042
Epoch	862:	cost=	0.042
Epoch	863:	cost=	0.042
Epoch	864:	cost=	0.042
Epoch	865:	cost=	0.042
Epoch	866:	cost=	0.042
Epoch	867:	cost=	0.042
Epoch	868:	cost=	0.042
Epoch	869:	cost=	0.042
Epoch	870:	cost=	0.042
Epoch	871:	cost=	0.042

Epoch	872:	cost=	0.042
Epoch	873:	cost=	0.042
Epoch	874:	cost=	0.042
Epoch	875:	cost=	0.042
Epoch	876:	cost=	0.042
Epoch	877:	cost=	0.042
Epoch	878:	cost=	0.042
Epoch	879:	cost=	0.041
Epoch	880:	cost=	0.041
Epoch	881:	cost=	0.041
Epoch	882:	cost=	0.041
Epoch	883:	cost=	0.041
Epoch	884:	cost=	0.041
Epoch	885:	cost=	0.041
Epoch	886:	cost=	0.041
Epoch	887:	cost=	0.041
Epoch	888:	cost=	0.041
Epoch	889:	cost=	0.041
Epoch	890:	cost=	0.041
Epoch	891:	cost=	0.041
Epoch	892:	cost=	0.041
Epoch	893:	cost=	0.041
Epoch	894:	cost=	0.041
Epoch	895:	cost=	0.041
Epoch	896:	cost=	0.041
Epoch	897:	cost=	0.041
Epoch	898:	cost=	0.041
Epoch	899:	cost=	0.041
Epoch	900:	cost=	0.040
Epoch	901:	cost=	0.040
Epoch	902:	cost=	0.040
Epoch	903:	cost=	0.040
Epoch	904:	cost=	0.040
Epoch	905:	cost=	0.040
Epoch	906:	cost=	0.040
Epoch	907:	cost=	0.040
Epoch	908:	cost=	0.040
Epoch	909:	cost=	0.040
Epoch	910:	cost=	0.040
Epoch	911:	cost=	0.040
Epoch	912:	cost=	0.040
Epoch	913:	cost=	0.040
Epoch	914:	cost=	0.040
Epoch	915:	cost=	0.040
Epoch	916:	cost=	0.040
Epoch	917:	cost=	0.040
Epoch	918:	cost=	0.040
Epoch	919:	cost=	0.040

Epoch	920:	cost=	0.040
Epoch	921:	cost=	0.040
Epoch	922:	cost=	0.040
Epoch	923:	cost=	0.039
Epoch	924:	cost=	0.039
Epoch	925:	cost=	0.039
Epoch	926:	cost=	0.039
Epoch	927:	cost=	0.039
Epoch	928:	cost=	0.039
Epoch	929:	cost=	0.039
Epoch	930:	cost=	0.039
Epoch	931:	cost=	0.039
Epoch	932:	cost=	0.039
Epoch	933:	cost=	0.039
Epoch	934:	cost=	0.039
Epoch	935:	cost=	0.039
Epoch	936:	cost=	0.039
Epoch	937:	cost=	0.039
Epoch	938:	cost=	0.039
Epoch	939:	cost=	0.039
Epoch	940:	cost=	0.039
Epoch	941:	cost=	0.039
Epoch	942:	cost=	0.039
Epoch	943:	cost=	0.039
Epoch	944:	cost=	0.039
Epoch	945:	cost=	0.039
Epoch	946:	cost=	0.039
Epoch	947:	cost=	0.038
Epoch	948:	cost=	0.038
Epoch	949:	cost=	0.038
Epoch	950:	cost=	0.038
Epoch	951:	cost=	0.038
Epoch	952:	cost=	0.038
Epoch	953:	cost=	0.038
Epoch	954:	cost=	0.038
Epoch	955:	cost=	0.038
Epoch	956:	cost=	0.038
Epoch	957:	cost=	0.038
Epoch	958:	cost=	0.038
Epoch	959:	cost=	0.038
Epoch	960:	cost=	0.038
Epoch	961:	cost=	0.038
Epoch	962:	cost=	0.038
Epoch	963:	cost=	0.038
Epoch	964:	cost=	0.038
Epoch	965:	cost=	0.038
Epoch	966:	cost=	0.038
Epoch	967:	cost=	0.038

Epoch	968:	cost=	0.038
Epoch	969:	cost=	0.038
Epoch	970:	cost=	0.038
Epoch	971:	cost=	0.038
Epoch	972:	cost=	0.038
Epoch	973:	cost=	0.038
Epoch	974:	cost=	0.038
Epoch	975:	cost=	0.037
Epoch	976:	cost=	0.037
Epoch	977:	cost=	0.037
Epoch	978:	cost=	0.037
Epoch	979:	cost=	0.037
Epoch	980:	cost=	0.037
Epoch	981:	cost=	0.037
Epoch	982:	cost=	0.037
Epoch	983:	cost=	0.037
Epoch	984:	cost=	0.037
Epoch	985:	cost=	0.037
Epoch	986:	cost=	0.037
Epoch	987:	cost=	0.037
Epoch	988:	cost=	0.037
Epoch	989:	cost=	0.037
Epoch	990:	cost=	0.037
Epoch	991:	cost=	0.037
Epoch	992:	cost=	0.037
Epoch	993:	cost=	0.037
Epoch	994:	cost=	0.037
Epoch	995:	cost=	0.037
Epoch	996:	cost=	0.037
Epoch	997:	cost=	0.037
Epoch	998:	cost=	0.037
Epoch	999:	cost=	0.037
Epoch	1000:	cost=	0.037
Epoch	1001:	cost=	0.036
Epoch	1002:	cost=	0.036
Epoch	1003:	cost=	0.036
Epoch	1004:	cost=	0.036
Epoch	1005:	cost=	0.036
Epoch	1006:	cost=	0.036
Epoch	1007:	cost=	0.036
Epoch	1008:	cost=	0.036
Epoch	1009:	cost=	0.036
Epoch	1010:	cost=	0.036
Epoch	1011:	cost=	0.036
Epoch	1012:	cost=	0.036
Epoch	1013:	cost=	0.036
Epoch	1014:	cost=	0.036
Epoch	1015:	cost=	0.036

Epoch	1016:	cost=	0.036
Epoch	1017:	cost=	0.036
Epoch	1018:	cost=	0.036
Epoch	1019:	cost=	0.036
Epoch	1020:	cost=	0.036
Epoch	1021:	cost=	0.036
Epoch	1022:	cost=	0.036
Epoch	1023:	cost=	0.036
Epoch	1024:	cost=	0.036
Epoch	1025:	cost=	0.036
Epoch	1026:	cost=	0.036
Epoch	1027:	cost=	0.036
Epoch	1028:	cost=	0.036
Epoch	1029:	cost=	0.036
Epoch	1030:	cost=	0.036
Epoch	1031:	cost=	0.036
Epoch	1032:	cost=	0.035
Epoch	1033:	cost=	0.035
Epoch	1034:	cost=	0.035
Epoch	1035:	cost=	0.035
Epoch	1036:	cost=	0.035
Epoch	1037:	cost=	0.035
Epoch	1038:	cost=	0.035
Epoch	1039:	cost=	0.035
Epoch	1040:	cost=	0.035
Epoch	1041:	cost=	0.035
Epoch	1042:	cost=	0.035
Epoch	1043:	cost=	0.035
Epoch	1044:	cost=	0.035
Epoch	1045:	cost=	0.035
Epoch	1046:	cost=	0.035
Epoch	1047:	cost=	0.035
Epoch	1048:	cost=	0.035
Epoch	1049:	cost=	0.035
Epoch	1050:	cost=	0.035
Epoch	1051:	cost=	0.035
Epoch	1052:	cost=	0.035
Epoch	1053:	cost=	0.035
Epoch	1054:	cost=	0.035
Epoch	1055:	cost=	0.035
Epoch	1056:	cost=	0.035
Epoch	1057:	cost=	0.035
Epoch	1058:	cost=	0.035
Epoch	1059:	cost=	0.035
Epoch	1060:	cost=	0.035
Epoch	1061:	cost=	0.035
Epoch	1062:	cost=	0.035
Epoch	1063:	cost=	0.035

Epoch	1064:	cost=	0.035
Epoch	1065:	cost=	0.034
Epoch	1066:	cost=	0.034
Epoch	1067:	cost=	0.034
Epoch	1068:	cost=	0.034
Epoch	1069:	cost=	0.034
Epoch	1070:	cost=	0.034
Epoch	1071:	cost=	0.034
Epoch	1072:	cost=	0.034
Epoch	1073:	cost=	0.034
Epoch	1074:	cost=	0.034
Epoch	1075:	cost=	0.034
Epoch	1076:	cost=	0.034
Epoch	1077:	cost=	0.034
Epoch	1078:	cost=	0.034
Epoch	1079:	cost=	0.034
Epoch	1080:	cost=	0.034
Epoch	1081:	cost=	0.034
Epoch	1082:	cost=	0.034
Epoch	1083:	cost=	0.034
Epoch	1084:	cost=	0.034
Epoch	1085:	cost=	0.034
Epoch	1086:	cost=	0.034
Epoch	1087:	cost=	0.034
Epoch	1088:	cost=	0.034
Epoch	1089:	cost=	0.034
Epoch	1090:	cost=	0.034
Epoch	1091:	cost=	0.034
Epoch	1092:	cost=	0.034
Epoch	1093:	cost=	0.034
Epoch	1094:	cost=	0.034
Epoch	1095:	cost=	0.034
Epoch	1096:	cost=	0.034
Epoch	1097:	cost=	0.034
Epoch	1098:	cost=	0.034
Epoch	1099:	cost=	0.034
Epoch	1100:	cost=	0.034
Epoch	1101:	cost=	0.033
Epoch	1102:	cost=	0.033
Epoch	1103:	cost=	0.033
Epoch	1104:	cost=	0.033
Epoch	1105:	cost=	0.033
Epoch	1106:	cost=	0.033
Epoch	1107:	cost=	0.033
Epoch	1108:	cost=	0.033
Epoch	1109:	cost=	0.033
Epoch	1110:	cost=	0.033
Epoch	1111:	cost=	0.033

Epoch	1112:	cost=	0.033
Epoch	1113:	cost=	0.033
Epoch	1114:	cost=	0.033
Epoch	1115:	cost=	0.034
Epoch	1116:	cost=	0.034
Epoch	1117:	cost=	0.035
Epoch	1118:	cost=	0.034
Epoch	1119:	cost=	0.033
Epoch	1120:	cost=	0.034
Epoch	1121:	cost=	0.034
Epoch	1122:	cost=	0.033
Epoch	1123:	cost=	0.033
Epoch	1124:	cost=	0.033
Epoch	1125:	cost=	0.033
Epoch	1126:	cost=	0.033
Epoch	1127:	cost=	0.033
Epoch	1128:	cost=	0.033
Epoch	1129:	cost=	0.033
Epoch	1130:	cost=	0.033
Epoch	1131:	cost=	0.033
Epoch	1132:	cost=	0.033
Epoch	1133:	cost=	0.033
Epoch	1134:	cost=	0.033
Epoch	1135:	cost=	0.033
Epoch	1136:	cost=	0.033
Epoch	1137:	cost=	0.033
Epoch	1138:	cost=	0.033
Epoch	1139:	cost=	0.032
Epoch	1140:	cost=	0.033
Epoch	1141:	cost=	0.032
Epoch	1142:	cost=	0.032
Epoch	1143:	cost=	0.032
Epoch	1144:	cost=	0.032
Epoch	1145:	cost=	0.032
Epoch	1146:	cost=	0.032
Epoch	1147:	cost=	0.032
Epoch	1148:	cost=	0.032
Epoch	1149:	cost=	0.032
Epoch	1150:	cost=	0.032
Epoch	1151:	cost=	0.032
Epoch	1152:	cost=	0.032
Epoch	1153:	cost=	0.032
Epoch	1154:	cost=	0.032
Epoch	1155:	cost=	0.032
Epoch	1156:	cost=	0.032
Epoch	1157:	cost=	0.032
Epoch	1158:	cost=	0.032
Epoch	1159:	cost=	0.032

Epoch	1160:	cost=	0.032
Epoch	1161:	cost=	0.032
Epoch	1162:	cost=	0.032
Epoch	1163:	cost=	0.032
Epoch	1164:	cost=	0.032
Epoch	1165:	cost=	0.032
Epoch	1166:	cost=	0.032
Epoch	1167:	cost=	0.032
Epoch	1168:	cost=	0.032
Epoch	1169:	cost=	0.032
Epoch	1170:	cost=	0.032
Epoch	1171:	cost=	0.032
Epoch	1172:	cost=	0.032
Epoch	1173:	cost=	0.032
Epoch	1174:	cost=	0.032
Epoch	1175:	cost=	0.032
Epoch	1176:	cost=	0.032
Epoch	1177:	cost=	0.032
Epoch	1178:	cost=	0.031
Epoch	1179:	cost=	0.031
Epoch	1180:	cost=	0.031
Epoch	1181:	cost=	0.031
Epoch	1182:	cost=	0.031
Epoch	1183:	cost=	0.031
Epoch	1184:	cost=	0.031
Epoch	1185:	cost=	0.031
Epoch	1186:	cost=	0.031
Epoch	1187:	cost=	0.031
Epoch	1188:	cost=	0.031
Epoch	1189:	cost=	0.031
Epoch	1190:	cost=	0.031
Epoch	1191:	cost=	0.031
Epoch	1192:	cost=	0.031
Epoch	1193:	cost=	0.031
Epoch	1194:	cost=	0.031
Epoch	1195:	cost=	0.031
Epoch	1196:	cost=	0.031
Epoch	1197:	cost=	0.031
Epoch	1198:	cost=	0.031
Epoch	1199:	cost=	0.031
Epoch	1200:	cost=	0.031
Epoch	1201:	cost=	0.031
Epoch	1202:	cost=	0.031
Epoch	1203:	cost=	0.031
Epoch	1204:	cost=	0.031
Epoch	1205:	cost=	0.031
Epoch	1206:	cost=	0.031
Epoch	1207:	cost=	0.030

Epoch	1208:	cost=	0.030
Epoch	1209:	cost=	0.030
Epoch	1210:	cost=	0.030
Epoch	1211:	cost=	0.030
Epoch	1212:	cost=	0.030
Epoch	1213:	cost=	0.030
Epoch	1214:	cost=	0.030
Epoch	1215:	cost=	0.030
Epoch	1216:	cost=	0.029
Epoch	1217:	cost=	0.029
Epoch	1218:	cost=	0.029
Epoch	1219:	cost=	0.029
Epoch	1220:	cost=	0.029
Epoch	1221:	cost=	0.029
Epoch	1222:	cost=	0.029
Epoch	1223:	cost=	0.028
Epoch	1224:	cost=	0.028
Epoch	1225:	cost=	0.028
Epoch	1226:	cost=	0.028
Epoch	1227:	cost=	0.028
Epoch	1228:	cost=	0.028
Epoch	1229:	cost=	0.028
Epoch	1230:	cost=	0.028
Epoch	1231:	cost=	0.028
Epoch	1232:	cost=	0.028
Epoch	1233:	cost=	0.028
Epoch	1234:	cost=	0.028
Epoch	1235:	cost=	0.028
Epoch	1236:	cost=	0.028
Epoch	1237:	cost=	0.028
Epoch	1238:	cost=	0.028
Epoch	1239:	cost=	0.028
Epoch	1240:	cost=	0.028
Epoch	1241:	cost=	0.027
Epoch	1242:	cost=	0.027
Epoch	1243:	cost=	0.027
Epoch	1244:	cost=	0.027
Epoch	1245:	cost=	0.027
Epoch	1246:	cost=	0.027
Epoch	1247:	cost=	0.027
Epoch	1248:	cost=	0.027
Epoch	1249:	cost=	0.027
Epoch	1250:	cost=	0.027
Epoch	1251:	cost=	0.027
Epoch	1252:	cost=	0.027
Epoch	1253:	cost=	0.027
Epoch	1254:	cost=	0.027
Epoch	1255:	cost=	0.027

Epoch	1256:	cost=	0.027
Epoch	1257:	cost=	0.027
Epoch	1258:	cost=	0.027
Epoch	1259:	cost=	0.027
Epoch	1260:	cost=	0.027
Epoch	1261:	cost=	0.027
Epoch	1262:	cost=	0.026
Epoch	1263:	cost=	0.026
Epoch	1264:	cost=	0.026
Epoch	1265:	cost=	0.027
Epoch	1266:	cost=	0.027
Epoch	1267:	cost=	0.027
Epoch	1268:	cost=	0.028
Epoch	1269:	cost=	0.027
Epoch	1270:	cost=	0.026
Epoch	1271:	cost=	0.027
Epoch	1272:	cost=	0.027
Epoch	1273:	cost=	0.026
Epoch	1274:	cost=	0.026
Epoch	1275:	cost=	0.027
Epoch	1276:	cost=	0.026
Epoch	1277:	cost=	0.026
Epoch	1278:	cost=	0.026
Epoch	1279:	cost=	0.026
Epoch	1280:	cost=	0.026
Epoch	1281:	cost=	0.026
Epoch	1282:	cost=	0.026
Epoch	1283:	cost=	0.026
Epoch	1284:	cost=	0.026
Epoch	1285:	cost=	0.026
Epoch	1286:	cost=	0.026
Epoch	1287:	cost=	0.026
Epoch	1288:	cost=	0.026
Epoch	1289:	cost=	0.025
Epoch	1290:	cost=	0.025
Epoch	1291:	cost=	0.026
Epoch	1292:	cost=	0.025
Epoch	1293:	cost=	0.025
Epoch	1294:	cost=	0.025
Epoch	1295:	cost=	0.025
Epoch	1296:	cost=	0.025
Epoch	1297:	cost=	0.025
Epoch	1298:	cost=	0.025
Epoch	1299:	cost=	0.025
Epoch	1300:	cost=	0.025
Epoch	1301:	cost=	0.025
Epoch	1302:	cost=	0.025
Epoch	1303:	cost=	0.025

