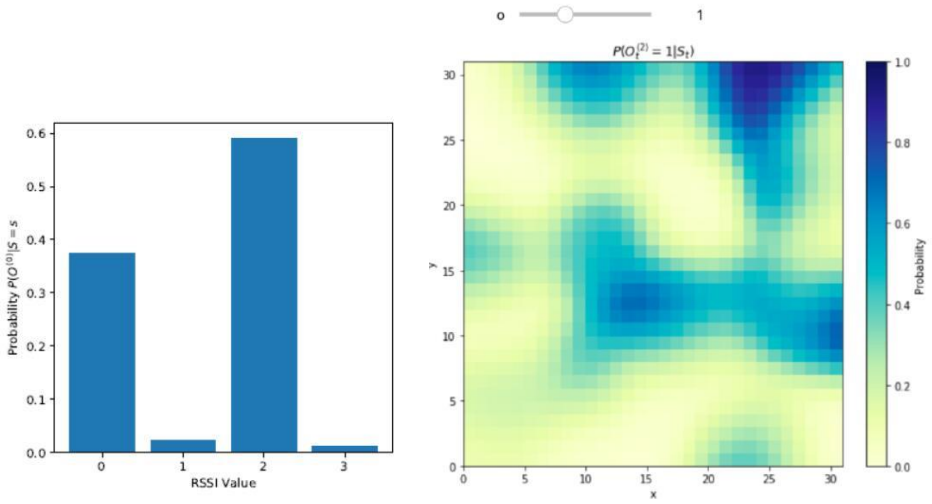
Experiment 2: Hidden Markov Models

Please read the rules for assignments on the course web page. Of course, **do not share code**. Acknowledge any help you received and any sources you used. Please use the online website （Zhihuishu） to turn in the assignment.

1 Introduction

In this assignment, we will consider an indoor positioning system using hidden Markov models (HMMs). The aim here is to locate a mobile sensor receiving signals from beacons placed in a building or a room. In an open area, the signal strength of a message decreases as the distance between the transmitter and the receiver increases; therefore, the strength of the signal received can be directly used to estimate the position of the sensor. However, this is not the case in buildings. The same message can be received with different signal strengths due to reflection and scattering. One way to overcome this problem is to collect the signal strengths for each position and construct a probability distribution. This procedure is called fingerprinting. An example distribution for a particular position is shown in Figure 1a. We can estimate the position of the sensor using these probability distributions as our observation model. We also need to specify the transition probabilities to obtain a complete HMM. If there is no prior knowledge about the motion of the sensor, we can model the transitions as a random walk.



(a) The probability of observing a RSSI value for a particular position

(b) The probability of observing the RSSI value 1

Figure 1: An example observation model

2 Helper Code

In this assignment, you are given the transition and observation models, and you need to code up the algorithms that calculate the monitoring probabilities. The helper code uses the following Python libraries: numpy, matplotlib and ipywidgets. You can install them using pip:

```
$ pip install numpy matplotlib ipywidgets
```

We strongly recommend that you use a Jupyter notebook (or JupyterLab) to plot the probabilities you compute. A couple of examples are shown in Figure 1b and 2. You also need to enable the widget extension to have interactive controls:

```
$ pip install jupyter
$ jupyter nbextension enable --py widgetsnbextension
```

Once you install it, you can run it by:

```
$ jupyter notebook /path/to/hw/directory
```

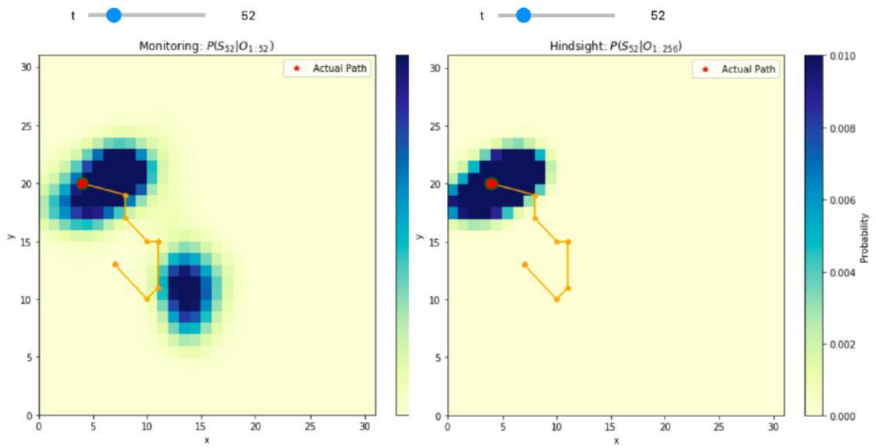


Figure 2: Example Monitoring and Hindsight Plots

The indoor environment is represented by a grid where each cell corresponds to a state in the HMM, which makes the total number of states n_state = (width * length). Thus, a probability distribution of the location of the target sensor is a vector of length n_states. The transition probabilities $P(S_{t+1}|S_t)$ are stored in a 2-dimensional NumPy array, which can be accessed through self.trans probs. For example, the transition probability from State 1 to State 3 can be obtained by self.trans prob [3,1]. In each time step, the sensor receives an observation vector o of length n_beacons. The Received Signal Strength Indicator (RSSI) value of the message received from the beacon b is stored in o[b]. The probability of observing o[b] in a particular state s, $P(O^{(b)} = o_b|S = s)$, can be obtained by self.obs probs[b,o[b],s]. Here, we assume the

RSSI values of different messages are conditionally independent. Therefore, you can decompose the probability of observing the vector o, $P(O = o \mid S = s)$ into a product of individual likelihood functions:

$$P(O = o|S = s) = \prod_{i=0}^{\text{n\_beacons}-1} P(O^{(i)} = o_i|S = s).$$

3 Submission Instructions and Testing

You need to implement three methods of the HMM class, namely predict, update and monitoring in hmm.py file. The backward reasoning has already been implemented. You can access and use the attributes listed right above the constructor. You can also use NumPy (you should!) and the standard library, but you cannot use/import anyone else's code or any other package. You should only submit hmm.py to the Zhihushu assignment "Experiment 2: Hidden Markov Models", and you should upload it directly without zipping.

We have given you a Jupyter notebook hmm.ipynb where you can find several examples about how to use the methods and how to plot the computed probabilities. You can also test your code using hmm test.py:

```
$ python -m unittest hmm_test.py
```