Group lab 2

Kevin Neville

19 september 2016

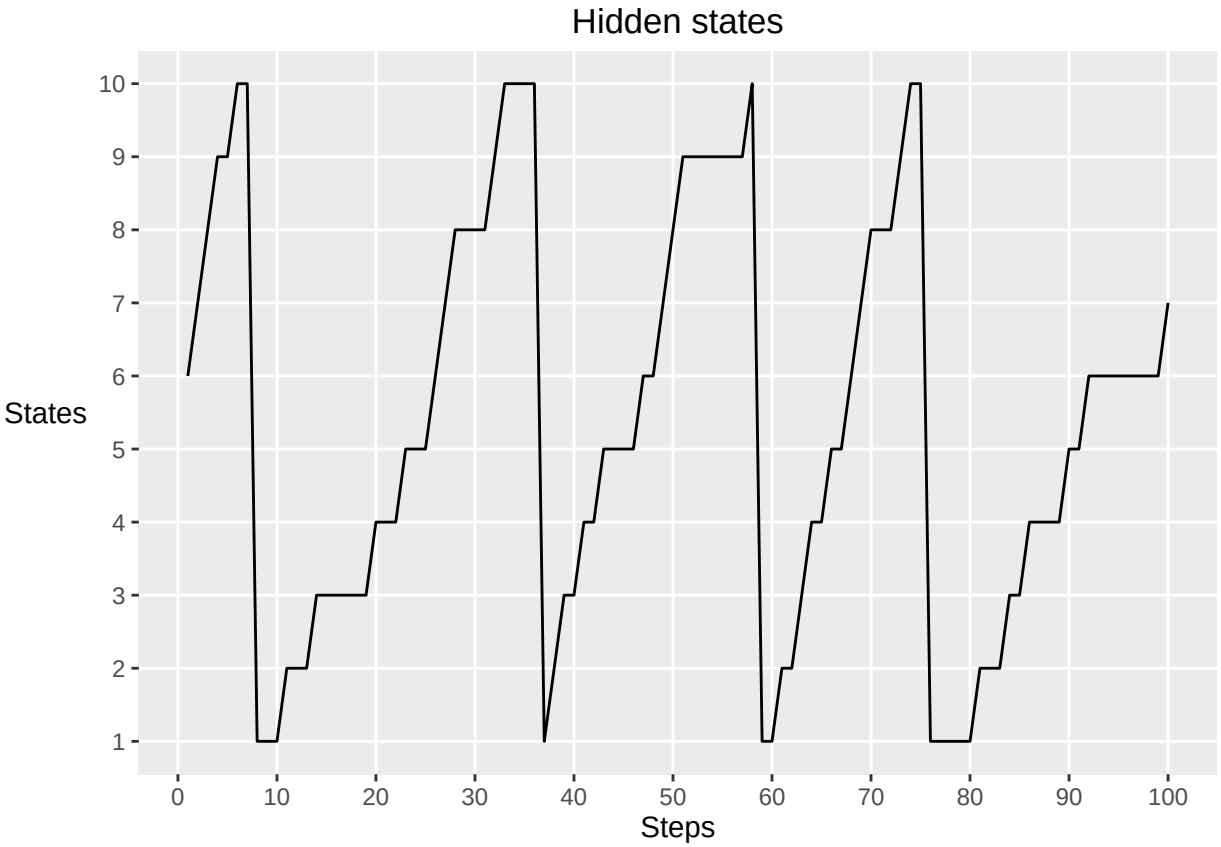
1

According to the following scenario we estimate the model by this code:

```
hmm <- initHMM(States= 1:10, Symbols = 1:10,
               startProbs = rep(0.1,10), transProbs = trans,
               emissionProbs = emission)
```

The robot has a chance of staying of 50 % and 50 % of moving one state up. The GPS of the robot is of poor quailty therefore the transmission matrix will have 20 % probability for all states +- 2 states from the current state. We set the starting probability equal for all the states.

2



We simulate a HMM for 100 of steps.For this specific simulation we started at state six, and ended at state seven after 100 steps. As expected,the robot does not take any steps backwards, either it stays or it moves to a higher state.