Epoch	1304:	cost=	0.025
Epoch	1305:	cost=	0.025
Epoch	1306:	cost=	0.025
Epoch	1307:	cost=	0.025
Epoch	1308:	cost=	0.025
Epoch	1309:	cost=	0.025
Epoch	1310:	cost=	0.025
Epoch	1311:	cost=	0.025
Epoch	1312:	cost=	0.025
Epoch	1313:	cost=	0.025
Epoch	1314:	cost=	0.025
Epoch	1315:	cost=	0.025
Epoch	1316:	cost=	0.025
Epoch	1317:	cost=	0.024
Epoch	1318:	cost=	0.024
Epoch	1319:	cost=	0.024
Epoch	1320:	cost=	0.024
Epoch	1321:	cost=	0.024
Epoch	1322:	cost=	0.024
Epoch	1323:	cost=	0.024
Epoch	1324:	cost=	0.024
Epoch	1325:	cost=	0.024
Epoch	1326:	cost=	0.024
Epoch	1327:	cost=	0.024
Epoch	1328:	cost=	0.024
Epoch	1329:	cost=	0.024
Epoch	1330:	cost=	0.024
Epoch	1331:	cost=	0.024
Epoch	1332:	cost=	0.024
Epoch	1333:	cost=	0.024
Epoch	1334:	cost=	0.024
Epoch	1335:	cost=	0.024
Epoch	1336:	cost=	0.024
Epoch	1337:	cost=	0.024
Epoch	1338:	cost=	0.024
Epoch	1339:	cost=	0.024
Epoch	1340:	cost=	0.024
Epoch	1341:	cost=	0.024
Epoch	1342:	cost=	0.024
Epoch	1343:	cost=	0.024
Epoch	1344:	cost=	0.024
Epoch	1345:	cost=	0.023
Epoch	1346:	cost=	0.023
Epoch	1347:	cost=	0.023
Epoch	1348:	cost=	0.023
Epoch	1349:	cost=	0.023
Epoch	1350:	cost=	0.023
Epoch	1351:	cost=	0.023

Epoch	1352:	cost=	0.023
Epoch	1353:	cost=	0.023
Epoch	1354:	cost=	0.023
Epoch	1355:	cost=	0.023
Epoch	1356:	cost=	0.023
Epoch	1357:	cost=	0.023
Epoch	1358:	cost=	0.023
Epoch	1359:	cost=	0.023
Epoch	1360:	cost=	0.023
Epoch	1361:	cost=	0.023
Epoch	1362:	cost=	0.023
Epoch	1363:	cost=	0.023
Epoch	1364:	cost=	0.023
Epoch	1365:	cost=	0.023
Epoch	1366:	cost=	0.023
Epoch	1367:	cost=	0.023
Epoch	1368:	cost=	0.023
Epoch	1369:	cost=	0.023
Epoch	1370:	cost=	0.023
Epoch	1371:	cost=	0.023
Epoch	1372:	cost=	0.022
Epoch	1373:	cost=	0.022
Epoch	1374:	cost=	0.022
Epoch	1375:	cost=	0.022
Epoch	1376:	cost=	0.022
Epoch	1377:	cost=	0.022
Epoch	1378:	cost=	0.022
Epoch	1379:	cost=	0.022
Epoch	1380:	cost=	0.022
Epoch	1381:	cost=	0.022
Epoch	1382:	cost=	0.022
Epoch	1383:	cost=	0.022
Epoch	1384:	cost=	0.022
Epoch	1385:	cost=	0.022
Epoch	1386:	cost=	0.022
Epoch	1387:	cost=	0.022
Epoch	1388:	cost=	0.022
Epoch	1389:	cost=	0.022
Epoch	1390:	cost=	0.022
Epoch	1391:	cost=	0.022
Epoch	1392:	cost=	0.022
Epoch	1393:	cost=	0.022
Epoch	1394:	cost=	0.022
Epoch	1395:	cost=	0.022
Epoch	1396:	cost=	0.022
Epoch	1397:	cost=	0.022
Epoch	1398:	cost=	0.022
Epoch	1399:	cost=	0.022

Epoch	1400:	cost=	0.021
Epoch	1401:	cost=	0.021
Epoch	1402:	cost=	0.021
Epoch	1403:	cost=	0.021
Epoch	1404:	cost=	0.021
Epoch	1405:	cost=	0.021
Epoch	1406:	cost=	0.021
Epoch	1407:	cost=	0.021
Epoch	1408:	cost=	0.021
Epoch	1409:	cost=	0.021
Epoch	1410:	cost=	0.021
Epoch	1411:	cost=	0.021
Epoch	1412:	cost=	0.021
Epoch	1413:	cost=	0.021
Epoch	1414:	cost=	0.021
Epoch	1415:	cost=	0.021
Epoch	1416:	cost=	0.021
Epoch	1417:	cost=	0.021
Epoch	1418:	cost=	0.021
Epoch	1419:	cost=	0.021
Epoch	1420:	cost=	0.021
Epoch	1421:	cost=	0.021
Epoch	1422:	cost=	0.021
Epoch	1423:	cost=	0.021
Epoch	1424:	cost=	0.021
Epoch	1425:	cost=	0.021
Epoch	1426:	cost=	0.021
Epoch	1427:	cost=	0.021
Epoch	1428:	cost=	0.020
Epoch	1429:	cost=	0.020
Epoch	1430:	cost=	0.020
Epoch	1431:	cost=	0.020
Epoch	1432:	cost=	0.020
Epoch	1433:	cost=	0.020
Epoch	1434:	cost=	0.020
Epoch	1435:	cost=	0.020
Epoch	1436:	cost=	0.020
Epoch	1437:	cost=	0.020
Epoch	1438:	cost=	0.020
Epoch	1439:	cost=	0.020
Epoch	1440:	cost=	0.020
Epoch	1441:	cost=	0.020
Epoch	1442:	cost=	0.020
Epoch	1443:	cost=	0.020
Epoch	1444:	cost=	0.020
Epoch	1445:	cost=	0.020
Epoch	1446:	cost=	0.020
Epoch	1447:	cost=	0.020

Epoch	1448:	cost=	0.020
Epoch	1449:	cost=	0.020
Epoch	1450:	cost=	0.020
Epoch	1451:	cost=	0.020
Epoch	1452:	cost=	0.020
Epoch	1453:	cost=	0.020
Epoch	1454:	cost=	0.020
Epoch	1455:	cost=	0.019
Epoch	1456:	cost=	0.019
Epoch	1457:	cost=	0.019
Epoch	1458:	cost=	0.019
Epoch	1459:	cost=	0.019
Epoch	1460:	cost=	0.019
Epoch	1461:	cost=	0.019
Epoch	1462:	cost=	0.019
Epoch	1463:	cost=	0.019
Epoch	1464:	cost=	0.020
Epoch	1465:	cost=	0.020
Epoch	1466:	cost=	0.020
Epoch	1467:	cost=	0.020
Epoch	1468:	cost=	0.019
Epoch	1469:	cost=	0.019
Epoch	1470:	cost=	0.019
Epoch	1471:	cost=	0.020
Epoch	1472:	cost=	0.019
Epoch	1473:	cost=	0.019
Epoch	1474:	cost=	0.019
Epoch	1475:	cost=	0.019
Epoch	1476:	cost=	0.019
Epoch	1477:	cost=	0.019
Epoch	1478:	cost=	0.019
Epoch	1479:	cost=	0.019
Epoch	1480:	cost=	0.019
Epoch	1481:	cost=	0.019
Epoch	1482:	cost=	0.019
Epoch	1483:	cost=	0.019
Epoch	1484:	cost=	0.019
Epoch	1485:	cost=	0.019
Epoch	1486:	cost=	0.019
Epoch	1487:	cost=	0.019
Epoch	1488:	cost=	0.018
Epoch	1489:	cost=	0.018
Epoch	1490:	cost=	0.018
Epoch	1491:	cost=	0.018
Epoch	1492:	cost=	0.018
Epoch	1493:	cost=	0.018
Epoch	1494:	cost=	0.018
Epoch	1495:	cost=	0.018

Epoch	1496:	cost=	0.018
Epoch	1497:	cost=	0.018
Epoch	1498:	cost=	0.018
Epoch	1499:	cost=	0.018
Epoch	1500:	cost=	0.018
Epoch	1501:	cost=	0.018
Epoch	1502:	cost=	0.018
Epoch	1503:	cost=	0.018
Epoch	1504:	cost=	0.018
Epoch	1505:	cost=	0.018
Epoch	1506:	cost=	0.018
Epoch	1507:	cost=	0.018
Epoch	1508:	cost=	0.018
Epoch	1509:	cost=	0.018
Epoch	1510:	cost=	0.018
Epoch	1511:	cost=	0.018
Epoch	1512:	cost=	0.018
Epoch	1513:	cost=	0.018
Epoch	1514:	cost=	0.018
Epoch	1515:	cost=	0.018
Epoch	1516:	cost=	0.018
Epoch	1517:	cost=	0.018
Epoch	1518:	cost=	0.018
Epoch	1519:	cost=	0.018
Epoch	1520:	cost=	0.018
Epoch	1521:	cost=	0.018
Epoch	1522:	cost=	0.017
Epoch	1523:	cost=	0.017
Epoch	1524:	cost=	0.017
Epoch	1525:	cost=	0.017
Epoch	1526:	cost=	0.017
Epoch	1527:	cost=	0.017
Epoch	1528:	cost=	0.017
Epoch	1529:	cost=	0.017
Epoch	1530:	cost=	0.017
Epoch	1531:	cost=	0.017
Epoch	1532:	cost=	0.017
Epoch	1533:	cost=	0.017
Epoch	1534:	cost=	0.017
Epoch	1535:	cost=	0.017
Epoch	1536:	cost=	0.017
Epoch	1537:	cost=	0.017
Epoch	1538:	cost=	0.017
Epoch	1539:	cost=	0.017
Epoch	1540:	cost=	0.017
Epoch	1541:	cost=	0.017
Epoch	1542:	cost=	0.017
Epoch	1543:	cost=	0.017

Epoch	1544:	cost=	0.017
Epoch	1545:	cost=	0.017
Epoch	1546:	cost=	0.017
Epoch	1547:	cost=	0.017
Epoch	1548:	cost=	0.017
Epoch	1549:	cost=	0.017
Epoch	1550:	cost=	0.017
Epoch	1551:	cost=	0.017
Epoch	1552:	cost=	0.017
Epoch	1553:	cost=	0.017
Epoch	1554:	cost=	0.017
Epoch	1555:	cost=	0.017
Epoch	1556:	cost=	0.017
Epoch	1557:	cost=	0.017
Epoch	1558:	cost=	0.017
Epoch	1559:	cost=	0.016
Epoch	1560:	cost=	0.016
Epoch	1561:	cost=	0.016
Epoch	1562:	cost=	0.016
Epoch	1563:	cost=	0.016
Epoch	1564:	cost=	0.016
Epoch	1565:	cost=	0.016
Epoch	1566:	cost=	0.016
Epoch	1567:	cost=	0.016
Epoch	1568:	cost=	0.016
Epoch	1569:	cost=	0.016
Epoch	1570:	cost=	0.016
Epoch	1571:	cost=	0.016
Epoch	1572:	cost=	0.016
Epoch	1573:	cost=	0.016
Epoch	1574:	cost=	0.016
Epoch	1575:	cost=	0.016
Epoch	1576:	cost=	0.016
Epoch	1577:	cost=	0.016
Epoch	1578:	cost=	0.016
Epoch	1579:	cost=	0.016
Epoch	1580:	cost=	0.016
Epoch	1581:	cost=	0.016
Epoch	1582:	cost=	0.016
Epoch	1583:	cost=	0.016
Epoch	1584:	cost=	0.016
Epoch	1585:	cost=	0.016
Epoch	1586:	cost=	0.016
Epoch	1587:	cost=	0.016
Epoch	1588:	cost=	0.016
Epoch	1589:	cost=	0.016
Epoch	1590:	cost=	0.016
Epoch	1591:	cost=	0.016

Epoch	1592:	cost=	0.016
Epoch	1593:	cost=	0.016
Epoch	1594:	cost=	0.016
Epoch	1595:	cost=	0.016
Epoch	1596:	cost=	0.016
Epoch	1597:	cost=	0.016
Epoch	1598:	cost=	0.015
Epoch	1599:	cost=	0.015
Epoch	1600:	cost=	0.015
Epoch	1601:	cost=	0.015
Epoch	1602:	cost=	0.015
Epoch	1603:	cost=	0.015
Epoch	1604:	cost=	0.015
Epoch	1605:	cost=	0.015
Epoch	1606:	cost=	0.015
Epoch	1607:	cost=	0.015
Epoch	1608:	cost=	0.015
Epoch	1609:	cost=	0.015
Epoch	1610:	cost=	0.015
Epoch	1611:	cost=	0.015
Epoch	1612:	cost=	0.015
Epoch	1613:	cost=	0.015
Epoch	1614:	cost=	0.016
Epoch	1615:	cost=	0.016
Epoch	1616:	cost=	0.016
Epoch	1617:	cost=	0.016
Epoch	1618:	cost=	0.015
Epoch	1619:	cost=	0.015
Epoch	1620:	cost=	0.015
Epoch	1621:	cost=	0.015
Epoch	1622:	cost=	0.015
Epoch	1623:	cost=	0.015
Epoch	1624:	cost=	0.015
Epoch	1625:	cost=	0.015
Epoch	1626:	cost=	0.015
Epoch	1627:	cost=	0.015
Epoch	1628:	cost=	0.015
Epoch	1629:	cost=	0.015
Epoch	1630:	cost=	0.015
Epoch	1631:	cost=	0.015
Epoch	1632:	cost=	0.015
Epoch	1633:	cost=	0.015
Epoch	1634:	cost=	0.015
Epoch	1635:	cost=	0.015
Epoch	1636:	cost=	0.015
Epoch	1637:	cost=	0.015
Epoch	1638:	cost=	0.015
Epoch	1639:	cost=	0.015

Epoch	1640:	cost=	0.015
Epoch	1641:	cost=	0.015
Epoch	1642:	cost=	0.015
Epoch	1643:	cost=	0.014
Epoch	1644:	cost=	0.014
Epoch	1645:	cost=	0.014
Epoch	1646:	cost=	0.014
Epoch	1647:	cost=	0.014
Epoch	1648:	cost=	0.014
Epoch	1649:	cost=	0.014
Epoch	1650:	cost=	0.014
Epoch	1651:	cost=	0.014
Epoch	1652:	cost=	0.014
Epoch	1653:	cost=	0.014
Epoch	1654:	cost=	0.014
Epoch	1655:	cost=	0.014
Epoch	1656:	cost=	0.014
Epoch	1657:	cost=	0.014
Epoch	1658:	cost=	0.014
Epoch	1659:	cost=	0.014
Epoch	1660:	cost=	0.014
Epoch	1661:	cost=	0.014
Epoch	1662:	cost=	0.014
Epoch	1663:	cost=	0.014
Epoch	1664:	cost=	0.014
Epoch	1665:	cost=	0.014
Epoch	1666:	cost=	0.014
Epoch	1667:	cost=	0.014
Epoch	1668:	cost=	0.014
Epoch	1669:	cost=	0.014
Epoch	1670:	cost=	0.014
Epoch	1671:	cost=	0.014
Epoch	1672:	cost=	0.014
Epoch	1673:	cost=	0.014
Epoch	1674:	cost=	0.014
Epoch	1675:	cost=	0.014
Epoch	1676:	cost=	0.014
Epoch	1677:	cost=	0.014
Epoch	1678:	cost=	0.014
Epoch	1679:	cost=	0.014
Epoch	1680:	cost=	0.014
Epoch	1681:	cost=	0.014
Epoch	1682:	cost=	0.014
Epoch	1683:	cost=	0.014
Epoch	1684:	cost=	0.014
Epoch	1685:	cost=	0.014
Epoch	1686:	cost=	0.014
Epoch	1687:	cost=	0.014

Epoch	1688:	cost=	0.014
Epoch	1689:	cost=	0.014
Epoch	1690:	cost=	0.014
Epoch	1691:	cost=	0.014
Epoch	1692:	cost=	0.014
Epoch	1693:	cost=	0.014
Epoch	1694:	cost=	0.014
Epoch	1695:	cost=	0.014
Epoch	1696:	cost=	0.013
Epoch	1697:	cost=	0.013
Epoch	1698:	cost=	0.013
Epoch	1699:	cost=	0.013
Epoch	1700:	cost=	0.013
Epoch	1701:	cost=	0.013
Epoch	1702:	cost=	0.013
Epoch	1703:	cost=	0.013
Epoch	1704:	cost=	0.013
Epoch	1705:	cost=	0.013
Epoch	1706:	cost=	0.013
Epoch	1707:	cost=	0.013
Epoch	1708:	cost=	0.013
Epoch	1709:	cost=	0.013
Epoch	1710:	cost=	0.013
Epoch	1711:	cost=	0.013
Epoch	1712:	cost=	0.013
Epoch	1713:	cost=	0.013
Epoch	1714:	cost=	0.013
Epoch	1715:	cost=	0.013
Epoch	1716:	cost=	0.013
Epoch	1717:	cost=	0.013
Epoch	1718:	cost=	0.013
Epoch	1719:	cost=	0.013
Epoch	1720:	cost=	0.013
Epoch	1721:	cost=	0.013
Epoch	1722:	cost=	0.013
Epoch	1723:	cost=	0.013
Epoch	1724:	cost=	0.013
Epoch	1725:	cost=	0.013
Epoch	1726:	cost=	0.013
Epoch	1727:	cost=	0.013
Epoch	1728:	cost=	0.013
Epoch	1729:	cost=	0.013
Epoch	1730:	cost=	0.013
Epoch	1731:	cost=	0.013
Epoch	1732:	cost=	0.013
Epoch	1733:	cost=	0.013
Epoch	1734:	cost=	0.013
Epoch	1735:	cost=	0.013

Epoch	1736:	cost=	0.013
Epoch	1737:	cost=	0.013
Epoch	1738:	cost=	0.013
Epoch	1739:	cost=	0.013
Epoch	1740:	cost=	0.013
Epoch	1741:	cost=	0.013
Epoch	1742:	cost=	0.013
Epoch	1743:	cost=	0.013
Epoch	1744:	cost=	0.013
Epoch	1745:	cost=	0.013
Epoch	1746:	cost=	0.013
Epoch	1747:	cost=	0.013
Epoch	1748:	cost=	0.013
Epoch	1749:	cost=	0.013
Epoch	1750:	cost=	0.013
Epoch	1751:	cost=	0.013
Epoch	1752:	cost=	0.013
Epoch	1753:	cost=	0.013
Epoch	1754:	cost=	0.013
Epoch	1755:	cost=	0.013
Epoch	1756:	cost=	0.013
Epoch	1757:	cost=	0.013
Epoch	1758:	cost=	0.012
Epoch	1759:	cost=	0.012
Epoch	1760:	cost=	0.012
Epoch	1761:	cost=	0.012
Epoch	1762:	cost=	0.012
Epoch	1763:	cost=	0.012
Epoch	1764:	cost=	0.012
Epoch	1765:	cost=	0.012
Epoch	1766:	cost=	0.012
Epoch	1767:	cost=	0.012
Epoch	1768:	cost=	0.012
Epoch	1769:	cost=	0.012
Epoch	1770:	cost=	0.012
Epoch	1771:	cost=	0.012
Epoch	1772:	cost=	0.012
Epoch	1773:	cost=	0.012
Epoch	1774:	cost=	0.012
Epoch	1775:	cost=	0.013
Epoch	1776:	cost=	0.013
Epoch	1777:	cost=	0.013
Epoch	1778:	cost=	0.014
Epoch	1779:	cost=	0.014
Epoch	1780:	cost=	0.013
Epoch	1781:	cost=	0.012
Epoch	1782:	cost=	0.012
Epoch	1783:	cost=	0.013

