LinearMDP Likelihood Testing

MDP parameters = {'n':2, 'b':1, 'determinism':1.0, 'discount':0.99, 'seed': 0}

Same paths and true R

Matlab Reward Function Output

```
True R
   0.000528330112439   0.000528330112439   0.000528330112439   0.000528330112439   0.000528330112439
   0.000000002399272   0.000000002399272   0.000000002399272   0.000000002399272   0.000000002399272
   4.510928037168985   4.510928037168985   4.510928037168985   4.510928037168985   4.510928037168985
   4.533927110477543   4.533927110477543   4.533927110477543   4.533927110477543   4.533927110477543

True Likelihood
     1.043640655503507e+04   = 10436.64

Optimal Policy
     2
     2
     3
     1

Found R
  -2.282269456199177  -2.282269456199177  -2.282269456199177  -2.282269456199177  -2.282269456199177
  -1.630248097698442  -1.630248097698442  -1.630248097698442  -1.630248097698442  -1.630248097698442
   2.853838697068992   2.853838697068992   2.853838697068992   2.853838697068992   2.853838697068992
   2.874558020258835   2.874558020258835   2.874558020258835   2.874558020258835   2.874558020258835

Found Likelihood
     1.043568101290607e+04  = 10435.68

Optimal Policy
     2
     2      = [1,1,2,0] in python
     3      cos index diff
     1
```

Numpy Reward Function Output

```
True R is
[[0.0005 0.0005 0.0005 0.0005 0.0005]
 [0.     0.     0.     0.     0.    ]
 [4.5109 4.5109 4.5109 4.5109 4.5109]
 [4.5339 4.5339 4.5339 4.5339 4.5339]]
 with negated likelihood of 10436.4065555
 and optimal policy [1 1 2 0]

Found R is
[[-2.73117114 -2.73117114 -2.73117114 -2.73117114 -2.73117114]
 [-2.05036447 -2.05036447 -2.05036447 -2.05036447 -2.05036447]
 [ 2.43509587  2.43509587  2.43509587  2.43509587  2.43509587]
 [ 2.45579707  2.45579707  2.45579707  2.45579707  2.45579707]]
 with negated likelihood of 10435.6824895
 and optimal policy [1 1 2 0]
```

PyTorch Reward Function Output

```
True R is
tensor([[5.000e-04, 5.000e-04, 5.000e-04, 5.000e-04, 5.000e-04],
        [0.000e+00, 0.000e+00, 0.000e+00, 0.000e+00, 0.000e+00],
        [4.511e+00, 4.511e+00, 4.511e+00, 4.511e+00, 4.511e+00],
        [4.534e+00, 4.534e+00, 4.534e+00, 4.534e+00, 4.534e+00]])
 with negated likelihood of 10436.40655548196
 and optimal policy [1 1 2 0]


Found R is
tensor([[-3.024, -3.024, -3.024, -3.024, -3.024],
        [-2.375, -2.375, -2.375, -2.375, -2.375],
        [ 2.118,  2.118,  2.118,  2.118,  2.118],
        [ 2.139,  2.139,  2.139,  2.139,  2.139]], dtype=torch.float64)
 with negated likelihood of 10435.681479438554
 and optimal policy [1 1 2 0]
```

PyTorch with custom sampled paths using linear NN w/out backward function

```
EstR: tensor([[-1.620, -1.458,  3.217,  3.242]], grad_fn=<AddmmBackward>) | loss: 10436.089196453906 | EVD: 0.00283340380700825
```

Matlab MDP Solution Output

```
P
   0.002880150863316   0.988350220895028   0.003009326515067   0.002880150863316   0.002880150863316
   0.002877354167903   0.988614094053452   0.002877354167903   0.002877354167903   0.002753843442797
   0.246970950890849   0.246970950890849   0.258367449408286   0.000719697919203   0.246970950890849
   0.252601744935846   0.252601744935846   0.252601744935846   0.000735195551007   0.241459569641488


Q
   1.0e+02 *

   5.791843647590228   5.850225591926778   5.792282383877290   5.791843647590228   5.791843647590228
   5.792277100600159   5.850671429371738   5.792277100600159   5.792277100600159   5.791838364313096
   5.895329588997344   5.895329588997344   5.895780709719435   5.836947644660794   5.895329588997344
   5.896010700452520   5.896010700452520   5.896010700452520   5.837616371680941   5.895559579730429


V
   1.0e+02 *

   5.850342773621333
   5.850785941588063
   5.909314434567343
   5.909770112064405
```

Python MDP Solution Output

```
P
 [[ 0.00288015  0.98835014  0.00300941  0.00288015  0.00288015]
 [ 0.00287743  0.98861386  0.00287743  0.00287743  0.00275384]
 [ 0.24697086  0.24697086  0.25836772  0.0007197   0.24697086]
 [ 0.25260182  0.25260182  0.25260182  0.00073522  0.24145931]]
Q
 [[ 579.18159787  585.01979289  579.22550107  579.18159787  579.18159787]
 [ 579.22500107  585.06440635  579.22500107  579.22500107  579.18109787]
 [ 589.53019291  589.53019291  589.57530637  583.69199789  589.53019291]
 [ 589.59830614  589.59830614  589.59830614  583.75890085  589.55319267]]
V
 [[ 585.03151114]
 [ 585.07585781]
 [ 590.92867782]
 [ 590.97424698]]
```

Scaling R and recalculating with same paths in Python:

```
True R is
[[ 0.82299711  0.82299711  0.82299711  0.82299711  0.82299711]
 [ 6.84492019  6.84492019  6.84492019  6.84492019  6.84492019]
 [ 1.74250829  1.74250829  1.74250829  1.74250829  1.74250829]
 [ 0.77701112  0.77701112  0.77701112  0.77701112  0.77701112]]
 with negated likelihood of 7926.74335351
 and optimal policy [2 0 3 3]

 ... Doubling R & recalcuating ...


Double R is
[[  1.64599423   1.64599423   1.64599423   1.64599423   1.64599423]
 [ 13.68984039  13.68984039  13.68984039  13.68984039  13.68984039]
 [  3.48501658   3.48501658   3.48501658   3.48501658   3.48501658]
 [  1.55402224   1.55402224   1.55402224   1.55402224   1.55402224]]
 with negated likelihood of 7926.74335351
 and optimal policy [2 0 3 3]

 ... Quadrupling R & recalculating ...


Quadrupled R is
[[  3.29198845   3.29198845   3.29198845   3.29198845   3.29198845]
 [ 27.37968078  27.37968078  27.37968078  27.37968078  27.37968078]
 [  6.97003317   6.97003317   6.97003317   6.97003317   6.97003317]
 [  3.10804448   3.10804448   3.10804448   3.10804448   3.10804448]]
 with negated likelihood of 7926.74335351
 and optimal policy [2 0 3 3]
```