3-4

To solve this assignment I choose to make a function to be able to calculate the filtering algorithm. This is done by using the following formula:

$$Filtering: \frac{\alpha(z^t)}{\sum_{\alpha^t} \alpha(z^t)}$$

For the smoothing probabilities I use the function *posterior()*. Which calculates accordingly to this following formula:

$$Smoothing: \frac{\alpha(z^t)\beta(Z^t)}{\sum_{\alpha^t} \alpha(z^t)\beta(Z^t)}$$

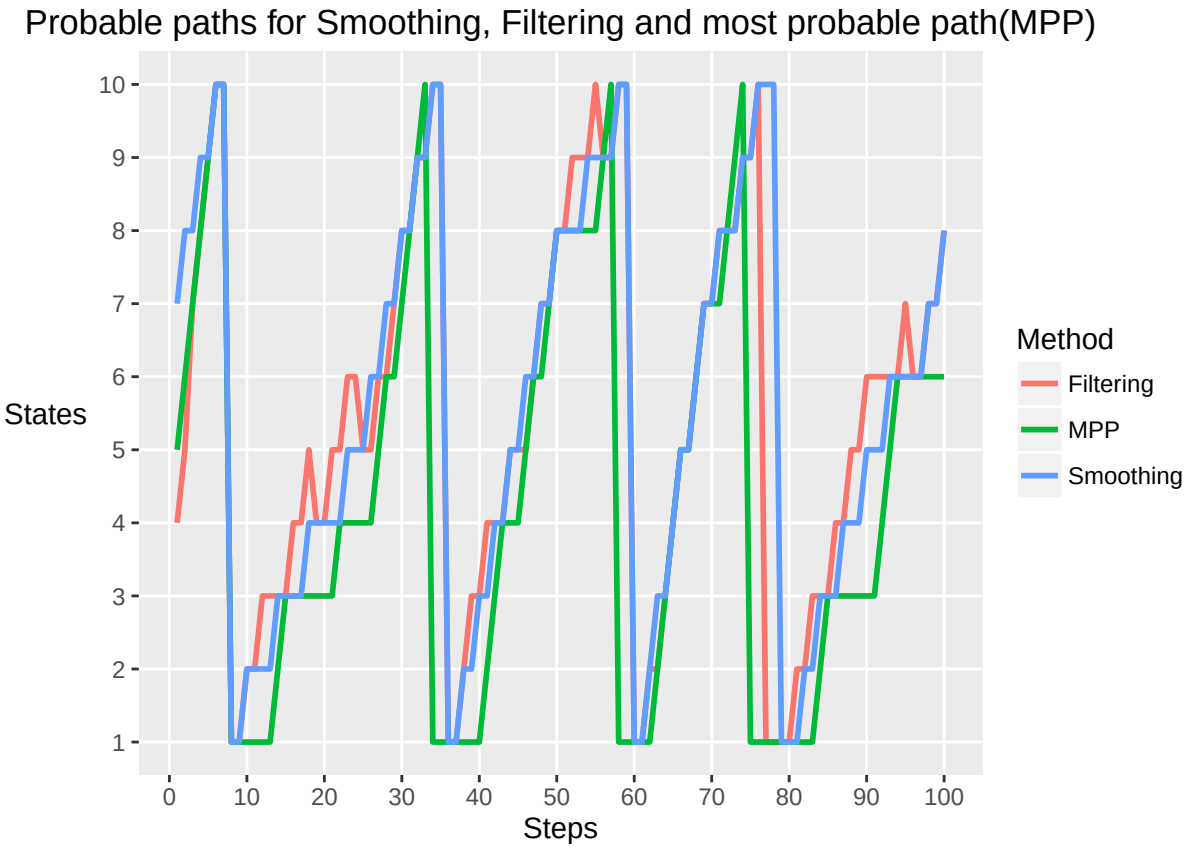
And finally I compute the most probable path using the *Viterbi* algorithm.

The accuarcy is shown below.

```
##     AccuarySmooth AccuarcyFiltering   AccuarcyViterbi
##              0.66              0.57              0.44
```

Here we notice that the accuarcy is highest for the smoothed distribution and the lowest for the most probable path.