Epoch	1784:	cost=	0.013
Epoch	1785:	cost=	0.013
Epoch	1786:	cost=	0.012
Epoch	1787:	cost=	0.012
Epoch	1788:	cost=	0.013
Epoch	1789:	cost=	0.013
Epoch	1790:	cost=	0.012
Epoch	1791:	cost=	0.012
Epoch	1792:	cost=	0.012
Epoch	1793:	cost=	0.012
Epoch	1794:	cost=	0.012
Epoch	1795:	cost=	0.012
Epoch	1796:	cost=	0.012
Epoch	1797:	cost=	0.012
Epoch	1798:	cost=	0.012
Epoch	1799:	cost=	0.012
Epoch	1800:	cost=	0.012
Epoch	1801:	cost=	0.012
Epoch	1802:	cost=	0.012
Epoch	1803:	cost=	0.012
Epoch	1804:	cost=	0.012
Epoch	1805:	cost=	0.012
Epoch	1806:	cost=	0.012
Epoch	1807:	cost=	0.012
Epoch	1808:	cost=	0.012
Epoch	1809:	cost=	0.012
Epoch	1810:	cost=	0.012
Epoch	1811:	cost=	0.012
Epoch	1812:	cost=	0.012
Epoch	1813:	cost=	0.012
Epoch	1814:	cost=	0.012
Epoch	1815:	cost=	0.012
Epoch	1816:	cost=	0.012
Epoch	1817:	cost=	0.012
Epoch	1818:	cost=	0.012
Epoch	1819:	cost=	0.012
Epoch	1820:	cost=	0.012
Epoch	1821:	cost=	0.012
Epoch	1822:	cost=	0.012
Epoch	1823:	cost=	0.012
Epoch	1824:	cost=	0.012
Epoch	1825:	cost=	0.012
Epoch	1826:	cost=	0.012
Epoch	1827:	cost=	0.012
Epoch	1828:	cost=	0.012
Epoch	1829:	cost=	0.012
Epoch	1830:	cost=	0.012
Epoch	1831:	cost=	0.012

Epoch	1832:	cost=	0.012
Epoch	1833:	cost=	0.012
Epoch	1834:	cost=	0.012
Epoch	1835:	cost=	0.012
Epoch	1836:	cost=	0.012
Epoch	1837:	cost=	0.012
Epoch	1838:	cost=	0.012
Epoch	1839:	cost=	0.012
Epoch	1840:	cost=	0.012
Epoch	1841:	cost=	0.012
Epoch	1842:	cost=	0.011
Epoch	1843:	cost=	0.011
Epoch	1844:	cost=	0.011
Epoch	1845:	cost=	0.011
Epoch	1846:	cost=	0.011
Epoch	1847:	cost=	0.011
Epoch	1848:	cost=	0.011
Epoch	1849:	cost=	0.011
Epoch	1850:	cost=	0.011
Epoch	1851:	cost=	0.011
Epoch	1852:	cost=	0.011
Epoch	1853:	cost=	0.011
Epoch	1854:	cost=	0.011
Epoch	1855:	cost=	0.011
Epoch	1856:	cost=	0.011
Epoch	1857:	cost=	0.011
Epoch	1858:	cost=	0.011
Epoch	1859:	cost=	0.011
Epoch	1860:	cost=	0.011
Epoch	1861:	cost=	0.011
Epoch	1862:	cost=	0.011
Epoch	1863:	cost=	0.011
Epoch	1864:	cost=	0.011
Epoch	1865:	cost=	0.011
Epoch	1866:	cost=	0.011
Epoch	1867:	cost=	0.011
Epoch	1868:	cost=	0.011
Epoch	1869:	cost=	0.011
Epoch	1870:	cost=	0.011
Epoch	1871:	cost=	0.011
Epoch	1872:	cost=	0.011
Epoch	1873:	cost=	0.011
Epoch	1874:	cost=	0.011
Epoch	1875:	cost=	0.011
Epoch	1876:	cost=	0.011
Epoch	1877:	cost=	0.011
Epoch	1878:	cost=	0.011
Epoch	1879:	cost=	0.011

Epoch	1880:	cost=	0.011
Epoch	1881:	cost=	0.011
Epoch	1882:	cost=	0.011
Epoch	1883:	cost=	0.011
Epoch	1884:	cost=	0.011
Epoch	1885:	cost=	0.011
Epoch	1886:	cost=	0.011
Epoch	1887:	cost=	0.011
Epoch	1888:	cost=	0.011
Epoch	1889:	cost=	0.011
Epoch	1890:	cost=	0.011
Epoch	1891:	cost=	0.011
Epoch	1892:	cost=	0.011
Epoch	1893:	cost=	0.011
Epoch	1894:	cost=	0.011
Epoch	1895:	cost=	0.011
Epoch	1896:	cost=	0.011
Epoch	1897:	cost=	0.011
Epoch	1898:	cost=	0.011
Epoch	1899:	cost=	0.011
Epoch	1900:	cost=	0.011
Epoch	1901:	cost=	0.011
Epoch	1902:	cost=	0.011
Epoch	1903:	cost=	0.011
Epoch	1904:	cost=	0.011
Epoch	1905:	cost=	0.011
Epoch	1906:	cost=	0.011
Epoch	1907:	cost=	0.011
Epoch	1908:	cost=	0.011
Epoch	1909:	cost=	0.011
Epoch	1910:	cost=	0.011
Epoch	1911:	cost=	0.011
Epoch	1912:	cost=	0.011
Epoch	1913:	cost=	0.011
Epoch	1914:	cost=	0.011
Epoch	1915:	cost=	0.011
Epoch	1916:	cost=	0.011
Epoch	1917:	cost=	0.011
Epoch	1918:	cost=	0.011
Epoch	1919:	cost=	0.011
Epoch	1920:	cost=	0.011
Epoch	1921:	cost=	0.011
Epoch	1922:	cost=	0.011
Epoch	1923:	cost=	0.011
Epoch	1924:	cost=	0.011
Epoch	1925:	cost=	0.011
Epoch	1926:	cost=	0.011
Epoch	1927:	cost=	0.011

Epoch	1928:	cost=	0.011
Epoch	1929:	cost=	0.011
Epoch	1930:	cost=	0.011
Epoch	1931:	cost=	0.011
Epoch	1932:	cost=	0.011
Epoch	1933:	cost=	0.011
Epoch	1934:	cost=	0.011
Epoch	1935:	cost=	0.011
Epoch	1936:	cost=	0.011
Epoch	1937:	cost=	0.011
Epoch	1938:	cost=	0.011
Epoch	1939:	cost=	0.011
Epoch	1940:	cost=	0.011
Epoch	1941:	cost=	0.011
Epoch	1942:	cost=	0.011
Epoch	1943:	cost=	0.011
Epoch	1944:	cost=	0.011
Epoch	1945:	cost=	0.011
Epoch	1946:	cost=	0.011
Epoch	1947:	cost=	0.011
Epoch	1948:	cost=	0.011
Epoch	1949:	cost=	0.011
Epoch	1950:	cost=	0.011
Epoch	1951:	cost=	0.011
Epoch	1952:	cost=	0.011
Epoch	1953:	cost=	0.011
Epoch	1954:	cost=	0.011
Epoch	1955:	cost=	0.010
Epoch	1956:	cost=	0.010
Epoch	1957:	cost=	0.010
Epoch	1958:	cost=	0.010
Epoch	1959:	cost=	0.010
Epoch	1960:	cost=	0.010
Epoch	1961:	cost=	0.010
Epoch	1962:	cost=	0.010
Epoch	1963:	cost=	0.010
Epoch	1964:	cost=	0.010
Epoch	1965:	cost=	0.010
Epoch	1966:	cost=	0.010
Epoch	1967:	cost=	0.010
Epoch	1968:	cost=	0.010
Epoch	1969:	cost=	0.010
Epoch	1970:	cost=	0.010
Epoch	1971:	cost=	0.010
Epoch	1972:	cost=	0.010
Epoch	1973:	cost=	0.010
Epoch	1974:	cost=	0.010
Epoch	1975:	cost=	0.010

Epoch	1976:	cost=	0.010
Epoch	1977:	cost=	0.010
Epoch	1978:	cost=	0.010
Epoch	1979:	cost=	0.010
Epoch	1980:	cost=	0.010
Epoch	1981:	cost=	0.010
Epoch	1982:	cost=	0.010
Epoch	1983:	cost=	0.010
Epoch	1984:	cost=	0.010
Epoch	1985:	cost=	0.010
Epoch	1986:	cost=	0.010
Epoch	1987:	cost=	0.010
Epoch	1988:	cost=	0.010
Epoch	1989:	cost=	0.010
Epoch	1990:	cost=	0.010
Epoch	1991:	cost=	0.010
Epoch	1992:	cost=	0.010
Epoch	1993:	cost=	0.010
Epoch	1994:	cost=	0.011
Epoch	1995:	cost=	0.011
Epoch	1996:	cost=	0.011
Epoch	1997:	cost=	0.011
Epoch	1998:	cost=	0.012
Epoch	1999:	cost=	0.012
Epoch	2000:	cost=	0.011
Epoch	2001:	cost=	0.010
Epoch	2002:	cost=	0.010
Epoch	2003:	cost=	0.010
Epoch	2004:	cost=	0.011
Epoch	2005:	cost=	0.011
Epoch	2006:	cost=	0.011
Epoch	2007:	cost=	0.010
Epoch	2008:	cost=	0.010
Epoch	2009:	cost=	0.010
Epoch	2010:	cost=	0.011
Epoch	2011:	cost=	0.010
Epoch	2012:	cost=	0.010
Epoch	2013:	cost=	0.010
Epoch	2014:	cost=	0.010
Epoch	2015:	cost=	0.010
Epoch	2016:	cost=	0.010
Epoch	2017:	cost=	0.010
Epoch	2018:	cost=	0.010
Epoch	2019:	cost=	0.010
Epoch	2020:	cost=	0.010
Epoch	2021:	cost=	0.010
Epoch	2022:	cost=	0.010
Epoch	2023:	cost=	0.010

Epoch	2024:	cost=	0.010
Epoch	2025:	cost=	0.010
Epoch	2026:	cost=	0.010
Epoch	2027:	cost=	0.010
Epoch	2028:	cost=	0.010
Epoch	2029:	cost=	0.010
Epoch	2030:	cost=	0.010
Epoch	2031:	cost=	0.010
Epoch	2032:	cost=	0.010
Epoch	2033:	cost=	0.010
Epoch	2034:	cost=	0.010
Epoch	2035:	cost=	0.010
Epoch	2036:	cost=	0.010
Epoch	2037:	cost=	0.010
Epoch	2038:	cost=	0.010
Epoch	2039:	cost=	0.010
Epoch	2040:	cost=	0.010
Epoch	2041:	cost=	0.010
Epoch	2042:	cost=	0.010
Epoch	2043:	cost=	0.010
Epoch	2044:	cost=	0.010
Epoch	2045:	cost=	0.010
Epoch	2046:	cost=	0.010
Epoch	2047:	cost=	0.010
Epoch	2048:	cost=	0.010
Epoch	2049:	cost=	0.010
Epoch	2050:	cost=	0.010
Epoch	2051:	cost=	0.010
Epoch	2052:	cost=	0.010
Epoch	2053:	cost=	0.010
Epoch	2054:	cost=	0.010
Epoch	2055:	cost=	0.010
Epoch	2056:	cost=	0.010
Epoch	2057:	cost=	0.010
Epoch	2058:	cost=	0.010
Epoch	2059:	cost=	0.010
Epoch	2060:	cost=	0.010
Epoch	2061:	cost=	0.010
Epoch	2062:	cost=	0.010
Epoch	2063:	cost=	0.010
Epoch	2064:	cost=	0.010
Epoch	2065:	cost=	0.010
Epoch	2066:	cost=	0.010
Epoch	2067:	cost=	0.010
Epoch	2068:	cost=	0.010
Epoch	2069:	cost=	0.010
Epoch	2070:	cost=	0.010
Epoch	2071:	cost=	0.010

Epoch	2072:	cost=	0.010
Epoch	2073:	cost=	0.010
Epoch	2074:	cost=	0.010
Epoch	2075:	cost=	0.010
Epoch	2076:	cost=	0.010
Epoch	2077:	cost=	0.010
Epoch	2078:	cost=	0.010
Epoch	2079:	cost=	0.010
Epoch	2080:	cost=	0.010
Epoch	2081:	cost=	0.010
Epoch	2082:	cost=	0.010
Epoch	2083:	cost=	0.010
Epoch	2084:	cost=	0.010
Epoch	2085:	cost=	0.010
Epoch	2086:	cost=	0.010
Epoch	2087:	cost=	0.010
Epoch	2088:	cost=	0.010
Epoch	2089:	cost=	0.010
Epoch	2090:	cost=	0.010
Epoch	2091:	cost=	0.010
Epoch	2092:	cost=	0.010
Epoch	2093:	cost=	0.010
Epoch	2094:	cost=	0.010
Epoch	2095:	cost=	0.010
Epoch	2096:	cost=	0.010
Epoch	2097:	cost=	0.010
Epoch	2098:	cost=	0.010
Epoch	2099:	cost=	0.010
Epoch	2100:	cost=	0.010
Epoch	2101:	cost=	0.010
Epoch	2102:	cost=	0.010
Epoch	2103:	cost=	0.010
Epoch	2104:	cost=	0.010
Epoch	2105:	cost=	0.010
Epoch	2106:	cost=	0.010
Epoch	2107:	cost=	0.010
Epoch	2108:	cost=	0.010
Epoch	2109:	cost=	0.010
Epoch	2110:	cost=	0.010
Epoch	2111:	cost=	0.010
Epoch	2112:	cost=	0.010
Epoch	2113:	cost=	0.010
Epoch	2114:	cost=	0.010
Epoch	2115:	cost=	0.010
Epoch	2116:	cost=	0.010
Epoch	2117:	cost=	0.010
Epoch	2118:	cost=	0.010
Epoch	2119:	cost=	0.010

Epoch	2120:	cost=	0.010
Epoch	2121:	cost=	0.010
Epoch	2122:	cost=	0.010
Epoch	2123:	cost=	0.010
Epoch	2124:	cost=	0.010
Epoch	2125:	cost=	0.010
Epoch	2126:	cost=	0.010
Epoch	2127:	cost=	0.010
Epoch	2128:	cost=	0.010
Epoch	2129:	cost=	0.010
Epoch	2130:	cost=	0.010
Epoch	2131:	cost=	0.010
Epoch	2132:	cost=	0.010
Epoch	2133:	cost=	0.010
Epoch	2134:	cost=	0.010
Epoch	2135:	cost=	0.010
Epoch	2136:	cost=	0.010
Epoch	2137:	cost=	0.010
Epoch	2138:	cost=	0.010
Epoch	2139:	cost=	0.010
Epoch	2140:	cost=	0.010
Epoch	2141:	cost=	0.010
Epoch	2142:	cost=	0.009
Epoch	2143:	cost=	0.009
Epoch	2144:	cost=	0.009
Epoch	2145:	cost=	0.009
Epoch	2146:	cost=	0.009
Epoch	2147:	cost=	0.009
Epoch	2148:	cost=	0.009
Epoch	2149:	cost=	0.009
Epoch	2150:	cost=	0.009
Epoch	2151:	cost=	0.009
Epoch	2152:	cost=	0.009
Epoch	2153:	cost=	0.009
Epoch	2154:	cost=	0.009
Epoch	2155:	cost=	0.009
Epoch	2156:	cost=	0.009
Epoch	2157:	cost=	0.009
Epoch	2158:	cost=	0.009
Epoch	2159:	cost=	0.009
Epoch	2160:	cost=	0.009
Epoch	2161:	cost=	0.009
Epoch	2162:	cost=	0.009
Epoch	2163:	cost=	0.009
Epoch	2164:	cost=	0.009
Epoch	2165:	cost=	0.009
Epoch	2166:	cost=	0.009
Epoch	2167:	cost=	0.009

Epoch	2168:	cost=	0.009
Epoch	2169:	cost=	0.009
Epoch	2170:	cost=	0.009
Epoch	2171:	cost=	0.009
Epoch	2172:	cost=	0.009
Epoch	2173:	cost=	0.009
Epoch	2174:	cost=	0.009
Epoch	2175:	cost=	0.009
Epoch	2176:	cost=	0.009
Epoch	2177:	cost=	0.009
Epoch	2178:	cost=	0.009
Epoch	2179:	cost=	0.009
Epoch	2180:	cost=	0.009
Epoch	2181:	cost=	0.009
Epoch	2182:	cost=	0.009
Epoch	2183:	cost=	0.009
Epoch	2184:	cost=	0.009
Epoch	2185:	cost=	0.009
Epoch	2186:	cost=	0.009
Epoch	2187:	cost=	0.009
Epoch	2188:	cost=	0.010
Epoch	2189:	cost=	0.010
Epoch	2190:	cost=	0.010
Epoch	2191:	cost=	0.011
Epoch	2192:	cost=	0.011
Epoch	2193:	cost=	0.011
Epoch	2194:	cost=	0.011
Epoch	2195:	cost=	0.010
Epoch	2196:	cost=	0.010
Epoch	2197:	cost=	0.009
Epoch	2198:	cost=	0.010
Epoch	2199:	cost=	0.010
Epoch	2200:	cost=	0.010
Epoch	2201:	cost=	0.010
Epoch	2202:	cost=	0.009
Epoch	2203:	cost=	0.009
Epoch	2204:	cost=	0.010
Epoch	2205:	cost=	0.010
Epoch	2206:	cost=	0.010
Epoch	2207:	cost=	0.009
Epoch	2208:	cost=	0.009
Epoch	2209:	cost=	0.009
Epoch	2210:	cost=	0.010
Epoch	2211:	cost=	0.009
Epoch	2212:	cost=	0.009
Epoch	2213:	cost=	0.009
Epoch	2214:	cost=	0.009
Epoch	2215:	cost=	0.009

Epoch	2216:	cost=	0.009
Epoch	2217:	cost=	0.009
Epoch	2218:	cost=	0.009
Epoch	2219:	cost=	0.009
Epoch	2220:	cost=	0.009
Epoch	2221:	cost=	0.009
Epoch	2222:	cost=	0.009
Epoch	2223:	cost=	0.009
Epoch	2224:	cost=	0.009
Epoch	2225:	cost=	0.009
Epoch	2226:	cost=	0.009
Epoch	2227:	cost=	0.009
Epoch	2228:	cost=	0.009
Epoch	2229:	cost=	0.009
Epoch	2230:	cost=	0.009
Epoch	2231:	cost=	0.009
Epoch	2232:	cost=	0.009
Epoch	2233:	cost=	0.009
Epoch	2234:	cost=	0.009
Epoch	2235:	cost=	0.009
Epoch	2236:	cost=	0.009
Epoch	2237:	cost=	0.009
Epoch	2238:	cost=	0.009
Epoch	2239:	cost=	0.009
Epoch	2240:	cost=	0.009
Epoch	2241:	cost=	0.009
Epoch	2242:	cost=	0.009
Epoch	2243:	cost=	0.009
Epoch	2244:	cost=	0.009
Epoch	2245:	cost=	0.009
Epoch	2246:	cost=	0.009
Epoch	2247:	cost=	0.009
Epoch	2248:	cost=	0.009
Epoch	2249:	cost=	0.009
Epoch	2250:	cost=	0.009
Epoch	2251:	cost=	0.009
Epoch	2252:	cost=	0.009
Epoch	2253:	cost=	0.009
Epoch	2254:	cost=	0.009
Epoch	2255:	cost=	0.009
Epoch	2256:	cost=	0.009
Epoch	2257:	cost=	0.009
Epoch	2258:	cost=	0.009
Epoch	2259:	cost=	0.009
Epoch	2260:	cost=	0.009
Epoch	2261:	cost=	0.009
Epoch	2262:	cost=	0.009
Epoch	2263:	cost=	0.009

