Solving MDPs using the MDP package in R (exercises)

Lars Relund Nielsen*

Department of Genetics and Biotechnology, University of Aarhus, P.O. Box 50, DK-8830 Tjele, Denmark, lars@relund.dk.

Anders Ringgaard Kristensen

Department of Large Animal Sciences, University of Copenhagen, Groennegaardsvej 2, DK-1870 Frederiksberg C, Denmark.

July 26, 2009

Solutions for Exercise 1

1. Create a folder for your exercises on your computer.
2. Download the data files needed for the exercises at http://www.research.relund.dk/?p=150 and extract the files in the folder you just have created.
3. Open R GUI and set the working directory to the folder you just created (File->Change dir...).
4. Load the MDP package.

```
> library(MDP)
```

5. Check that you have version 1.0 of the MDP package installed (?MDP - the version number appear in the bottom).
6. Consider the sow replacement problem in [2]. Generate the MDP in R. Remember that you can copy text from a pdf file if you read it using Acrobat reader, so you do not have to type in the code fragments yourself.

```
> prefix="sow_"
> w<-binaryMDPWriter(prefix)
> w$setWeights(c("Duration", "Net reward", "Piglets"))
> w$process()
>   w$stage()
>     w$state(label="Small litter")
```

*Corresponding author

```
>       w$action(label="Keep",weights=c(1,10000,8),prob=c(1,0,0.6, 1,1,0.3,
1,2,0.1))
>       w$endAction()
>       w$action(label="Replace",weights=c(1,9000,8),prob=c(1,0,1/3, 1,1,1/3,
1,2,1/3))
>       w$endAction()
>     w$endState()
>     w$state(label="Average litter")
>       w$action(label="Keep",weights=c(1,12000,11),prob=c(1,0,0.2, 1,1,0.6,
1,2,0.2))
>       w$endAction()
>       w$action(label="Replace",weights=c(1,11000,11),prob=c(1,0,1/3, 1,1,1/3,
1,2,1/3))
>       w$endAction()
>     w$endState()
>     w$state(label="Big litter")
>       w$action(label="Keep",weights=c(1,14000,14),prob=c(1,0,0.1, 1,1,0.3,
1,2,0.6))
>       w$endAction()
>       w$action(label="Replace",weights=c(1,13000,14),prob=c(1,0,1/3, 1,1,1/3,
1,2,1/3))
>       w$endAction()
>     w$endState()
>   w$endStage()
> w$endProcess()
> w$closeWriter()
```

```
Statistics:
  states : 3
  actions: 6
  weights: 3

Closing binary MDP writer.
```

7. Have a look at the data frames with information about the states and actions. What is the transition probability for a low litter size given a big litter size in the current stage and action keep?

```
> stateIdxDf(prefix)      # states of the MDP with labels returned as a data frame
```

```
  sId n0 s0          label
1   0  0  0   Small litter
2   1  0  1 Average litter
3   2  0  2     Big litter
```

```
> actionInfo(prefix)      # all action information of the MDP returned in a single
data frame
```

```
  aId sId Duration Net reward Piglets scp0 idx0       pr0 scp1 idx1       pr1 scp2 idx2       pr2   label
1   0   0        1      10000       8    1    0 0.6000000    1    1 0.3000000    1    2 0.1000000    Keep
2   1   0        1       9000       8    1    0 0.3333333    1    1 0.3333333    1    2 0.3333333 Replace
3   2   1        1      12000      11    1    0 0.2000000    1    1 0.6000000    1    2 0.2000000    Keep
4   3   1        1      11000      11    1    0 0.3333333    1    1 0.3333333    1    2 0.3333333 Replace
5   4   2        1      14000      14    1    0 0.1000000    1    1 0.3000000    1    2 0.6000000    Keep
6   5   2        1      13000      14    1    0 0.3333333    1    1 0.3333333    1    2 0.3333333 Replace
```

Since state `big litter` have an sId=2 and we consider the `keep` action the action have aId=4. The transition probability to `low litter` is now given in the `pr0` column and equals 0.1.

8. Find the optimal policy under the expected discounted reward criterion using an interest rate of 5%. What is the interpretation of the weights?

   Using policy iteration we get

```
> mdp<-loadMDP(prefix)
```

```
Cpu for reading the binary files: 0s
Cpu time for checking MDP 0s.
Cpu time for building state-expanded hypergraph 0s
```

```
> iW<-1                    # index of the weight we want to optimize
> iDur<-0                  # index of the duration/time
> rate<-0.05               # discount rate
> rateBase<-1              # rate base
> policyIteDiscount(mdp, iW, iDur, rate, rateBase)
```

```
Run policy iteration using quantity 'Net reward' under discounting criterion
with 'Duration' as duration using interest rate 0.05 and a rate basis equal 1.
Iteration(s): 1 2 3 finished.
```

```
> policy<-getPolicy(mdp, labels=TRUE)
> sIdx<-stateIdxDf(prefix)
> policy<-merge(sIdx,policy)
> policyW<-getPolicyW(mdp, iW)
> policy<-merge(policy,policyW)
> policy$w1<-round(policy$w1,0)
> policy
```

```
  sId n0 s0          label  aLabel     w1
1   0  0  0   Small litter Replace 246159
2   1  0  1 Average litter    Keep 249061
3   2  0  2     Big litter    Keep 252735
```

   The optimal policy is given in column `aLabel` and the optimal expected discounted reward in column `w1`.