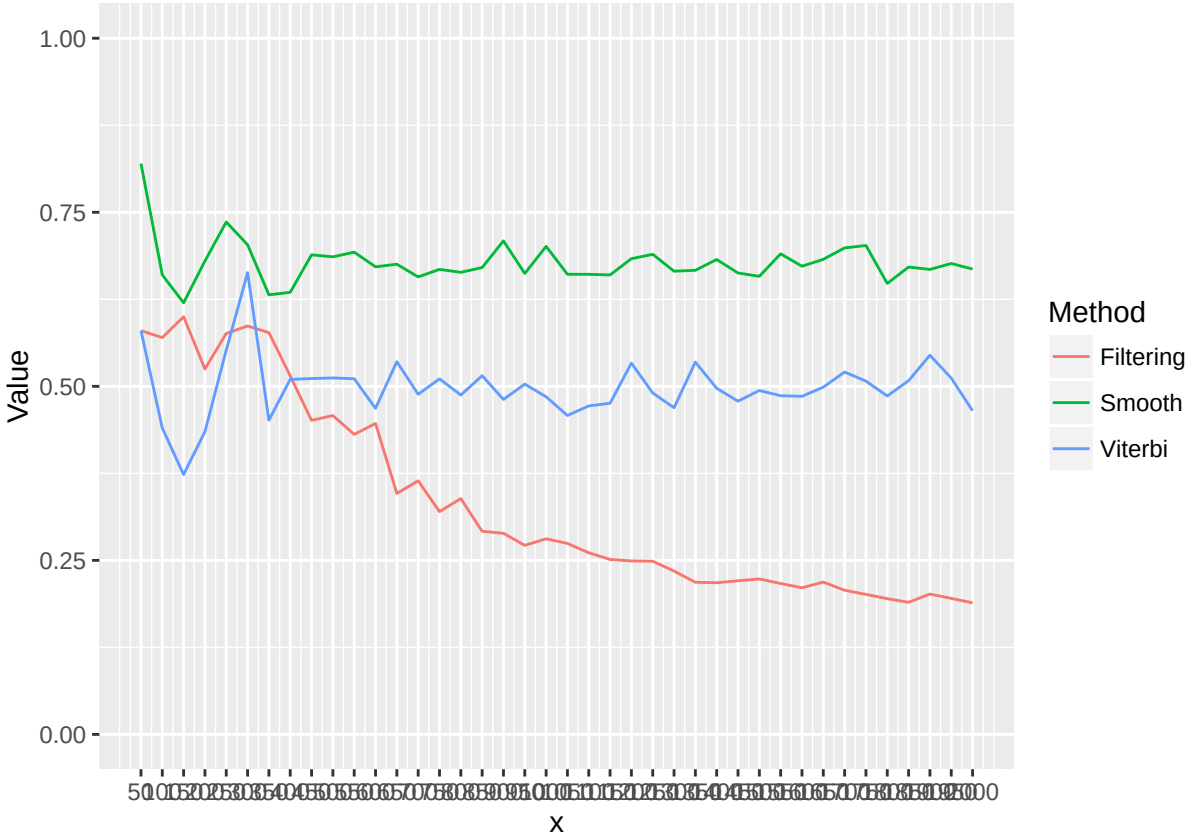
We can plot the predicted paths and investigate them further.



We notice that the different paths start and end in different states. One intresting finding is that the Filtered path at least moves backwards four times, this is not allowed and therefore we suspect that this distribution

will not give the most optimal result. We also notice that the Viterbi algorithm returns a path that get stuck for long sequences, especially for state one.