Epoch	2264:	cost=	0.009
Epoch	2265:	cost=	0.009
Epoch	2266:	cost=	0.009
Epoch	2267:	cost=	0.009
Epoch	2268:	cost=	0.009
Epoch	2269:	cost=	0.009
Epoch	2270:	cost=	0.009
Epoch	2271:	cost=	0.009
Epoch	2272:	cost=	0.009
Epoch	2273:	cost=	0.009
Epoch	2274:	cost=	0.009
Epoch	2275:	cost=	0.009
Epoch	2276:	cost=	0.009
Epoch	2277:	cost=	0.009
Epoch	2278:	cost=	0.009
Epoch	2279:	cost=	0.009
Epoch	2280:	cost=	0.009
Epoch	2281:	cost=	0.009
Epoch	2282:	cost=	0.009
Epoch	2283:	cost=	0.009
Epoch	2284:	cost=	0.009
Epoch	2285:	cost=	0.009
Epoch	2286:	cost=	0.009
Epoch	2287:	cost=	0.009
Epoch	2288:	cost=	0.009
Epoch	2289:	cost=	0.009
Epoch	2290:	cost=	0.009
Epoch	2291:	cost=	0.009
Epoch	2292:	cost=	0.009
Epoch	2293:	cost=	0.009
Epoch	2294:	cost=	0.009
Epoch	2295:	cost=	0.009
Epoch	2296:	cost=	0.009
Epoch	2297:	cost=	0.009
Epoch	2298:	cost=	0.009
Epoch	2299:	cost=	0.009
Epoch	2300:	cost=	0.009
Epoch	2301:	cost=	0.009
Epoch	2302:	cost=	0.009
Epoch	2303:	cost=	0.009
Epoch	2304:	cost=	0.009
Epoch	2305:	cost=	0.009
Epoch	2306:	cost=	0.009
Epoch	2307:	cost=	0.009
Epoch	2308:	cost=	0.009
Epoch	2309:	cost=	0.009
Epoch	2310:	cost=	0.009
Epoch	2311:	cost=	0.009

Epoch	2312:	cost=	0.009
Epoch	2313:	cost=	0.009
Epoch	2314:	cost=	0.009
Epoch	2315:	cost=	0.009
Epoch	2316:	cost=	0.009
Epoch	2317:	cost=	0.009
Epoch	2318:	cost=	0.009
Epoch	2319:	cost=	0.009
Epoch	2320:	cost=	0.009
Epoch	2321:	cost=	0.009
Epoch	2322:	cost=	0.009
Epoch	2323:	cost=	0.009
Epoch	2324:	cost=	0.009
Epoch	2325:	cost=	0.009
Epoch	2326:	cost=	0.009
Epoch	2327:	cost=	0.009
Epoch	2328:	cost=	0.009
Epoch	2329:	cost=	0.009
Epoch	2330:	cost=	0.009
Epoch	2331:	cost=	0.009
Epoch	2332:	cost=	0.009
Epoch	2333:	cost=	0.009
Epoch	2334:	cost=	0.009
Epoch	2335:	cost=	0.009
Epoch	2336:	cost=	0.009
Epoch	2337:	cost=	0.009
Epoch	2338:	cost=	0.009
Epoch	2339:	cost=	0.009
Epoch	2340:	cost=	0.009
Epoch	2341:	cost=	0.009
Epoch	2342:	cost=	0.009
Epoch	2343:	cost=	0.009
Epoch	2344:	cost=	0.009
Epoch	2345:	cost=	0.009
Epoch	2346:	cost=	0.009
Epoch	2347:	cost=	0.009
Epoch	2348:	cost=	0.009
Epoch	2349:	cost=	0.009
Epoch	2350:	cost=	0.009
Epoch	2351:	cost=	0.009
Epoch	2352:	cost=	0.009
Epoch	2353:	cost=	0.009
Epoch	2354:	cost=	0.009
Epoch	2355:	cost=	0.009
Epoch	2356:	cost=	0.009
Epoch	2357:	cost=	0.009
Epoch	2358:	cost=	0.009
Epoch	2359:	cost=	0.009

Epoch	2360:	cost=	0.009
Epoch	2361:	cost=	0.009
Epoch	2362:	cost=	0.009
Epoch	2363:	cost=	0.009
Epoch	2364:	cost=	0.009
Epoch	2365:	cost=	0.009
Epoch	2366:	cost=	0.009
Epoch	2367:	cost=	0.009
Epoch	2368:	cost=	0.009
Epoch	2369:	cost=	0.009
Epoch	2370:	cost=	0.009
Epoch	2371:	cost=	0.009
Epoch	2372:	cost=	0.009
Epoch	2373:	cost=	0.009
Epoch	2374:	cost=	0.009
Epoch	2375:	cost=	0.009
Epoch	2376:	cost=	0.009
Epoch	2377:	cost=	0.009
Epoch	2378:	cost=	0.009
Epoch	2379:	cost=	0.009
Epoch	2380:	cost=	0.009
Epoch	2381:	cost=	0.009
Epoch	2382:	cost=	0.009
Epoch	2383:	cost=	0.009
Epoch	2384:	cost=	0.009
Epoch	2385:	cost=	0.009
Epoch	2386:	cost=	0.009
Epoch	2387:	cost=	0.009
Epoch	2388:	cost=	0.009
Epoch	2389:	cost=	0.009
Epoch	2390:	cost=	0.009
Epoch	2391:	cost=	0.009
Epoch	2392:	cost=	0.009
Epoch	2393:	cost=	0.009
Epoch	2394:	cost=	0.009
Epoch	2395:	cost=	0.009
Epoch	2396:	cost=	0.010
Epoch	2397:	cost=	0.010
Epoch	2398:	cost=	0.011
Epoch	2399:	cost=	0.011
Epoch	2400:	cost=	0.011
Epoch	2401:	cost=	0.010
Epoch	2402:	cost=	0.009
Epoch	2403:	cost=	0.009
Epoch	2404:	cost=	0.009
Epoch	2405:	cost=	0.010
Epoch	2406:	cost=	0.010
Epoch	2407:	cost=	0.010

Epoch	2408:	cost=	0.009
Epoch	2409:	cost=	0.009
Epoch	2410:	cost=	0.009
Epoch	2411:	cost=	0.009
Epoch	2412:	cost=	0.009
Epoch	2413:	cost=	0.009
Epoch	2414:	cost=	0.009
Epoch	2415:	cost=	0.009
Epoch	2416:	cost=	0.009
Epoch	2417:	cost=	0.009
Epoch	2418:	cost=	0.009
Epoch	2419:	cost=	0.009
Epoch	2420:	cost=	0.009
Epoch	2421:	cost=	0.009
Epoch	2422:	cost=	0.009
Epoch	2423:	cost=	0.009
Epoch	2424:	cost=	0.009
Epoch	2425:	cost=	0.009
Epoch	2426:	cost=	0.009
Epoch	2427:	cost=	0.009
Epoch	2428:	cost=	0.009
Epoch	2429:	cost=	0.009
Epoch	2430:	cost=	0.009
Epoch	2431:	cost=	0.009
Epoch	2432:	cost=	0.009
Epoch	2433:	cost=	0.009
Epoch	2434:	cost=	0.009
Epoch	2435:	cost=	0.009
Epoch	2436:	cost=	0.009
Epoch	2437:	cost=	0.009
Epoch	2438:	cost=	0.009
Epoch	2439:	cost=	0.009
Epoch	2440:	cost=	0.009
Epoch	2441:	cost=	0.009
Epoch	2442:	cost=	0.009
Epoch	2443:	cost=	0.009
Epoch	2444:	cost=	0.009
Epoch	2445:	cost=	0.009
Epoch	2446:	cost=	0.009
Epoch	2447:	cost=	0.009
Epoch	2448:	cost=	0.009
Epoch	2449:	cost=	0.009
Epoch	2450:	cost=	0.009
Epoch	2451:	cost=	0.009
Epoch	2452:	cost=	0.009
Epoch	2453:	cost=	0.009
Epoch	2454:	cost=	0.009
Epoch	2455:	cost=	0.009

Epoch	2456:	cost=	0.009
Epoch	2457:	cost=	0.009
Epoch	2458:	cost=	0.009
Epoch	2459:	cost=	0.009
Epoch	2460:	cost=	0.009
Epoch	2461:	cost=	0.009
Epoch	2462:	cost=	0.009
Epoch	2463:	cost=	0.009
Epoch	2464:	cost=	0.009
Epoch	2465:	cost=	0.009
Epoch	2466:	cost=	0.009
Epoch	2467:	cost=	0.009
Epoch	2468:	cost=	0.009
Epoch	2469:	cost=	0.009
Epoch	2470:	cost=	0.009
Epoch	2471:	cost=	0.009
Epoch	2472:	cost=	0.009
Epoch	2473:	cost=	0.009
Epoch	2474:	cost=	0.009
Epoch	2475:	cost=	0.009
Epoch	2476:	cost=	0.009
Epoch	2477:	cost=	0.009
Epoch	2478:	cost=	0.009
Epoch	2479:	cost=	0.009
Epoch	2480:	cost=	0.009
Epoch	2481:	cost=	0.009
Epoch	2482:	cost=	0.009
Epoch	2483:	cost=	0.009
Epoch	2484:	cost=	0.009
Epoch	2485:	cost=	0.009
Epoch	2486:	cost=	0.009
Epoch	2487:	cost=	0.009
Epoch	2488:	cost=	0.009
Epoch	2489:	cost=	0.009
Epoch	2490:	cost=	0.009
Epoch	2491:	cost=	0.009
Epoch	2492:	cost=	0.009
Epoch	2493:	cost=	0.009
Epoch	2494:	cost=	0.009
Epoch	2495:	cost=	0.009
Epoch	2496:	cost=	0.009
Epoch	2497:	cost=	0.009
Epoch	2498:	cost=	0.009
Epoch	2499:	cost=	0.009
Epoch	2500:	cost=	0.009
Epoch	2501:	cost=	0.009
Epoch	2502:	cost=	0.009
Epoch	2503:	cost=	0.009

Epoch	2504:	cost=	0.009
Epoch	2505:	cost=	0.009
Epoch	2506:	cost=	0.009
Epoch	2507:	cost=	0.009
Epoch	2508:	cost=	0.009
Epoch	2509:	cost=	0.009
Epoch	2510:	cost=	0.009
Epoch	2511:	cost=	0.009
Epoch	2512:	cost=	0.009
Epoch	2513:	cost=	0.009
Epoch	2514:	cost=	0.009
Epoch	2515:	cost=	0.009
Epoch	2516:	cost=	0.009
Epoch	2517:	cost=	0.009
Epoch	2518:	cost=	0.009
Epoch	2519:	cost=	0.009
Epoch	2520:	cost=	0.009
Epoch	2521:	cost=	0.009
Epoch	2522:	cost=	0.009
Epoch	2523:	cost=	0.009
Epoch	2524:	cost=	0.009
Epoch	2525:	cost=	0.009
Epoch	2526:	cost=	0.009
Epoch	2527:	cost=	0.009
Epoch	2528:	cost=	0.009
Epoch	2529:	cost=	0.009
Epoch	2530:	cost=	0.009
Epoch	2531:	cost=	0.009
Epoch	2532:	cost=	0.009
Epoch	2533:	cost=	0.009
Epoch	2534:	cost=	0.009
Epoch	2535:	cost=	0.009
Epoch	2536:	cost=	0.009
Epoch	2537:	cost=	0.009
Epoch	2538:	cost=	0.009
Epoch	2539:	cost=	0.009
Epoch	2540:	cost=	0.009
Epoch	2541:	cost=	0.009
Epoch	2542:	cost=	0.009
Epoch	2543:	cost=	0.009
Epoch	2544:	cost=	0.009
Epoch	2545:	cost=	0.009
Epoch	2546:	cost=	0.009
Epoch	2547:	cost=	0.009
Epoch	2548:	cost=	0.009
Epoch	2549:	cost=	0.009
Epoch	2550:	cost=	0.009
Epoch	2551:	cost=	0.009

Epoch	2552:	cost=	0.009
Epoch	2553:	cost=	0.009
Epoch	2554:	cost=	0.009
Epoch	2555:	cost=	0.009
Epoch	2556:	cost=	0.009
Epoch	2557:	cost=	0.009
Epoch	2558:	cost=	0.009
Epoch	2559:	cost=	0.009
Epoch	2560:	cost=	0.009
Epoch	2561:	cost=	0.009
Epoch	2562:	cost=	0.009
Epoch	2563:	cost=	0.009
Epoch	2564:	cost=	0.009
Epoch	2565:	cost=	0.009
Epoch	2566:	cost=	0.009
Epoch	2567:	cost=	0.009
Epoch	2568:	cost=	0.009
Epoch	2569:	cost=	0.009
Epoch	2570:	cost=	0.009
Epoch	2571:	cost=	0.009
Epoch	2572:	cost=	0.009
Epoch	2573:	cost=	0.009
Epoch	2574:	cost=	0.009
Epoch	2575:	cost=	0.009
Epoch	2576:	cost=	0.009
Epoch	2577:	cost=	0.009
Epoch	2578:	cost=	0.009
Epoch	2579:	cost=	0.009
Epoch	2580:	cost=	0.009
Epoch	2581:	cost=	0.009
Epoch	2582:	cost=	0.009
Epoch	2583:	cost=	0.009
Epoch	2584:	cost=	0.009
Epoch	2585:	cost=	0.009
Epoch	2586:	cost=	0.009
Epoch	2587:	cost=	0.009
Epoch	2588:	cost=	0.009
Epoch	2589:	cost=	0.009
Epoch	2590:	cost=	0.009
Epoch	2591:	cost=	0.009
Epoch	2592:	cost=	0.009
Epoch	2593:	cost=	0.009
Epoch	2594:	cost=	0.009
Epoch	2595:	cost=	0.009
Epoch	2596:	cost=	0.009
Epoch	2597:	cost=	0.009
Epoch	2598:	cost=	0.009
Epoch	2599:	cost=	0.009

Epoch	2600:	cost=	0.009
Epoch	2601:	cost=	0.009
Epoch	2602:	cost=	0.009
Epoch	2603:	cost=	0.009
Epoch	2604:	cost=	0.009
Epoch	2605:	cost=	0.009
Epoch	2606:	cost=	0.009
Epoch	2607:	cost=	0.009
Epoch	2608:	cost=	0.009
Epoch	2609:	cost=	0.009
Epoch	2610:	cost=	0.009
Epoch	2611:	cost=	0.009
Epoch	2612:	cost=	0.009
Epoch	2613:	cost=	0.009
Epoch	2614:	cost=	0.009
Epoch	2615:	cost=	0.009
Epoch	2616:	cost=	0.009
Epoch	2617:	cost=	0.009
Epoch	2618:	cost=	0.009
Epoch	2619:	cost=	0.009
Epoch	2620:	cost=	0.009
Epoch	2621:	cost=	0.009
Epoch	2622:	cost=	0.009
Epoch	2623:	cost=	0.010
Epoch	2624:	cost=	0.010
Epoch	2625:	cost=	0.011
Epoch	2626:	cost=	0.012
Epoch	2627:	cost=	0.012
Epoch	2628:	cost=	0.010
Epoch	2629:	cost=	0.009
Epoch	2630:	cost=	0.009
Epoch	2631:	cost=	0.009
Epoch	2632:	cost=	0.010
Epoch	2633:	cost=	0.010
Epoch	2634:	cost=	0.009
Epoch	2635:	cost=	0.008
Epoch	2636:	cost=	0.009
Epoch	2637:	cost=	0.009
Epoch	2638:	cost=	0.009
Epoch	2639:	cost=	0.009
Epoch	2640:	cost=	0.008
Epoch	2641:	cost=	0.009
Epoch	2642:	cost=	0.009
Epoch	2643:	cost=	0.009
Epoch	2644:	cost=	0.009
Epoch	2645:	cost=	0.009
Epoch	2646:	cost=	0.009
Epoch	2647:	cost=	0.009

Epoch	2648:	cost=	0.009
Epoch	2649:	cost=	0.008
Epoch	2650:	cost=	0.009
Epoch	2651:	cost=	0.009
Epoch	2652:	cost=	0.009
Epoch	2653:	cost=	0.009
Epoch	2654:	cost=	0.008
Epoch	2655:	cost=	0.009
Epoch	2656:	cost=	0.009
Epoch	2657:	cost=	0.009
Epoch	2658:	cost=	0.008
Epoch	2659:	cost=	0.008
Epoch	2660:	cost=	0.009
Epoch	2661:	cost=	0.009
Epoch	2662:	cost=	0.008
Epoch	2663:	cost=	0.008
Epoch	2664:	cost=	0.008
Epoch	2665:	cost=	0.009
Epoch	2666:	cost=	0.008
Epoch	2667:	cost=	0.008
Epoch	2668:	cost=	0.008
Epoch	2669:	cost=	0.008
Epoch	2670:	cost=	0.008
Epoch	2671:	cost=	0.008
Epoch	2672:	cost=	0.008
Epoch	2673:	cost=	0.008
Epoch	2674:	cost=	0.008
Epoch	2675:	cost=	0.008
Epoch	2676:	cost=	0.008
Epoch	2677:	cost=	0.008
Epoch	2678:	cost=	0.008
Epoch	2679:	cost=	0.008
Epoch	2680:	cost=	0.008
Epoch	2681:	cost=	0.008
Epoch	2682:	cost=	0.008
Epoch	2683:	cost=	0.008
Epoch	2684:	cost=	0.008
Epoch	2685:	cost=	0.008
Epoch	2686:	cost=	0.008
Epoch	2687:	cost=	0.008
Epoch	2688:	cost=	0.008
Epoch	2689:	cost=	0.008
Epoch	2690:	cost=	0.008
Epoch	2691:	cost=	0.008
Epoch	2692:	cost=	0.008
Epoch	2693:	cost=	0.008
Epoch	2694:	cost=	0.008
Epoch	2695:	cost=	0.008