9. Find the optimal policy under the expected discounted reward criterion using an interest rate of 2.5%. Compare the optimal values with the values under interest rate 10% and 5%. What is the trend in the numbers and why?

```
> mdp<-loadMDP(prefix)
```

```
Cpu for reading the binary files: 0s
Cpu time for checking MDP 0s.
Cpu time for building state-expanded hypergraph 0s
```

```
> iW<-1                    # index of the weight we want to optimize
> iDur<-0                  # index of the duration/time
> rate<-0.025              # discount rate
> rateBase<-1              # rate base
> policyIteDiscount(mdp, iW, iDur, rate, rateBase)
```

```
Run policy iteration using quantity 'Net reward' under discounting criterion
with 'Duration' as duration using interest rate 0.025 and a rate basis equal 1.
Iteration(s): 1 2 3 finished.
```

```
> policy<-getPolicy(mdp, labels=TRUE)
> sIdx<-stateIdxDf(prefix)
> policy<-merge(sIdx,policy)
> policyW<-getPolicyW(mdp, iW)
> policy<-merge(policy,policyW)
> policy$w1<-round(policy$w1,0)
> policy
```

```
  sId n0 s0          label  aLabel     w1
1   0  0  0   Small litter Replace 489869
2   1  0  1 Average litter    Keep 492758
3   2  0  2     Big litter    Keep 496499
```

It can be seen that for smaller interest rates the optimal values increase. This is due to that a smaller rate corresponds to a larger discount rate and as a result a higher net present value of the expected reward.

Solutions for Exercise 2

1. The model is supplied in the file `sheep.hmp`. Convert the MDP to binary format (hint see `?convertHMP2Binary`) and load the MDP into memory.

```
> prefix<-"sheep_"
> convertHMP2Binary("sheep.hmp",prefix)
```

```
  Statistics:
    states : 77
    actions: 152
    weights: 7

  Closing binary MDP writer.

Converted sheep.hmp to binary format.

   user  system elapsed
   0.69    0.00    0.69
```

```
> mdp<-loadMDP(prefix)
```

```
Cpu for reading the binary files: 0s
Cpu time for checking MDP 0s.
Cpu time for building state-expanded hypergraph 0.062s
```

2. Get an overview over the model (the number of stages, states, actions, criteria etc.). Have a look at data frames for the states and actions.

```
> stateIdxDf(prefix)
> actionInfo(prefix)
```

3. Calculate an optimal replacement policy under the following three criteria of optimality: (a) Maximum present value (discounting) using a discount rate of 10%.

```
> iW<-1                    # index of the weight we want to optimize
> iDur<-0                  # index of the duration/time
> rate<-0.1                # discount rate
> rateBase<-1              # rate base
> policyIteDiscount(mdp, iW=1, iDur=0, rate, rateBase)
```

```
Run policy iteration using quantity 'Net returns' under discounting criterion
with 'Duration' as duration using interest rate 0.1 and a rate basis equal 1.
Iteration(s): 1 2 3 4 finished.
```

```
> policy<-getPolicy(mdp, labels=TRUE)
> sIdx<-stateIdxDf(prefix)
> policy<-merge(sIdx,policy)
> policyW<-getPolicyW(mdp, iW)
> policy<-merge(policy,policyW)
> policy$w1<-round(policy$w1,0)
> policyDis<-policy
```

(b) Maximum average net reward per sheep per year.

```
> g<-policyIteAve(mdp, iW, iDur)
```

```
Run policy iteration under average reward criterion using
reward 'Net returns' over 'Duration'. Iterations (g):
1 (319.613) 2 (349.452) 3 (349.544) 4 (349.544) finished.
```

```
> g
```

```
[1] 349.5437
```

```
> policy<-getPolicy(mdp, labels=TRUE)
> sIdx<-stateIdxDf(prefix)
> policy<-merge(sIdx,policy)
> policyW<-getPolicyW(mdp, iW)
> policy<-merge(policy,policyW)
> policy$w1<-round(policy$w1,0)
> policyAve1<-policy
```

(c) Maximum average net returns per lamb produced.

```
> g<-policyIteAve(mdp, iW, iDur=2)
```

```
Run policy iteration under average reward criterion using
reward 'Net returns' over 'Lambs produced'. Iterations (g):
1 (236.877) 2 (255.998) 3 (256.019) 4 (256.019) finished.
```

```
> g
```

```
[1] 256.0191
```

```
> policy<-getPolicy(mdp, labels=TRUE)
> sIdx<-stateIdxDf(prefix)
> policy<-merge(sIdx,policy)
> policyW<-getPolicyW(mdp, iW)
> policy<-merge(policy,policyW)
> policy$w1<-round(policy$w1,0)
> policyAve2<-policy
```

4. Compare the three optimal policies and explain the differences (if any).

   Have a look at `policyDis`, `policyAve1` and `policyAve2`.