5

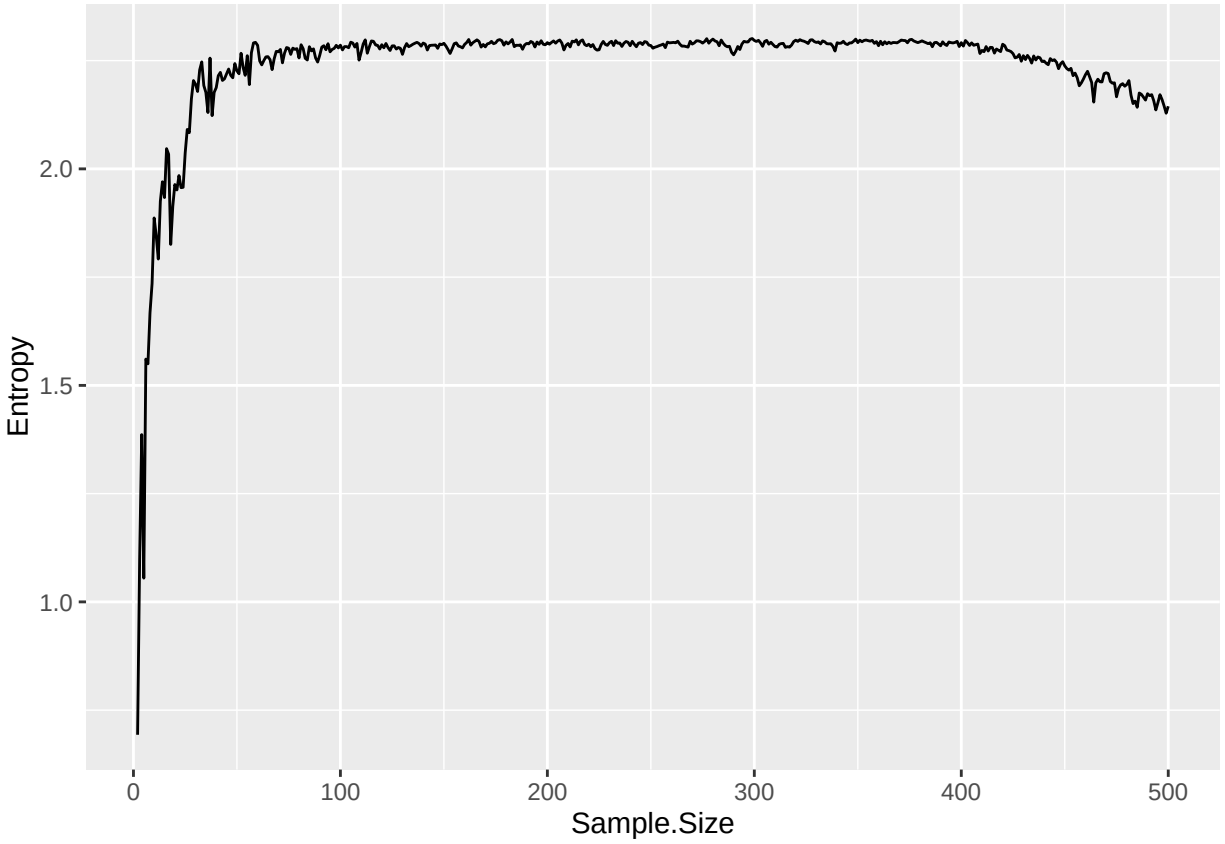


The three diffeerent accuaries is visualized in the above plot. This is done for different sample sizes. We noticed that the Smoothed accuarcys is superior to the two others for all sample sizes. On another note the filtered accuarcy is the worst and it becomes less accuarate as the sample size increases. It is also intresting to notice that both the smoother and MPP seem to settled at a certain level for sample sizes of 400 and more.

The explanation of the smoothed superior behavior compared to the filterd accuarcy is that that the smoothed uses more data. This can be seen in that formulas in assignment 3, where the smoothing algorithm uses both the backward and forward part of the algorithm, compared to the filtered algorithm which only uses the forward part.

The smoothing distribution is obtained by the *Forward-Backwards algorithm* while the most probable path is obtained by the *Viterbi algorithm*. The Forward-backwards algorithm gives marginal probability for each individual state while the Viterbi gives probability for the most likely sequence of states.

6



For sample sizes between 2 and 50 we are becoming more unsure about where the robot position. But it seems like the entropy stabilzes when the sample size equal to 50 or more. We also calculated the entropy when the sample size equal to 100 000, we then obtained a entropy of 0.03132453. This implies that we become more sure for very large samples. We can also see that this trend is present in the end part of the graph.

7

```
##         1         2         3         4         5         6         7         8
## 0.000000 0.000000 0.109531 0.890469 0.000000 0.000000 0.000000 0.000000
##         9        10
## 0.000000 0.000000
```

The predicted state in t=101 for this simulation is that the robot will move to state 4.

$$p(Z^{101}|X^{0:100}) = \sum_{Z^{100}} p(Z^{101}, Z^{100}|X^{0:100})$$

$$p(Z^{101}|Z^{100}, X^{0:100})p(Z^{100}|X^{0:100}) = p(Z^{101}|Z^{100})p(Z^{100}|X^{0:100})$$