Epoch	2696:	cost=	0.008
Epoch	2697:	cost=	0.008
Epoch	2698:	cost=	0.008
Epoch	2699:	cost=	0.008
Epoch	2700:	cost=	0.008
Epoch	2701:	cost=	0.008
Epoch	2702:	cost=	0.008
Epoch	2703:	cost=	0.008
Epoch	2704:	cost=	0.008
Epoch	2705:	cost=	0.008
Epoch	2706:	cost=	0.008
Epoch	2707:	cost=	0.008
Epoch	2708:	cost=	0.008
Epoch	2709:	cost=	0.008
Epoch	2710:	cost=	0.008
Epoch	2711:	cost=	0.008
Epoch	2712:	cost=	0.008
Epoch	2713:	cost=	0.008
Epoch	2714:	cost=	0.008
Epoch	2715:	cost=	0.008
Epoch	2716:	cost=	0.008
Epoch	2717:	cost=	0.008
Epoch	2718:	cost=	0.008
Epoch	2719:	cost=	0.008
Epoch	2720:	cost=	0.008
Epoch	2721:	cost=	0.008
Epoch	2722:	cost=	0.008
Epoch	2723:	cost=	0.008
Epoch	2724:	cost=	0.008
Epoch	2725:	cost=	0.008
Epoch	2726:	cost=	0.008
Epoch	2727:	cost=	0.008
Epoch	2728:	cost=	0.008
Epoch	2729:	cost=	0.008
Epoch	2730:	cost=	0.008
Epoch	2731:	cost=	0.008
Epoch	2732:	cost=	0.008
Epoch	2733:	cost=	0.008
Epoch	2734:	cost=	0.008
Epoch	2735:	cost=	0.008
Epoch	2736:	cost=	0.008
Epoch	2737:	cost=	0.008
Epoch	2738:	cost=	0.008
Epoch	2739:	cost=	0.008
Epoch	2740:	cost=	0.008
Epoch	2741:	cost=	0.008
Epoch	2742:	cost=	0.008
Epoch	2743:	cost=	0.008

Epoch	2744:	cost=	0.008
Epoch	2745:	cost=	0.008
Epoch	2746:	cost=	0.008
Epoch	2747:	cost=	0.008
Epoch	2748:	cost=	0.008
Epoch	2749:	cost=	0.008
Epoch	2750:	cost=	0.008
Epoch	2751:	cost=	0.008
Epoch	2752:	cost=	0.008
Epoch	2753:	cost=	0.008
Epoch	2754:	cost=	0.008
Epoch	2755:	cost=	0.008
Epoch	2756:	cost=	0.008
Epoch	2757:	cost=	0.008
Epoch	2758:	cost=	0.008
Epoch	2759:	cost=	0.008
Epoch	2760:	cost=	0.008
Epoch	2761:	cost=	0.008
Epoch	2762:	cost=	0.008
Epoch	2763:	cost=	0.008
Epoch	2764:	cost=	0.008
Epoch	2765:	cost=	0.008
Epoch	2766:	cost=	0.008
Epoch	2767:	cost=	0.008
Epoch	2768:	cost=	0.008
Epoch	2769:	cost=	0.008
Epoch	2770:	cost=	0.008
Epoch	2771:	cost=	0.008
Epoch	2772:	cost=	0.008
Epoch	2773:	cost=	0.008
Epoch	2774:	cost=	0.008
Epoch	2775:	cost=	0.008
Epoch	2776:	cost=	0.008
Epoch	2777:	cost=	0.008
Epoch	2778:	cost=	0.008
Epoch	2779:	cost=	0.008
Epoch	2780:	cost=	0.008
Epoch	2781:	cost=	0.008
Epoch	2782:	cost=	0.008
Epoch	2783:	cost=	0.008
Epoch	2784:	cost=	0.008
Epoch	2785:	cost=	0.008
Epoch	2786:	cost=	0.008
Epoch	2787:	cost=	0.008
Epoch	2788:	cost=	0.008
Epoch	2789:	cost=	0.008
Epoch	2790:	cost=	0.008
Epoch	2791:	cost=	0.008

Epoch	2792:	cost=	0.008
Epoch	2793:	cost=	0.008
Epoch	2794:	cost=	0.008
Epoch	2795:	cost=	0.008
Epoch	2796:	cost=	0.008
Epoch	2797:	cost=	0.008
Epoch	2798:	cost=	0.008
Epoch	2799:	cost=	0.008
Epoch	2800:	cost=	0.008
Epoch	2801:	cost=	0.008
Epoch	2802:	cost=	0.008
Epoch	2803:	cost=	0.008
Epoch	2804:	cost=	0.008
Epoch	2805:	cost=	0.008
Epoch	2806:	cost=	0.008
Epoch	2807:	cost=	0.008
Epoch	2808:	cost=	0.008
Epoch	2809:	cost=	0.008
Epoch	2810:	cost=	0.008
Epoch	2811:	cost=	0.008
Epoch	2812:	cost=	0.008
Epoch	2813:	cost=	0.008
Epoch	2814:	cost=	0.008
Epoch	2815:	cost=	0.008
Epoch	2816:	cost=	0.008
Epoch	2817:	cost=	0.008
Epoch	2818:	cost=	0.008
Epoch	2819:	cost=	0.008
Epoch	2820:	cost=	0.008
Epoch	2821:	cost=	0.008
Epoch	2822:	cost=	0.008
Epoch	2823:	cost=	0.008
Epoch	2824:	cost=	0.008
Epoch	2825:	cost=	0.008
Epoch	2826:	cost=	0.008
Epoch	2827:	cost=	0.008
Epoch	2828:	cost=	0.008
Epoch	2829:	cost=	0.008
Epoch	2830:	cost=	0.008
Epoch	2831:	cost=	0.008
Epoch	2832:	cost=	0.008
Epoch	2833:	cost=	0.008
Epoch	2834:	cost=	0.008
Epoch	2835:	cost=	0.008
Epoch	2836:	cost=	0.008
Epoch	2837:	cost=	0.008
Epoch	2838:	cost=	0.008
Epoch	2839:	cost=	0.008

Epoch	2840:	cost=	0.008
Epoch	2841:	cost=	0.008
Epoch	2842:	cost=	0.008
Epoch	2843:	cost=	0.008
Epoch	2844:	cost=	0.008
Epoch	2845:	cost=	0.008
Epoch	2846:	cost=	0.008
Epoch	2847:	cost=	0.008
Epoch	2848:	cost=	0.008
Epoch	2849:	cost=	0.008
Epoch	2850:	cost=	0.008
Epoch	2851:	cost=	0.008
Epoch	2852:	cost=	0.008
Epoch	2853:	cost=	0.008
Epoch	2854:	cost=	0.008
Epoch	2855:	cost=	0.008
Epoch	2856:	cost=	0.008
Epoch	2857:	cost=	0.008
Epoch	2858:	cost=	0.008
Epoch	2859:	cost=	0.008
Epoch	2860:	cost=	0.008
Epoch	2861:	cost=	0.008
Epoch	2862:	cost=	0.008
Epoch	2863:	cost=	0.008
Epoch	2864:	cost=	0.008
Epoch	2865:	cost=	0.008
Epoch	2866:	cost=	0.008
Epoch	2867:	cost=	0.008
Epoch	2868:	cost=	0.008
Epoch	2869:	cost=	0.008
Epoch	2870:	cost=	0.008
Epoch	2871:	cost=	0.008
Epoch	2872:	cost=	0.008
Epoch	2873:	cost=	0.008
Epoch	2874:	cost=	0.008
Epoch	2875:	cost=	0.008
Epoch	2876:	cost=	0.008
Epoch	2877:	cost=	0.008
Epoch	2878:	cost=	0.008
Epoch	2879:	cost=	0.008
Epoch	2880:	cost=	0.008
Epoch	2881:	cost=	0.008
Epoch	2882:	cost=	0.008
Epoch	2883:	cost=	0.008
Epoch	2884:	cost=	0.008
Epoch	2885:	cost=	0.008
Epoch	2886:	cost=	0.008
Epoch	2887:	cost=	0.008

Epoch	2888:	cost=	0.008
Epoch	2889:	cost=	0.008
Epoch	2890:	cost=	0.008
Epoch	2891:	cost=	0.008
Epoch	2892:	cost=	0.008
Epoch	2893:	cost=	0.008
Epoch	2894:	cost=	0.009
Epoch	2895:	cost=	0.009
Epoch	2896:	cost=	0.009
Epoch	2897:	cost=	0.010
Epoch	2898:	cost=	0.011
Epoch	2899:	cost=	0.011
Epoch	2900:	cost=	0.011
Epoch	2901:	cost=	0.011
Epoch	2902:	cost=	0.009
Epoch	2903:	cost=	0.008
Epoch	2904:	cost=	0.009
Epoch	2905:	cost=	0.009
Epoch	2906:	cost=	0.010
Epoch	2907:	cost=	0.009
Epoch	2908:	cost=	0.009
Epoch	2909:	cost=	0.008
Epoch	2910:	cost=	0.009
Epoch	2911:	cost=	0.009
Epoch	2912:	cost=	0.009
Epoch	2913:	cost=	0.009
Epoch	2914:	cost=	0.008
Epoch	2915:	cost=	0.008
Epoch	2916:	cost=	0.009
Epoch	2917:	cost=	0.009
Epoch	2918:	cost=	0.008
Epoch	2919:	cost=	0.008
Epoch	2920:	cost=	0.008
Epoch	2921:	cost=	0.008
Epoch	2922:	cost=	0.008
Epoch	2923:	cost=	0.008
Epoch	2924:	cost=	0.008
Epoch	2925:	cost=	0.008
Epoch	2926:	cost=	0.008
Epoch	2927:	cost=	0.008
Epoch	2928:	cost=	0.008
Epoch	2929:	cost=	0.008
Epoch	2930:	cost=	0.008
Epoch	2931:	cost=	0.008
Epoch	2932:	cost=	0.008
Epoch	2933:	cost=	0.008
Epoch	2934:	cost=	0.008
Epoch	2935:	cost=	0.008

Epoch	2936:	cost=	0.008
Epoch	2937:	cost=	0.008
Epoch	2938:	cost=	0.008
Epoch	2939:	cost=	0.008
Epoch	2940:	cost=	0.008
Epoch	2941:	cost=	0.008
Epoch	2942:	cost=	0.008
Epoch	2943:	cost=	0.008
Epoch	2944:	cost=	0.008
Epoch	2945:	cost=	0.008
Epoch	2946:	cost=	0.008
Epoch	2947:	cost=	0.008
Epoch	2948:	cost=	0.008
Epoch	2949:	cost=	0.008
Epoch	2950:	cost=	0.008
Epoch	2951:	cost=	0.008
Epoch	2952:	cost=	0.008
Epoch	2953:	cost=	0.008
Epoch	2954:	cost=	0.008
Epoch	2955:	cost=	0.008
Epoch	2956:	cost=	0.008
Epoch	2957:	cost=	0.008
Epoch	2958:	cost=	0.008
Epoch	2959:	cost=	0.008
Epoch	2960:	cost=	0.008
Epoch	2961:	cost=	0.008
Epoch	2962:	cost=	0.008
Epoch	2963:	cost=	0.008
Epoch	2964:	cost=	0.008
Epoch	2965:	cost=	0.008
Epoch	2966:	cost=	0.008
Epoch	2967:	cost=	0.008
Epoch	2968:	cost=	0.008
Epoch	2969:	cost=	0.008
Epoch	2970:	cost=	0.008
Epoch	2971:	cost=	0.008
Epoch	2972:	cost=	0.008
Epoch	2973:	cost=	0.008
Epoch	2974:	cost=	0.008
Epoch	2975:	cost=	0.008
Epoch	2976:	cost=	0.008
Epoch	2977:	cost=	0.008
Epoch	2978:	cost=	0.008
Epoch	2979:	cost=	0.008
Epoch	2980:	cost=	0.008
Epoch	2981:	cost=	0.008
Epoch	2982:	cost=	0.008
Epoch	2983:	cost=	0.008

Epoch	2984:	cost=	0.008
Epoch	2985:	cost=	0.008
Epoch	2986:	cost=	0.008
Epoch	2987:	cost=	0.008
Epoch	2988:	cost=	0.008
Epoch	2989:	cost=	0.008
Epoch	2990:	cost=	0.008
Epoch	2991:	cost=	0.008
Epoch	2992:	cost=	0.008
Epoch	2993:	cost=	0.008
Epoch	2994:	cost=	0.008
Epoch	2995:	cost=	0.008
Epoch	2996:	cost=	0.008
Epoch	2997:	cost=	0.008
Epoch	2998:	cost=	0.008
Epoch	2999:	cost=	0.008
Epoch	3000:	cost=	0.008
Epoch	3001:	cost=	0.008
Epoch	3002:	cost=	0.008
Epoch	3003:	cost=	0.008
Epoch	3004:	cost=	0.008
Epoch	3005:	cost=	0.008
Epoch	3006:	cost=	0.008
Epoch	3007:	cost=	0.008
Epoch	3008:	cost=	0.008
Epoch	3009:	cost=	0.008
Epoch	3010:	cost=	0.008
Epoch	3011:	cost=	0.008
Epoch	3012:	cost=	0.008
Epoch	3013:	cost=	0.008
Epoch	3014:	cost=	0.008
Epoch	3015:	cost=	0.008
Epoch	3016:	cost=	0.008
Epoch	3017:	cost=	0.008
Epoch	3018:	cost=	0.008
Epoch	3019:	cost=	0.008
Epoch	3020:	cost=	0.008
Epoch	3021:	cost=	0.008
Epoch	3022:	cost=	0.008
Epoch	3023:	cost=	0.008
Epoch	3024:	cost=	0.008
Epoch	3025:	cost=	0.008
Epoch	3026:	cost=	0.008
Epoch	3027:	cost=	0.008
Epoch	3028:	cost=	0.008
Epoch	3029:	cost=	0.008
Epoch	3030:	cost=	0.008
Epoch	3031:	cost=	0.008

Epoch	3032:	cost=	0.008
Epoch	3033:	cost=	0.008
Epoch	3034:	cost=	0.008
Epoch	3035:	cost=	0.008
Epoch	3036:	cost=	0.008
Epoch	3037:	cost=	0.008
Epoch	3038:	cost=	0.008
Epoch	3039:	cost=	0.008
Epoch	3040:	cost=	0.008
Epoch	3041:	cost=	0.008
Epoch	3042:	cost=	0.008
Epoch	3043:	cost=	0.008
Epoch	3044:	cost=	0.008
Epoch	3045:	cost=	0.008
Epoch	3046:	cost=	0.008
Epoch	3047:	cost=	0.008
Epoch	3048:	cost=	0.008
Epoch	3049:	cost=	0.008
Epoch	3050:	cost=	0.008
Epoch	3051:	cost=	0.008
Epoch	3052:	cost=	0.008
Epoch	3053:	cost=	0.008
Epoch	3054:	cost=	0.008
Epoch	3055:	cost=	0.008
Epoch	3056:	cost=	0.008
Epoch	3057:	cost=	0.008
Epoch	3058:	cost=	0.008
Epoch	3059:	cost=	0.008
Epoch	3060:	cost=	0.008
Epoch	3061:	cost=	0.008
Epoch	3062:	cost=	0.008
Epoch	3063:	cost=	0.008
Epoch	3064:	cost=	0.008
Epoch	3065:	cost=	0.008
Epoch	3066:	cost=	0.008
Epoch	3067:	cost=	0.008
Epoch	3068:	cost=	0.008
Epoch	3069:	cost=	0.008
Epoch	3070:	cost=	0.008
Epoch	3071:	cost=	0.008
Epoch	3072:	cost=	0.008
Epoch	3073:	cost=	0.008
Epoch	3074:	cost=	0.008
Epoch	3075:	cost=	0.008
Epoch	3076:	cost=	0.008
Epoch	3077:	cost=	0.008
Epoch	3078:	cost=	0.008
Epoch	3079:	cost=	0.008

Epoch	3080:	cost=	0.008
Epoch	3081:	cost=	0.008
Epoch	3082:	cost=	0.008
Epoch	3083:	cost=	0.008
Epoch	3084:	cost=	0.008
Epoch	3085:	cost=	0.008
Epoch	3086:	cost=	0.008
Epoch	3087:	cost=	0.008
Epoch	3088:	cost=	0.008
Epoch	3089:	cost=	0.008
Epoch	3090:	cost=	0.008
Epoch	3091:	cost=	0.008
Epoch	3092:	cost=	0.008
Epoch	3093:	cost=	0.008
Epoch	3094:	cost=	0.008
Epoch	3095:	cost=	0.008
Epoch	3096:	cost=	0.008
Epoch	3097:	cost=	0.008
Epoch	3098:	cost=	0.008
Epoch	3099:	cost=	0.008
Epoch	3100:	cost=	0.008
Epoch	3101:	cost=	0.008
Epoch	3102:	cost=	0.008
Epoch	3103:	cost=	0.008
Epoch	3104:	cost=	0.008
Epoch	3105:	cost=	0.008
Epoch	3106:	cost=	0.008
Epoch	3107:	cost=	0.008
Epoch	3108:	cost=	0.008
Epoch	3109:	cost=	0.008
Epoch	3110:	cost=	0.008
Epoch	3111:	cost=	0.008
Epoch	3112:	cost=	0.008
Epoch	3113:	cost=	0.008
Epoch	3114:	cost=	0.008
Epoch	3115:	cost=	0.008
Epoch	3116:	cost=	0.008
Epoch	3117:	cost=	0.008
Epoch	3118:	cost=	0.008
Epoch	3119:	cost=	0.008
Epoch	3120:	cost=	0.008
Epoch	3121:	cost=	0.008
Epoch	3122:	cost=	0.008
Epoch	3123:	cost=	0.008
Epoch	3124:	cost=	0.008
Epoch	3125:	cost=	0.008
Epoch	3126:	cost=	0.008
Epoch	3127:	cost=	0.008

Epoch	3128:	cost=	0.008
Epoch	3129:	cost=	0.008
Epoch	3130:	cost=	0.008
Epoch	3131:	cost=	0.008
Epoch	3132:	cost=	0.008
Epoch	3133:	cost=	0.008
Epoch	3134:	cost=	0.008
Epoch	3135:	cost=	0.008
Epoch	3136:	cost=	0.008
Epoch	3137:	cost=	0.008
Epoch	3138:	cost=	0.008
Epoch	3139:	cost=	0.008
Epoch	3140:	cost=	0.008
Epoch	3141:	cost=	0.008
Epoch	3142:	cost=	0.008
Epoch	3143:	cost=	0.008
Epoch	3144:	cost=	0.009
Epoch	3145:	cost=	0.009
Epoch	3146:	cost=	0.010
Epoch	3147:	cost=	0.010
Epoch	3148:	cost=	0.011
Epoch	3149:	cost=	0.012
Epoch	3150:	cost=	0.011
Epoch	3151:	cost=	0.010
Epoch	3152:	cost=	0.009
Epoch	3153:	cost=	0.008
Epoch	3154:	cost=	0.009
Epoch	3155:	cost=	0.009
Epoch	3156:	cost=	0.010
Epoch	3157:	cost=	0.009
Epoch	3158:	cost=	0.008
Epoch	3159:	cost=	0.008
Epoch	3160:	cost=	0.009
Epoch	3161:	cost=	0.009
Epoch	3162:	cost=	0.009
Epoch	3163:	cost=	0.008
Epoch	3164:	cost=	0.008
Epoch	3165:	cost=	0.008
Epoch	3166:	cost=	0.009
Epoch	3167:	cost=	0.008
Epoch	3168:	cost=	0.008
Epoch	3169:	cost=	0.008
Epoch	3170:	cost=	0.008
Epoch	3171:	cost=	0.008
Epoch	3172:	cost=	0.008
Epoch	3173:	cost=	0.008
Epoch	3174:	cost=	0.008
Epoch	3175:	cost=	0.008