5. For the optimal policy maximizing average net returns per sheep per year, calculate the following technical and economical key figures. (a) Average lifetime of a sheep.

   First optimize net returns per sheep per year

```
> g<-policyIteAve(mdp, iW, iDur=0)
```

```
Run policy iteration under average reward criterion using
reward 'Net returns' over 'Duration'. Iterations (g):
1 (319.613) 2 (349.452) 3 (349.544) 4 (349.544) finished.
```

   Next, we calculate the average lifetime of an sheep under the policy which must be equal the reciprocal to average number of sheep inserted per year.

```
> g<-calcWeights(mdp, iW=3, criterion="average", iDur = 0)
> avgAge<-1/g
> avgAge
```

```
[1] 8.489255
```

   (b) Average number of lambs per sheep per year.

```
> lambPerYear<-calcWeights(mdp, iW=2, criterion="average", iDur = 0)
> lambPerYear
```

```
[1] 1.367056
```

   (c) Average number of lambs produced over the lifetime of an sheep.

```
> lambPerYear*avgAge
```

```
[1] 11.60529
```

Solutions for Exercise 3

1. Generate the model.

```
> prefix="hawk_"
> w<-binaryMDPWriter(prefix)
> w$setWeights("Fitness")
> w$process()
>   for (d in 0:4) {
+     w$stage()
+       for (s in states$sId) {
+         sF<-states$sF[s+1]
+         sB<-states$sB[s+1]
+         w$state(label=states$label[s+1])
+           if (d<4) {
+             if (sF!=1 & sB!=1) {
+               w$action(label="Stay",weights=0,prob=getTransPr(d,s,0), end=T)
+               w$action(label="Hunt",weights=0,prob=getTransPr(d,s,1), end=T)
+             }
+             w$action(label="Desert",weights=0,prob=getTransPr(d,s,2), end=T)
+           }
+         w$endState()
+       }
+     w$endStage()
+   }
> w$endProcess()
> w$closeWriter()
```

```
  Statistics:
    states : 245
    actions: 484
    weights: 1

  Closing binary MDP writer.
```

2. Solve the MDP using value iteration. Hint: remember to set the terminal values of the MDP to `states$endFitness`.

```
> mdp<-loadMDP(prefix)
```

```
Cpu for reading the binary files: 0s
Cpu time for checking MDP 0.015s.
Cpu time for building state-expanded hypergraph 0.078s
```

```
> iW<-0                    # index of the weight we want to optimize
> valueIte(mdp, iW, termValues=states$endFitness)
```

```
Run value iteration using quantity 'Fitness' under expected reward criterion. Finished (0s).
```

```
> sIdx<-stateIdxDf(prefix)
> policy<-getPolicy(mdp, labels=TRUE)
> policy<-merge(sIdx,policy)
> policyW<-getPolicyW(mdp, iW)
> policy<-merge(policy,policyW)
> policy$sF<-as.numeric(substr(policy$label,1,1))
> policy$sB<-as.numeric(substr(policy$label,3,3))
> head(policy)
```

```
  sId n0 s0 label aLabel        w0 sF sB
1   0  0  0   1,1 Desert 0.1951332  1  1
2   1  0  1   2,1 Desert 0.1951332  2  1
3   2  0  2   3,1 Desert 0.1908761  3  1
4   3  0  3   4,1 Desert 0.1933499  4  1
5   4  0  4   5,1 Desert 0.1951332  5  1
6   5  0  5   6,1 Desert 0.1959113  6  1
```

3. A plot of the optimal policy can be created. What is the optimal action in the early fledgling-dependency period when the the physical condition of the female and the brood is 2 and 4, respectively?

   If we consider the subplot for the early fledgling-dependency period (period 2) we see that the optimal decision for the female is to hunt.

References

[1] E.J. Kelly and P.L. Kennedy. A dynamic state variable model of mate desertion in cooper's hawks. *Ecology*, 74(2):351–366, 1993.

[2] Lars Relund Nielsen. *Solving MDPs using the MDP package in R.* Department of Genetics and Biotechnology, University of Aarhus, P.O. Box 50, DK-8830 Tjele, Denmark, July 2009.