Epoch	3176:	cost=	0.008
Epoch	3177:	cost=	0.008
Epoch	3178:	cost=	0.008
Epoch	3179:	cost=	0.008
Epoch	3180:	cost=	0.008
Epoch	3181:	cost=	0.008
Epoch	3182:	cost=	0.008
Epoch	3183:	cost=	0.008
Epoch	3184:	cost=	0.008
Epoch	3185:	cost=	0.008
Epoch	3186:	cost=	0.008
Epoch	3187:	cost=	0.008
Epoch	3188:	cost=	0.008
Epoch	3189:	cost=	0.008
Epoch	3190:	cost=	0.008
Epoch	3191:	cost=	0.008
Epoch	3192:	cost=	0.008
Epoch	3193:	cost=	0.008
Epoch	3194:	cost=	0.008
Epoch	3195:	cost=	0.008
Epoch	3196:	cost=	0.008
Epoch	3197:	cost=	0.008
Epoch	3198:	cost=	0.008
Epoch	3199:	cost=	0.008
Epoch	3200:	cost=	0.008
Epoch	3201:	cost=	0.008
Epoch	3202:	cost=	0.008
Epoch	3203:	cost=	0.008
Epoch	3204:	cost=	0.008
Epoch	3205:	cost=	0.008
Epoch	3206:	cost=	0.008
Epoch	3207:	cost=	0.008
Epoch	3208:	cost=	0.008
Epoch	3209:	cost=	0.008
Epoch	3210:	cost=	0.008
Epoch	3211:	cost=	0.008
Epoch	3212:	cost=	0.008
Epoch	3213:	cost=	0.008
Epoch	3214:	cost=	0.008
Epoch	3215:	cost=	0.008
Epoch	3216:	cost=	0.008
Epoch	3217:	cost=	0.008
Epoch	3218:	cost=	0.008
Epoch	3219:	cost=	0.008
Epoch	3220:	cost=	0.008
Epoch	3221:	cost=	0.008
Epoch	3222:	cost=	0.008
Epoch	3223:	cost=	0.008

Epoch	3224:	cost=	0.008
Epoch	3225:	cost=	0.008
Epoch	3226:	cost=	0.008
Epoch	3227:	cost=	0.008
Epoch	3228:	cost=	0.008
Epoch	3229:	cost=	0.008
Epoch	3230:	cost=	0.008
Epoch	3231:	cost=	0.008
Epoch	3232:	cost=	0.008
Epoch	3233:	cost=	0.008
Epoch	3234:	cost=	0.008
Epoch	3235:	cost=	0.008
Epoch	3236:	cost=	0.008
Epoch	3237:	cost=	0.008
Epoch	3238:	cost=	0.008
Epoch	3239:	cost=	0.008
Epoch	3240:	cost=	0.008
Epoch	3241:	cost=	0.008
Epoch	3242:	cost=	0.008
Epoch	3243:	cost=	0.008
Epoch	3244:	cost=	0.008
Epoch	3245:	cost=	0.008
Epoch	3246:	cost=	0.008
Epoch	3247:	cost=	0.008
Epoch	3248:	cost=	0.008
Epoch	3249:	cost=	0.008
Epoch	3250:	cost=	0.008
Epoch	3251:	cost=	0.008
Epoch	3252:	cost=	0.008
Epoch	3253:	cost=	0.008
Epoch	3254:	cost=	0.008
Epoch	3255:	cost=	0.008
Epoch	3256:	cost=	0.008
Epoch	3257:	cost=	0.008
Epoch	3258:	cost=	0.008
Epoch	3259:	cost=	0.008
Epoch	3260:	cost=	0.008
Epoch	3261:	cost=	0.008
Epoch	3262:	cost=	0.008
Epoch	3263:	cost=	0.008
Epoch	3264:	cost=	0.008
Epoch	3265:	cost=	0.008
Epoch	3266:	cost=	0.008
Epoch	3267:	cost=	0.008
Epoch	3268:	cost=	0.008
Epoch	3269:	cost=	0.008
Epoch	3270:	cost=	0.008
Epoch	3271:	cost=	0.008

Epoch	3272:	cost=	0.008
Epoch	3273:	cost=	0.008
Epoch	3274:	cost=	0.008
Epoch	3275:	cost=	0.008
Epoch	3276:	cost=	0.008
Epoch	3277:	cost=	0.008
Epoch	3278:	cost=	0.008
Epoch	3279:	cost=	0.008
Epoch	3280:	cost=	0.008
Epoch	3281:	cost=	0.008
Epoch	3282:	cost=	0.008
Epoch	3283:	cost=	0.008
Epoch	3284:	cost=	0.008
Epoch	3285:	cost=	0.008
Epoch	3286:	cost=	0.008
Epoch	3287:	cost=	0.008
Epoch	3288:	cost=	0.008
Epoch	3289:	cost=	0.008
Epoch	3290:	cost=	0.008
Epoch	3291:	cost=	0.008
Epoch	3292:	cost=	0.008
Epoch	3293:	cost=	0.008
Epoch	3294:	cost=	0.008
Epoch	3295:	cost=	0.008
Epoch	3296:	cost=	0.008
Epoch	3297:	cost=	0.008
Epoch	3298:	cost=	0.008
Epoch	3299:	cost=	0.008
Epoch	3300:	cost=	0.008
Epoch	3301:	cost=	0.008
Epoch	3302:	cost=	0.008
Epoch	3303:	cost=	0.008
Epoch	3304:	cost=	0.008
Epoch	3305:	cost=	0.008
Epoch	3306:	cost=	0.008
Epoch	3307:	cost=	0.008
Epoch	3308:	cost=	0.008
Epoch	3309:	cost=	0.008
Epoch	3310:	cost=	0.008
Epoch	3311:	cost=	0.008
Epoch	3312:	cost=	0.008
Epoch	3313:	cost=	0.008
Epoch	3314:	cost=	0.008
Epoch	3315:	cost=	0.008
Epoch	3316:	cost=	0.008
Epoch	3317:	cost=	0.008
Epoch	3318:	cost=	0.008
Epoch	3319:	cost=	0.008

Epoch	3320:	cost=	0.008
Epoch	3321:	cost=	0.008
Epoch	3322:	cost=	0.008
Epoch	3323:	cost=	0.008
Epoch	3324:	cost=	0.008
Epoch	3325:	cost=	0.008
Epoch	3326:	cost=	0.008
Epoch	3327:	cost=	0.008
Epoch	3328:	cost=	0.008
Epoch	3329:	cost=	0.008
Epoch	3330:	cost=	0.008
Epoch	3331:	cost=	0.008
Epoch	3332:	cost=	0.008
Epoch	3333:	cost=	0.008
Epoch	3334:	cost=	0.008
Epoch	3335:	cost=	0.008
Epoch	3336:	cost=	0.008
Epoch	3337:	cost=	0.008
Epoch	3338:	cost=	0.008
Epoch	3339:	cost=	0.008
Epoch	3340:	cost=	0.008
Epoch	3341:	cost=	0.008
Epoch	3342:	cost=	0.008
Epoch	3343:	cost=	0.008
Epoch	3344:	cost=	0.008
Epoch	3345:	cost=	0.008
Epoch	3346:	cost=	0.008
Epoch	3347:	cost=	0.008
Epoch	3348:	cost=	0.008
Epoch	3349:	cost=	0.008
Epoch	3350:	cost=	0.008
Epoch	3351:	cost=	0.008
Epoch	3352:	cost=	0.008
Epoch	3353:	cost=	0.008
Epoch	3354:	cost=	0.008
Epoch	3355:	cost=	0.008
Epoch	3356:	cost=	0.008
Epoch	3357:	cost=	0.008
Epoch	3358:	cost=	0.008
Epoch	3359:	cost=	0.008
Epoch	3360:	cost=	0.008
Epoch	3361:	cost=	0.008
Epoch	3362:	cost=	0.008
Epoch	3363:	cost=	0.008
Epoch	3364:	cost=	0.008
Epoch	3365:	cost=	0.008
Epoch	3366:	cost=	0.008
Epoch	3367:	cost=	0.008

Epoch	3368:	cost=	0.008
Epoch	3369:	cost=	0.008
Epoch	3370:	cost=	0.008
Epoch	3371:	cost=	0.008
Epoch	3372:	cost=	0.008
Epoch	3373:	cost=	0.008
Epoch	3374:	cost=	0.008
Epoch	3375:	cost=	0.008
Epoch	3376:	cost=	0.008
Epoch	3377:	cost=	0.008
Epoch	3378:	cost=	0.008
Epoch	3379:	cost=	0.008
Epoch	3380:	cost=	0.008
Epoch	3381:	cost=	0.008
Epoch	3382:	cost=	0.008
Epoch	3383:	cost=	0.008
Epoch	3384:	cost=	0.008
Epoch	3385:	cost=	0.008
Epoch	3386:	cost=	0.008
Epoch	3387:	cost=	0.008
Epoch	3388:	cost=	0.008
Epoch	3389:	cost=	0.008
Epoch	3390:	cost=	0.008
Epoch	3391:	cost=	0.008
Epoch	3392:	cost=	0.008
Epoch	3393:	cost=	0.008
Epoch	3394:	cost=	0.008
Epoch	3395:	cost=	0.008
Epoch	3396:	cost=	0.008
Epoch	3397:	cost=	0.009
Epoch	3398:	cost=	0.009
Epoch	3399:	cost=	0.010
Epoch	3400:	cost=	0.011
Epoch	3401:	cost=	0.011
Epoch	3402:	cost=	0.012
Epoch	3403:	cost=	0.011
Epoch	3404:	cost=	0.009
Epoch	3405:	cost=	0.008
Epoch	3406:	cost=	0.008
Epoch	3407:	cost=	0.009
Epoch	3408:	cost=	0.009
Epoch	3409:	cost=	0.009
Epoch	3410:	cost=	0.009
Epoch	3411:	cost=	0.008
Epoch	3412:	cost=	0.008
Epoch	3413:	cost=	0.009
Epoch	3414:	cost=	0.009
Epoch	3415:	cost=	0.009

Epoch	3416:	cost=	0.008
Epoch	3417:	cost=	0.008
Epoch	3418:	cost=	0.008
Epoch	3419:	cost=	0.008
Epoch	3420:	cost=	0.008
Epoch	3421:	cost=	0.008
Epoch	3422:	cost=	0.008
Epoch	3423:	cost=	0.008
Epoch	3424:	cost=	0.008
Epoch	3425:	cost=	0.008
Epoch	3426:	cost=	0.008
Epoch	3427:	cost=	0.008
Epoch	3428:	cost=	0.008
Epoch	3429:	cost=	0.008
Epoch	3430:	cost=	0.008
Epoch	3431:	cost=	0.008
Epoch	3432:	cost=	0.008
Epoch	3433:	cost=	0.008
Epoch	3434:	cost=	0.008
Epoch	3435:	cost=	0.008
Epoch	3436:	cost=	0.008
Epoch	3437:	cost=	0.008
Epoch	3438:	cost=	0.008
Epoch	3439:	cost=	0.008
Epoch	3440:	cost=	0.008
Epoch	3441:	cost=	0.008
Epoch	3442:	cost=	0.008
Epoch	3443:	cost=	0.008
Epoch	3444:	cost=	0.008
Epoch	3445:	cost=	0.008
Epoch	3446:	cost=	0.008
Epoch	3447:	cost=	0.008
Epoch	3448:	cost=	0.008
Epoch	3449:	cost=	0.008
Epoch	3450:	cost=	0.008
Epoch	3451:	cost=	0.008
Epoch	3452:	cost=	0.008
Epoch	3453:	cost=	0.008
Epoch	3454:	cost=	0.008
Epoch	3455:	cost=	0.008
Epoch	3456:	cost=	0.008
Epoch	3457:	cost=	0.008
Epoch	3458:	cost=	0.008
Epoch	3459:	cost=	0.008
Epoch	3460:	cost=	0.008
Epoch	3461:	cost=	0.008
Epoch	3462:	cost=	0.008
Epoch	3463:	cost=	0.008

Epoch	3464:	cost=	0.008
Epoch	3465:	cost=	0.008
Epoch	3466:	cost=	0.008
Epoch	3467:	cost=	0.008
Epoch	3468:	cost=	0.008
Epoch	3469:	cost=	0.008
Epoch	3470:	cost=	0.008
Epoch	3471:	cost=	0.008
Epoch	3472:	cost=	0.008
Epoch	3473:	cost=	0.008
Epoch	3474:	cost=	0.008
Epoch	3475:	cost=	0.008
Epoch	3476:	cost=	0.008
Epoch	3477:	cost=	0.008
Epoch	3478:	cost=	0.008
Epoch	3479:	cost=	0.008
Epoch	3480:	cost=	0.008
Epoch	3481:	cost=	0.008
Epoch	3482:	cost=	0.008
Epoch	3483:	cost=	0.008
Epoch	3484:	cost=	0.008
Epoch	3485:	cost=	0.008
Epoch	3486:	cost=	0.008
Epoch	3487:	cost=	0.008
Epoch	3488:	cost=	0.008
Epoch	3489:	cost=	0.008
Epoch	3490:	cost=	0.008
Epoch	3491:	cost=	0.008
Epoch	3492:	cost=	0.008
Epoch	3493:	cost=	0.008
Epoch	3494:	cost=	0.008
Epoch	3495:	cost=	0.008
Epoch	3496:	cost=	0.008
Epoch	3497:	cost=	0.008
Epoch	3498:	cost=	0.008
Epoch	3499:	cost=	0.008
Epoch	3500:	cost=	0.008
Epoch	3501:	cost=	0.008
Epoch	3502:	cost=	0.008
Epoch	3503:	cost=	0.008
Epoch	3504:	cost=	0.008
Epoch	3505:	cost=	0.008
Epoch	3506:	cost=	0.008
Epoch	3507:	cost=	0.008
Epoch	3508:	cost=	0.008
Epoch	3509:	cost=	0.008
Epoch	3510:	cost=	0.008
Epoch	3511:	cost=	0.008

Epoch	3512:	cost=	0.008
Epoch	3513:	cost=	0.008
Epoch	3514:	cost=	0.008
Epoch	3515:	cost=	0.008
Epoch	3516:	cost=	0.008
Epoch	3517:	cost=	0.008
Epoch	3518:	cost=	0.008
Epoch	3519:	cost=	0.008
Epoch	3520:	cost=	0.008
Epoch	3521:	cost=	0.008
Epoch	3522:	cost=	0.008
Epoch	3523:	cost=	0.008
Epoch	3524:	cost=	0.008
Epoch	3525:	cost=	0.008
Epoch	3526:	cost=	0.008
Epoch	3527:	cost=	0.008
Epoch	3528:	cost=	0.008
Epoch	3529:	cost=	0.008
Epoch	3530:	cost=	0.008
Epoch	3531:	cost=	0.008
Epoch	3532:	cost=	0.008
Epoch	3533:	cost=	0.008
Epoch	3534:	cost=	0.008
Epoch	3535:	cost=	0.008
Epoch	3536:	cost=	0.008
Epoch	3537:	cost=	0.008
Epoch	3538:	cost=	0.008
Epoch	3539:	cost=	0.008
Epoch	3540:	cost=	0.008
Epoch	3541:	cost=	0.008
Epoch	3542:	cost=	0.008
Epoch	3543:	cost=	0.008
Epoch	3544:	cost=	0.008
Epoch	3545:	cost=	0.008
Epoch	3546:	cost=	0.008
Epoch	3547:	cost=	0.008
Epoch	3548:	cost=	0.008
Epoch	3549:	cost=	0.008
Epoch	3550:	cost=	0.008
Epoch	3551:	cost=	0.008
Epoch	3552:	cost=	0.008
Epoch	3553:	cost=	0.008
Epoch	3554:	cost=	0.008
Epoch	3555:	cost=	0.008
Epoch	3556:	cost=	0.008
Epoch	3557:	cost=	0.008
Epoch	3558:	cost=	0.008
Epoch	3559:	cost=	0.008

Epoch	3560:	cost=	0.008
Epoch	3561:	cost=	0.008
Epoch	3562:	cost=	0.008
Epoch	3563:	cost=	0.008
Epoch	3564:	cost=	0.008
Epoch	3565:	cost=	0.008
Epoch	3566:	cost=	0.008
Epoch	3567:	cost=	0.008
Epoch	3568:	cost=	0.008
Epoch	3569:	cost=	0.008
Epoch	3570:	cost=	0.008
Epoch	3571:	cost=	0.008
Epoch	3572:	cost=	0.008
Epoch	3573:	cost=	0.008
Epoch	3574:	cost=	0.008
Epoch	3575:	cost=	0.008
Epoch	3576:	cost=	0.008
Epoch	3577:	cost=	0.008
Epoch	3578:	cost=	0.008
Epoch	3579:	cost=	0.008
Epoch	3580:	cost=	0.008
Epoch	3581:	cost=	0.008
Epoch	3582:	cost=	0.008
Epoch	3583:	cost=	0.008
Epoch	3584:	cost=	0.008
Epoch	3585:	cost=	0.008
Epoch	3586:	cost=	0.008
Epoch	3587:	cost=	0.008
Epoch	3588:	cost=	0.008
Epoch	3589:	cost=	0.008
Epoch	3590:	cost=	0.008
Epoch	3591:	cost=	0.008
Epoch	3592:	cost=	0.008
Epoch	3593:	cost=	0.008
Epoch	3594:	cost=	0.008
Epoch	3595:	cost=	0.008
Epoch	3596:	cost=	0.008
Epoch	3597:	cost=	0.008
Epoch	3598:	cost=	0.008
Epoch	3599:	cost=	0.008
Epoch	3600:	cost=	0.008
Epoch	3601:	cost=	0.008
Epoch	3602:	cost=	0.008
Epoch	3603:	cost=	0.008
Epoch	3604:	cost=	0.008
Epoch	3605:	cost=	0.008
Epoch	3606:	cost=	0.008
Epoch	3607:	cost=	0.008

Epoch	3608:	cost=	0.008
Epoch	3609:	cost=	0.008
Epoch	3610:	cost=	0.008
Epoch	3611:	cost=	0.008
Epoch	3612:	cost=	0.008
Epoch	3613:	cost=	0.008
Epoch	3614:	cost=	0.008
Epoch	3615:	cost=	0.008
Epoch	3616:	cost=	0.008
Epoch	3617:	cost=	0.008
Epoch	3618:	cost=	0.008
Epoch	3619:	cost=	0.008
Epoch	3620:	cost=	0.008
Epoch	3621:	cost=	0.008
Epoch	3622:	cost=	0.008
Epoch	3623:	cost=	0.008
Epoch	3624:	cost=	0.008
Epoch	3625:	cost=	0.008
Epoch	3626:	cost=	0.008
Epoch	3627:	cost=	0.008
Epoch	3628:	cost=	0.008
Epoch	3629:	cost=	0.008
Epoch	3630:	cost=	0.008
Epoch	3631:	cost=	0.008
Epoch	3632:	cost=	0.008
Epoch	3633:	cost=	0.008
Epoch	3634:	cost=	0.008
Epoch	3635:	cost=	0.008
Epoch	3636:	cost=	0.008
Epoch	3637:	cost=	0.008
Epoch	3638:	cost=	0.008
Epoch	3639:	cost=	0.008
Epoch	3640:	cost=	0.008
Epoch	3641:	cost=	0.008
Epoch	3642:	cost=	0.008
Epoch	3643:	cost=	0.008
Epoch	3644:	cost=	0.008
Epoch	3645:	cost=	0.008
Epoch	3646:	cost=	0.008
Epoch	3647:	cost=	0.008
Epoch	3648:	cost=	0.008
Epoch	3649:	cost=	0.008
Epoch	3650:	cost=	0.008
Epoch	3651:	cost=	0.008
Epoch	3652:	cost=	0.009
Epoch	3653:	cost=	0.009
Epoch	3654:	cost=	0.010
Epoch	3655:	cost=	0.012

Epoch	3656:	cost=	0.013
Epoch	3657:	cost=	0.012
Epoch	3658:	cost=	0.011
Epoch	3659:	cost=	0.009
Epoch	3660:	cost=	0.008
Epoch	3661:	cost=	0.009
Epoch	3662:	cost=	0.010
Epoch	3663:	cost=	0.010
Epoch	3664:	cost=	0.009
Epoch	3665:	cost=	0.008
Epoch	3666:	cost=	0.008
Epoch	3667:	cost=	0.009
Epoch	3668:	cost=	0.009
Epoch	3669:	cost=	0.008
Epoch	3670:	cost=	0.008
Epoch	3671:	cost=	0.008
Epoch	3672:	cost=	0.008
Epoch	3673:	cost=	0.009
Epoch	3674:	cost=	0.008
Epoch	3675:	cost=	0.008
Epoch	3676:	cost=	0.008
Epoch	3677:	cost=	0.008
Epoch	3678:	cost=	0.008
Epoch	3679:	cost=	0.008
Epoch	3680:	cost=	0.008
Epoch	3681:	cost=	0.008
Epoch	3682:	cost=	0.008
Epoch	3683:	cost=	0.008
Epoch	3684:	cost=	0.008
Epoch	3685:	cost=	0.008
Epoch	3686:	cost=	0.008
Epoch	3687:	cost=	0.008
Epoch	3688:	cost=	0.008
Epoch	3689:	cost=	0.008
Epoch	3690:	cost=	0.008
Epoch	3691:	cost=	0.008
Epoch	3692:	cost=	0.008
Epoch	3693:	cost=	0.008
Epoch	3694:	cost=	0.008
Epoch	3695:	cost=	0.008
Epoch	3696:	cost=	0.008
Epoch	3697:	cost=	0.008
Epoch	3698:	cost=	0.008
Epoch	3699:	cost=	0.008
Epoch	3700:	cost=	0.008
Epoch	3701:	cost=	0.008
Epoch	3702:	cost=	0.008
Epoch	3703:	cost=	0.008

```

Epoch 3704: cost= 0.008
Epoch 3705: cost= 0.008
Epoch 3706: cost= 0.008
Epoch 3707: cost= 0.008
Epoch 3708: cost= 0.008
Epoch 3709: cost= 0.008
Epoch 3710: cost= 0.008
Epoch 3711: cost= 0.008
Epoch 3712: cost= 0.008
Epoch 3713: cost= 0.008
Epoch 3714: cost= 0.008
Epoch 3715: cost= 0.008
Epoch 3716: cost= 0.008
Epoch 3717: cost= 0.008
Epoch 3718: cost= 0.008
Epoch 3719: cost= 0.008
Epoch 3720: cost= 0.008
Epoch 3721: cost= 0.008
Epoch 3722: cost= 0.008
Epoch 3723: cost= 0.008
Epoch 3724: cost= 0.008
Epoch 3725: cost= 0.008
Epoch 3726: cost= 0.008
Epoch 3727: cost= 0.008
Epoch 3728: cost= 0.008
Epoch 3729: cost= 0.008
Epoch 3730: cost= 0.008
Epoch 3731: cost= 0.008
Epoch 3732: cost= 0.008
Epoch 3733: cost= 0.008
Epoch 3734: cost= 0.008
Epoch 3735: cost= 0.008
Epoch 3736: cost= 0.008
best_cost=0.008

```

```

[ ]: # Experiment with different number of layers and activation functions. Here is
      # an example with three hidden layers (of sizes 4, 5, and 6) and ReLU
      ↪ activations.
      #
      # You can also plot the outputs of the hidden neurons in the first layer (using
      # the same code above).
      model = train1([4, 5, 6], nreps=50, phi=F.relu)
      nextplot()
      plot1(X1, y1, label="train")
      plot1(X1test, y1test, label="test")
      plot1fit(torch.linspace(0, 13, 500).unsqueeze(1), model)
      print("Training error:", F.mse_loss(y1, model(X1)).item())

```

```
print("Test error      :", F.mse_loss(y1test, model(X1test)).item())
```

```
X1 shape: torch.Size([100, 1])
```

```
Repetition  0:
```

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
```

```
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
```

```
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
```

```
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
        Current function value: 0.081754
```

```
        Iterations: 197
```

```
        Function evaluations: 330
```

```
        Gradient evaluations: 324
```

```
best_cost=0.082
```

```
X1 shape: torch.Size([100, 1])
```

```
Repetition  1:          Current function value: 0.309341
```

```
        Iterations: 47
```

```
        Function evaluations: 174
```

```
        Gradient evaluations: 166
```

```
best_cost=0.082
```

```
X1 shape: torch.Size([100, 1])
```

```
Repetition  2:          Current function value: 0.021929
```

```
        Iterations: 48
```

```
        Function evaluations: 133
```

```
        Gradient evaluations: 127
```

```
best_cost=0.022
```

```
X1 shape: torch.Size([100, 1])
```

```
Repetition  3:
```

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
```

```
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
        Current function value: 0.082625
```

```
        Iterations: 94
```

```
        Function evaluations: 176
```

```
        Gradient evaluations: 169
```

```
best_cost=0.022
```

```
X1 shape: torch.Size([100, 1])
```

```
Repetition  4:
```

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
```

```
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.089863
        Iterations: 120
        Function evaluations: 226
        Gradient evaluations: 220
best_cost=0.022
X1 shape: torch.Size([100, 1])
Repetition 5:          Current function value: 0.082474
        Iterations: 39
        Function evaluations: 129
        Gradient evaluations: 126
best_cost=0.022
X1 shape: torch.Size([100, 1])
Repetition 6:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.098277
        Iterations: 132
        Function evaluations: 207
        Gradient evaluations: 203
best_cost=0.022
X1 shape: torch.Size([100, 1])
Repetition 7:          Current function value: 0.098335
        Iterations: 99
        Function evaluations: 203
        Gradient evaluations: 198
best_cost=0.022
X1 shape: torch.Size([100, 1])
Repetition 8:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.084660
        Iterations: 63
        Function evaluations: 150
        Gradient evaluations: 143
best_cost=0.022
X1 shape: torch.Size([100, 1])
Repetition 9:

```



```

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.008690
        Iterations: 159
        Function evaluations: 265
        Gradient evaluations: 258
best_cost=0.009
X1 shape: torch.Size([100, 1])
Repetition 10:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.011249
        Iterations: 201
        Function evaluations: 311
        Gradient evaluations: 309
best_cost=0.009
X1 shape: torch.Size([100, 1])
Repetition 11:          Current function value: 0.356422
        Iterations: 75
        Function evaluations: 170
        Gradient evaluations: 166
best_cost=0.009
X1 shape: torch.Size([100, 1])
Repetition 12:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.356366
        Iterations: 63
        Function evaluations: 157
        Gradient evaluations: 152
best_cost=0.009
X1 shape: torch.Size([100, 1])
Repetition 13:          Current function value: 0.082331
        Iterations: 81
        Function evaluations: 196
        Gradient evaluations: 190

```

```

best_cost=0.009
X1 shape: torch.Size([100, 1])
Repetition 14:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.008332
        Iterations: 214
        Function evaluations: 393
        Gradient evaluations: 381
best_cost=0.008
X1 shape: torch.Size([100, 1])
Repetition 15:          Current function value: 0.356475
        Iterations: 82
        Function evaluations: 181
        Gradient evaluations: 178
best_cost=0.008
X1 shape: torch.Size([100, 1])
Repetition 16:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.011342
        Iterations: 188
        Function evaluations: 319
        Gradient evaluations: 307
best_cost=0.008
X1 shape: torch.Size([100, 1])
Repetition 17:          Current function value: 0.355779
        Iterations: 67
        Function evaluations: 164
        Gradient evaluations: 159
best_cost=0.008
X1 shape: torch.Size([100, 1])
Repetition 18:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.084557

```

```

        Iterations: 96
        Function evaluations: 230
        Gradient evaluations: 223
best_cost=0.008
X1 shape: torch.Size([100, 1])
Repetition 19:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.013663
        Iterations: 89
        Function evaluations: 205
        Gradient evaluations: 193
best_cost=0.008
X1 shape: torch.Size([100, 1])
Repetition 20:          Current function value: 0.355771
        Iterations: 97
        Function evaluations: 192
        Gradient evaluations: 185
best_cost=0.008
X1 shape: torch.Size([100, 1])
Repetition 21:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.008507
        Iterations: 199
        Function evaluations: 414
        Gradient evaluations: 402
best_cost=0.008
X1 shape: torch.Size([100, 1])
Repetition 22:          Current function value: 0.084680
        Iterations: 65
        Function evaluations: 149
        Gradient evaluations: 143
best_cost=0.008
X1 shape: torch.Size([100, 1])
Repetition 23: Optimization terminated successfully.
        Current function value: 0.506238
        Iterations: 2

```

```

        Function evaluations: 4
        Gradient evaluations: 4
best_cost=0.008
X1 shape: torch.Size([100, 1])
Repetition 24:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.355875
        Iterations: 83
        Function evaluations: 189
        Gradient evaluations: 182
best_cost=0.008
X1 shape: torch.Size([100, 1])
Repetition 25:          Current function value: 0.006384
        Iterations: 112
        Function evaluations: 216
        Gradient evaluations: 209
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 26:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.356134
        Iterations: 58
        Function evaluations: 164
        Gradient evaluations: 156
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 27:          Current function value: 0.357622
        Iterations: 36
        Function evaluations: 116
        Gradient evaluations: 110
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 28:          Current function value: 0.398953

```

```

        Iterations: 20
        Function evaluations: 115
        Gradient evaluations: 109
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 29:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.356162
        Iterations: 45
        Function evaluations: 156
        Gradient evaluations: 154
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 30:          Current function value: 0.355884
        Iterations: 79
        Function evaluations: 185
        Gradient evaluations: 181
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 31:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.098431
        Iterations: 51
        Function evaluations: 134
        Gradient evaluations: 128
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 32:          Current function value: 0.355989
        Iterations: 86
        Function evaluations: 181
        Gradient evaluations: 177
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 33:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:

```

```

OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.006600
        Iterations: 307
        Function evaluations: 433
        Gradient evaluations: 421
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 34:          Current function value: 0.357715
        Iterations: 28
        Function evaluations: 138
        Gradient evaluations: 131
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 35:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.355809
        Iterations: 94
        Function evaluations: 180
        Gradient evaluations: 176
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 36:          Current function value: 0.023239
        Iterations: 108
        Function evaluations: 223
        Gradient evaluations: 212
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 37:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.355985
        Iterations: 62

```

```

        Function evaluations: 169
        Gradient evaluations: 166
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 38:          Current function value: 0.082410
        Iterations: 46
        Function evaluations: 133
        Gradient evaluations: 123
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 39:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.008140
        Iterations: 86
        Function evaluations: 172
        Gradient evaluations: 165
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 40: Optimization terminated successfully.
        Current function value: 0.506238
        Iterations: 4
        Function evaluations: 6
        Gradient evaluations: 6
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 41:          Current function value: 0.351567
        Iterations: 14
        Function evaluations: 112
        Gradient evaluations: 101
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 42:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.011727
        Iterations: 96
        Function evaluations: 177
        Gradient evaluations: 172
best_cost=0.006
X1 shape: torch.Size([100, 1])

```

Repetition 43:

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:  
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
        Current function value: 0.008865
```

```
        Iterations: 152
```

```
        Function evaluations: 273
```

```
        Gradient evaluations: 266
```

```
best_cost=0.006
```

```
X1 shape: torch.Size([100, 1])
```

Repetition 44:

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:  
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:  
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
        Current function value: 0.355520
```

```
        Iterations: 96
```

```
        Function evaluations: 201
```

```
        Gradient evaluations: 196
```

```
best_cost=0.006
```

```
X1 shape: torch.Size([100, 1])
```

Repetition 45: Current function value: 0.356199

```
    Iterations: 74
```

```
    Function evaluations: 172
```

```
    Gradient evaluations: 166
```

```
best_cost=0.006
```

```
X1 shape: torch.Size([100, 1])
```

Repetition 46:

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:  
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:  
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
```

```
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

```
        Current function value: 0.084039
```

```
        Iterations: 66
```

```
        Function evaluations: 130
```

```
        Gradient evaluations: 125
```

```
best_cost=0.006
```

```
X1 shape: torch.Size([100, 1])
```

Repetition 47: Current function value: 0.311066

```
    Iterations: 28
```

```
    Function evaluations: 134
```



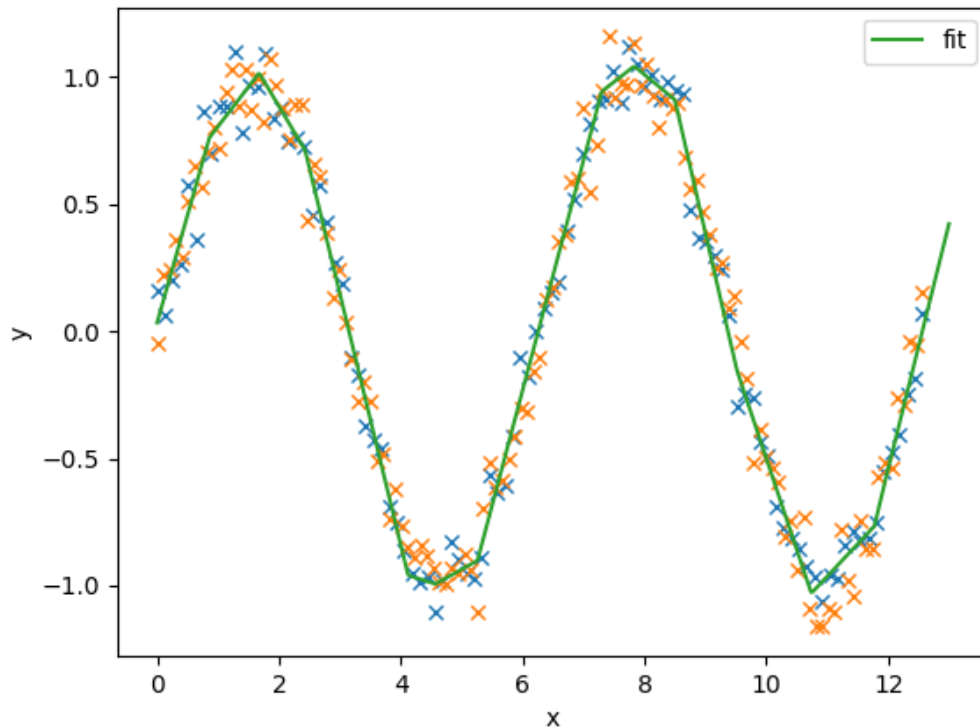
```

        Gradient evaluations: 130
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 48:

/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)

        Current function value: 0.013557
        Iterations: 162
        Function evaluations: 311
        Gradient evaluations: 302
best_cost=0.006
X1 shape: torch.Size([100, 1])
Repetition 49:          Current function value: 0.356307
        Iterations: 91
        Function evaluations: 212
        Gradient evaluations: 207
best_cost=0.006
Training error: 0.006383862346410751
Test error      : 0.010709108784794807

```



4 3 Backpropagation

```
[ ]: # Let's fit the model with one hidden layer consisting of 50 units.
model = train1([50], nreps=1)
print("Training error:", F.mse_loss(y1, model(X1)).item())
print("Test error      :", F.mse_loss(y1test, model(X1test)).item())
```

```
# Extract parameters
pars = dict(model.named_parameters())
W1 = pars["0_weight"].data # 1x50
b1 = pars["0_bias"].data # 50
W2 = pars["1_weight"].data # 50x1
b2 = pars["1_bias"].data # 1
```

```
X1 shape: torch.Size([100, 1])
Repetition 0:          Current function value: 0.004123
      Iterations: 4749
      Function evaluations: 5216
      Gradient evaluations: 5208
best_cost=0.004
```

Training error: 0.004123299848288298

Test error : 0.5570471286773682

```
/opt/homebrew/lib/python3.11/site-packages/scipy/optimize/_minimize.py:705:
OptimizeWarning: Desired error not necessarily achieved due to precision loss.
    res = _minimize_bfgs(fun, x0, args, jac, callback, **options)
```

4.1 3a Forward pass

```
[ ]: # Compute results of forward pass on an example x (i.e., z1, z2, z3, z4, yhat, l) using Pytorch
      x = X1test[1, :]
      y = y1test[1, :]
      print(f"x={x}, y={y}, yhat={model(x).detach()}, l={torch.nn.
      ↪MSELoss()(y,model(x))}")
```

```
x=tensor([0.1030]), y=tensor([0.2253]), yhat=tensor([0.0573]),
l=0.02819862589240074
```

```
[ ]: # Now do this by hand (including all intermediate values). You should get the
      ↪same
      # results as above.

      def compute_loss(y, y_hat):
          return torch.square(y - y_hat)

      # Compute the forward pass
      def forward_pass(x, y, W1, b1, W2, b2):
          z1 = torch.matmul(x, W1)
          z2 = z1 + b1
          z3 = torch.sigmoid(z2)
          z4 = torch.matmul(z3, W2)
          y_hat = z4 + b2

          loss = compute_loss(y, y_hat)
          return z1, z2, z3, z4, y_hat, loss
```

```
[ ]: z1, z2, z3, z4, y_hat, loss = forward_pass(x, y, W1, b1, W2, b2)
      print(f"x={x}, y={y}, yhat={y_hat}, l={loss}")
```

```
x=tensor([0.1030]), y=tensor([0.2253]), yhat=tensor([0.0573]),
l=tensor([0.0282])
```

4.2 3b Backward pass

```
[ ]: # Define the sigmoid derivative function
      def sigmoid_derivative(z):
          return torch.sigmoid(z) * (1 - torch.sigmoid(z))
```

```

# Compute the backward pass
def backward_pass(x, y, y_hat, z1, z2, z3, z4, W1, W2):
    delta_l = 1
    delta_y = 2 * (y - y_hat)
    delta_yhat = -delta_y
    delta_b2 = delta_yhat
    delta_z4 = delta_yhat
    delta_W2 = torch.matmul(z3.view(-1, 1), delta_z4.view(1, -1))
    delta_z3 = torch.matmul(delta_z4, W2.t())
    delta_z2 = delta_z3 * sigmoid_derivative(z2)
    delta_b1 = delta_z2
    delta_z1 = delta_z2
    delta_W1 = torch.matmul(x.view(-1, 1), delta_z1.view(1, -1))
    delta_x = torch.matmul(delta_z1, W1.t())
    return delta_x, delta_W1, delta_z1, delta_b1, delta_z2, delta_z3, delta_W2,
    delta_z4, delta_b2, delta_yhat, delta_l, delta_y

# Compute the forward pass
z1, z2, z3, z4, y_hat, loss = forward_pass(x, y, W1, b1, W2, b2)

# Compute the backward pass
delta_x, delta_W1, delta_z1, delta_b1, delta_z2, delta_z3, delta_W2, delta_z4,
delta_b2, delta_yhat, delta_l, delta_y = backward_pass(x, y, y_hat, z1, z2,
z3, z4, W1, W2)

# Print the results
print(f"delta_x={delta_x}, delta_W1={delta_W1}, delta_z1={delta_z1},
delta_b1={delta_b1}, delta_z2={delta_z2}, delta_z3={delta_z3},
delta_W2={delta_W2}, delta_z4={delta_z4}, delta_b2={delta_b2},
delta_yhat={delta_yhat}, delta_l={delta_l}, delta_y={delta_y}")

```

```

delta_x=tensor([-0.0546], grad_fn=<SqueezeBackward4>),
delta_W1=tensor([[[-1.2498e-20, -3.3407e-11, -7.5456e-05, -3.1627e-03,
6.6566e-07,
    0.0000e+00, -6.9054e-09, -1.8054e-02,  5.9882e-04, -1.1690e-04,
    -2.6779e-22, -9.5940e-25, -1.5367e-04,  9.9927e-05,  0.0000e+00,
    2.3231e-25,  2.5774e-05,  1.9602e-04, -3.6733e-03, -1.2149e-03,
    -4.7099e-03,  6.3735e-05,  0.0000e+00, -8.7016e-03,  6.0340e-05,
    0.0000e+00,  0.0000e+00, -8.7417e-08, -2.9660e-05, -4.9669e-03,
    -8.8448e-07, -2.5948e-06,  0.0000e+00,  0.0000e+00, -3.1340e-02,
    4.9667e-03, -3.2181e-04, -2.4263e-02,  8.8048e-08,  8.4567e-06,
    1.8478e-11, -9.2940e-06, -3.9415e-02, -2.5835e-15,  0.0000e+00,
    -9.7880e-06, -6.3194e-03,  0.0000e+00, -1.5671e-05, -1.5880e-03]]],
    grad_fn=<MmBackward0>), delta_z1=tensor([-1.2134e-19, -3.2433e-10,
-7.3256e-04, -3.0705e-02,  6.4626e-06,
    -0.0000e+00, -6.7041e-08, -1.7528e-01,  5.8137e-03, -1.1349e-03,
    -2.5998e-21, -9.3143e-24, -1.4919e-03,  9.7014e-04,  0.0000e+00,

```

```

    2.2554e-24,  2.5023e-04,  1.9031e-03, -3.5662e-02, -1.1795e-02,
    -4.5726e-02,  6.1877e-04, -0.0000e+00, -8.4479e-02,  5.8581e-04,
    -0.0000e+00,  0.0000e+00, -8.4869e-07, -2.8796e-04, -4.8221e-02,
    -8.5869e-06, -2.5191e-05, -0.0000e+00,  0.0000e+00, -3.0426e-01,
    4.8219e-02, -3.1243e-03, -2.3556e-01,  8.5481e-07,  8.2101e-05,
    1.7939e-10, -9.0230e-05, -3.8265e-01, -2.5082e-14, -0.0000e+00,
    -9.5027e-05, -6.1352e-02,  0.0000e+00, -1.5214e-04, -1.5417e-02],
    grad_fn=<MulBackward0>), delta_b1=tensor([-1.2134e-19, -3.2433e-10,
-7.3256e-04, -3.0705e-02,  6.4626e-06,
    -0.0000e+00, -6.7041e-08, -1.7528e-01,  5.8137e-03, -1.1349e-03,
    -2.5998e-21, -9.3143e-24, -1.4919e-03,  9.7014e-04,  0.0000e+00,
    2.2554e-24,  2.5023e-04,  1.9031e-03, -3.5662e-02, -1.1795e-02,
    -4.5726e-02,  6.1877e-04, -0.0000e+00, -8.4479e-02,  5.8581e-04,
    -0.0000e+00,  0.0000e+00, -8.4869e-07, -2.8796e-04, -4.8221e-02,
    -8.5869e-06, -2.5191e-05, -0.0000e+00,  0.0000e+00, -3.0426e-01,
    4.8219e-02, -3.1243e-03, -2.3556e-01,  8.5481e-07,  8.2101e-05,
    1.7939e-10, -9.0230e-05, -3.8265e-01, -2.5082e-14, -0.0000e+00,
    -9.5027e-05, -6.1352e-02,  0.0000e+00, -1.5214e-04, -1.5417e-02],
    grad_fn=<MulBackward0>), delta_z2=tensor([-1.2134e-19, -3.2433e-10,
-7.3256e-04, -3.0705e-02,  6.4626e-06,
    -0.0000e+00, -6.7041e-08, -1.7528e-01,  5.8137e-03, -1.1349e-03,
    -2.5998e-21, -9.3143e-24, -1.4919e-03,  9.7014e-04,  0.0000e+00,
    2.2554e-24,  2.5023e-04,  1.9031e-03, -3.5662e-02, -1.1795e-02,
    -4.5726e-02,  6.1877e-04, -0.0000e+00, -8.4479e-02,  5.8581e-04,
    -0.0000e+00,  0.0000e+00, -8.4869e-07, -2.8796e-04, -4.8221e-02,
    -8.5869e-06, -2.5191e-05, -0.0000e+00,  0.0000e+00, -3.0426e-01,
    4.8219e-02, -3.1243e-03, -2.3556e-01,  8.5481e-07,  8.2101e-05,
    1.7939e-10, -9.0230e-05, -3.8265e-01, -2.5082e-14, -0.0000e+00,
    -9.5027e-05, -6.1352e-02,  0.0000e+00, -1.5214e-04, -1.5417e-02],
    grad_fn=<MulBackward0>), delta_z3=tensor([ -5.7655,  -9.7092,  -0.2775,
-0.7629,   3.5219,  -4.6906,  -3.8438,
    -6.9920,  10.6412,  -2.2282, -11.9009, -17.5671,  -5.8399,   8.3463,
    40.9113,   8.0653,   0.7241,   0.1432,  -0.9021,  -2.3441,  -1.0778,
    7.0700,  -0.3174,  -1.1393,   0.5834,  -1.9495,   3.8695,  -7.1193,
    -3.5422,  -2.3220, -11.8693,  -2.1132,  -5.8690,   3.8300,  -2.1714,
    0.5442,  -1.1922,  -1.4804,   4.8671,   0.0427,  11.6495,  -0.6808,
    -3.1774,  -4.1564, -13.2173,  -3.6154,  -1.5305,   1.9429,  -0.9462,
    -1.7004], grad_fn=<SqueezeBackward4>), delta_W2=tensor([[ -7.0680e-21],
[-1.1219e-11],
[-3.3496e-01],
[-1.4111e-02],
[-6.1627e-07],
[-3.3585e-01],
[-5.8577e-09],
[-8.6417e-03],
[-3.3567e-01],
[-3.3568e-01],
[-7.3368e-23],

```

```

[-1.7807e-25],
[-3.3576e-01],
[-3.9042e-05],
[-3.3585e-01],
[-9.3917e-26],
[-3.3573e-01],
[-4.5245e-03],
[-1.3848e-02],
[-3.3415e-01],
[-3.2094e-01],
[-2.9396e-05],
[-3.3585e-01],
[-2.7087e-02],
[-3.3760e-04],
[-3.3585e-01],
[-3.3585e-01],
[-3.3585e-01],
[-3.3582e-01],
[-3.2872e-01],
[-2.4297e-07],
[-3.3584e-01],
[-3.3585e-01],
[-3.3585e-01],
[-5.6596e-02],
[-3.0285e-01],
[-3.3497e-01],
[-6.6676e-02],
[-5.8985e-08],
[-6.4641e-04],
[-5.1718e-12],
[-3.3580e-01],
[-4.7032e-02],
[-2.0267e-15],
[-3.3585e-01],
[-8.8276e-06],
[-3.2180e-01],
[-3.3585e-01],
[-3.3579e-01],
[-3.3278e-01]], grad_fn=<MmBackward0>), delta_z4=tensor([-0.3358],
grad_fn=<NegBackward0>), delta_b2=tensor([-0.3358], grad_fn=<NegBackward0>),
delta_yhat=tensor([-0.3358], grad_fn=<NegBackward0>), delta_l=1,
delta_y=tensor([0.3358], grad_fn=<MulBackward0>))

```

```

[ ]: # Use PyTorch's backprop
x.requires_grad = True
y.requires_grad = True
if x.grad is not None:

```

```

    x.grad.zero_()
    if y.grad is not None:
        y.grad.zero_()
    model.zero_grad()
    t_yhat = model(x)
    t_yhat.retain_grad()
    t_l = torch.nn.MSELoss()(t_yhat, y)
    t_l.backward()
    t_delta_l = 1
    t_delta_y = y.grad
    t_delta_yhat = t_yhat.grad
    t_delta_b2 = model.get_parameter("l_bias").grad
    t_delta_W2 = model.get_parameter("l_weight").grad
    t_delta_b1 = model.get_parameter("0_bias").grad
    t_delta_W1 = model.get_parameter("0_weight").grad
    t_delta_x = x.grad

```

```

[ ]: # Check if equal (show squared error)
for v in ["y", "yhat", "b2", "W2", "b1", "W1", "x"]:
    print(f'{v}, squared error={torch.
    ↪sum((eval("t_delta_"+v)-eval("delta_"+v))*2)}')

```

```

y, squared error=0.0
yhat, squared error=0.0
b2, squared error=0.0
W2, squared error=0.0
b1, squared error=2.542219572231237e-16
W1, squared error=3.809587755738482e-18
x, squared error=1.4210854715202004e-14

```

```

[ ]: # Check if equal (show actual values)
for v in ["l", "y", "yhat", "b2", "W2", "b1", "W1", "x"]:
    print(f'{v}, pytorch={eval("t_delta_"+v)}, you={eval("delta_"+v)}')

```

```

l, pytorch=1, you=1
y, pytorch=tensor([0.3358]), you=tensor([0.3358])
yhat, pytorch=tensor([-0.3358]), you=tensor([-0.3358])
b2, pytorch=tensor([-0.3358]), you=tensor([-0.3358])
W2, pytorch=tensor([[-7.0680e-21],
    [-1.1219e-11],
    [-3.3496e-01],
    [-1.4111e-02],
    [-6.1627e-07],
    [-3.3585e-01],
    [-5.8577e-09],
    [-8.6417e-03],
    [-3.3567e-01],
    [-3.3568e-01],

```

```

[-7.3368e-23],
[-1.7807e-25],
[-3.3576e-01],
[-3.9042e-05],
[-3.3585e-01],
[-9.3917e-26],
[-3.3573e-01],
[-4.5245e-03],
[-1.3848e-02],
[-3.3415e-01],
[-3.2094e-01],
[-2.9396e-05],
[-3.3585e-01],
[-2.7087e-02],
[-3.3760e-04],
[-3.3585e-01],
[-3.3585e-01],
[-3.3585e-01],
[-3.3582e-01],
[-3.2872e-01],
[-2.4297e-07],
[-3.3584e-01],
[-3.3585e-01],
[-3.3585e-01],
[-5.6596e-02],
[-3.0285e-01],
[-3.3497e-01],
[-6.6676e-02],
[-5.8985e-08],
[-6.4641e-04],
[-5.1718e-12],
[-3.3580e-01],
[-4.7032e-02],
[-2.0267e-15],
[-3.3585e-01],
[-8.8276e-06],
[-3.2180e-01],
[-3.3585e-01],
[-3.3579e-01],
[-3.3278e-01]], you=tensor([[ -7.0680e-21],
[-1.1219e-11],
[-3.3496e-01],
[-1.4111e-02],
[-6.1627e-07],
[-3.3585e-01],
[-5.8577e-09],
[-8.6417e-03],
[-3.3567e-01],

```



```

[-3.3568e-01],
[-7.3368e-23],
[-1.7807e-25],
[-3.3576e-01],
[-3.9042e-05],
[-3.3585e-01],
[-9.3917e-26],
[-3.3573e-01],
[-4.5245e-03],
[-1.3848e-02],
[-3.3415e-01],
[-3.2094e-01],
[-2.9396e-05],
[-3.3585e-01],
[-2.7087e-02],
[-3.3760e-04],
[-3.3585e-01],
[-3.3585e-01],
[-3.3585e-01],
[-3.3582e-01],
[-3.2872e-01],
[-2.4297e-07],
[-3.3584e-01],
[-3.3585e-01],
[-3.3585e-01],
[-5.6596e-02],
[-3.0285e-01],
[-3.3497e-01],
[-6.6676e-02],
[-5.8985e-08],
[-6.4641e-04],
[-5.1718e-12],
[-3.3580e-01],
[-4.7032e-02],
[-2.0267e-15],
[-3.3585e-01],
[-8.8276e-06],
[-3.2180e-01],
[-3.3585e-01],
[-3.3579e-01],
[-3.3278e-01]])
b1, pytorch=tensor([-1.2134e-19, -3.2433e-10, -7.3256e-04, -3.0705e-02,
6.4626e-06,
-0.0000e+00, -6.7041e-08, -1.7528e-01, 5.8137e-03, -1.1349e-03,
-2.5998e-21, -9.3143e-24, -1.4919e-03, 9.7014e-04, 0.0000e+00,
2.2554e-24, 2.5023e-04, 1.9031e-03, -3.5662e-02, -1.1795e-02,
-4.5726e-02, 6.1877e-04, -0.0000e+00, -8.4479e-02, 5.8581e-04,
-0.0000e+00, 0.0000e+00, -8.4869e-07, -2.8796e-04, -4.8221e-02,

```

```

-8.5869e-06, -2.5191e-05, -0.0000e+00, 0.0000e+00, -3.0426e-01,
4.8219e-02, -3.1243e-03, -2.3556e-01, 8.5481e-07, 8.2101e-05,
1.7939e-10, -9.0230e-05, -3.8265e-01, -2.5082e-14, -0.0000e+00,
-9.5027e-05, -6.1352e-02, 0.0000e+00, -1.5214e-04, -1.5417e-02]],
you=tensor([-1.2134e-19, -3.2433e-10, -7.3256e-04, -3.0705e-02, 6.4626e-06,
-0.0000e+00, -6.7041e-08, -1.7528e-01, 5.8137e-03, -1.1349e-03,
-2.5998e-21, -9.3143e-24, -1.4919e-03, 9.7014e-04, 0.0000e+00,
2.2554e-24, 2.5023e-04, 1.9031e-03, -3.5662e-02, -1.1795e-02,
-4.5726e-02, 6.1877e-04, -0.0000e+00, -8.4479e-02, 5.8581e-04,
-0.0000e+00, 0.0000e+00, -8.4869e-07, -2.8796e-04, -4.8221e-02,
-8.5869e-06, -2.5191e-05, -0.0000e+00, 0.0000e+00, -3.0426e-01,
4.8219e-02, -3.1243e-03, -2.3556e-01, 8.5481e-07, 8.2101e-05,
1.7939e-10, -9.0230e-05, -3.8265e-01, -2.5082e-14, -0.0000e+00,
-9.5027e-05, -6.1352e-02, 0.0000e+00, -1.5214e-04, -1.5417e-02])
W1, pytorch=tensor([[ -1.2498e-20, -3.3407e-11, -7.5456e-05, -3.1627e-03,
6.6566e-07,
0.0000e+00, -6.9054e-09, -1.8054e-02, 5.9882e-04, -1.1690e-04,
-2.6779e-22, -9.5940e-25, -1.5367e-04, 9.9927e-05, 0.0000e+00,
2.3231e-25, 2.5774e-05, 1.9602e-04, -3.6733e-03, -1.2149e-03,
-4.7099e-03, 6.3735e-05, 0.0000e+00, -8.7016e-03, 6.0340e-05,
0.0000e+00, 0.0000e+00, -8.7417e-08, -2.9660e-05, -4.9669e-03,
-8.8448e-07, -2.5948e-06, 0.0000e+00, 0.0000e+00, -3.1340e-02,
4.9667e-03, -3.2181e-04, -2.4263e-02, 8.8048e-08, 8.4567e-06,
1.8478e-11, -9.2940e-06, -3.9415e-02, -2.5835e-15, 0.0000e+00,
-9.7880e-06, -6.3194e-03, 0.0000e+00, -1.5671e-05, -1.5880e-03]]),
you=tensor([[ -1.2498e-20, -3.3407e-11, -7.5456e-05, -3.1627e-03, 6.6566e-07,
0.0000e+00, -6.9054e-09, -1.8054e-02, 5.9882e-04, -1.1690e-04,
-2.6779e-22, -9.5940e-25, -1.5367e-04, 9.9927e-05, 0.0000e+00,
2.3231e-25, 2.5774e-05, 1.9602e-04, -3.6733e-03, -1.2149e-03,
-4.7099e-03, 6.3735e-05, 0.0000e+00, -8.7016e-03, 6.0340e-05,
0.0000e+00, 0.0000e+00, -8.7417e-08, -2.9660e-05, -4.9669e-03,
-8.8448e-07, -2.5948e-06, 0.0000e+00, 0.0000e+00, -3.1340e-02,
4.9667e-03, -3.2181e-04, -2.4263e-02, 8.8048e-08, 8.4567e-06,
1.8478e-11, -9.2940e-06, -3.9415e-02, -2.5835e-15, 0.0000e+00,
-9.7880e-06, -6.3194e-03, 0.0000e+00, -1.5671e-05, -1.5880e-03]))
x, pytorch=tensor([-0.0546]), you=tensor([-0.0546])

```