Problem 2

In this Jupyter notebook, we visualize how HMMs work. This visualization corresponds to problem 2 in set 6.

Assuming your HMM module is complete and saved at the correct location, you can simply run all cells in the notebook without modification.

In [1]:
```
import os
import numpy as np
from IPython.display import HTML

from HMM import unsupervised_HMM
from HMM_helper import (
    text_to_wordcloud,
    states_to_wordclouds,
    parse_observations,
    sample_sentence,
    visualize_sparsities,
    animate_emission
)
```

Visualization of the dataset

We will be using the Constitution as our dataset. First, we visualize the entirety of the Constitution as a wordcloud:

In [2]:
```
text = open(os.path.join(os.getcwd(), 'data/constitution.txt')).read()
wordcloud = text_to_wordcloud(text, title='Constitution')
```

Training an HMM

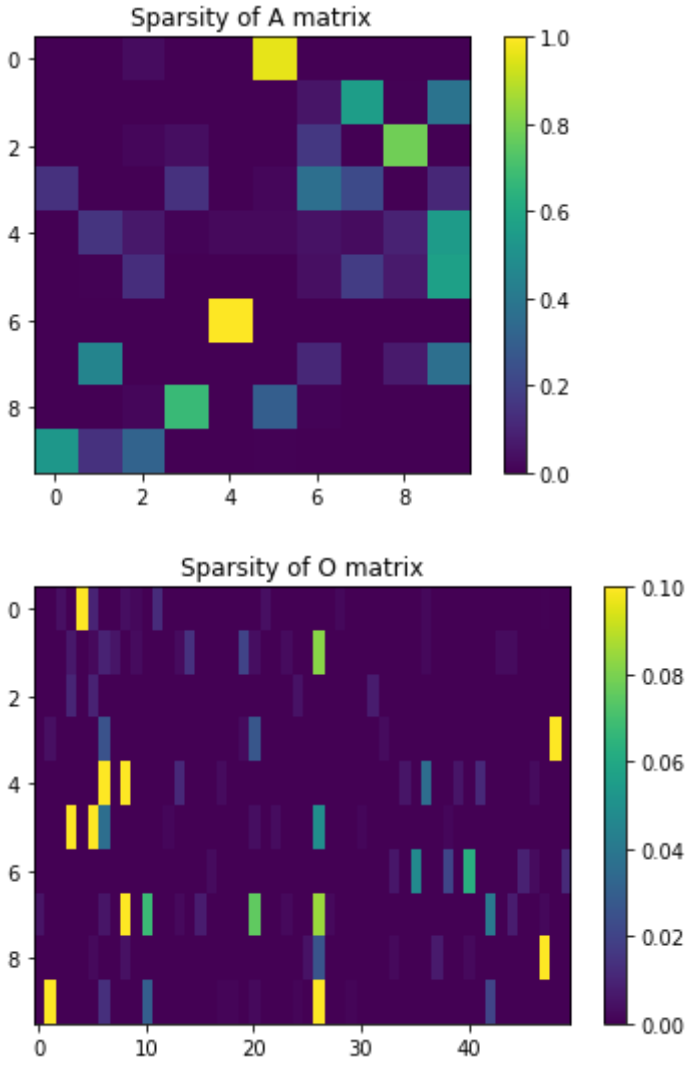
Now we train an HMM on our dataset. We use 10 hidden states and train over 100 iterations:

In [3]:
```
obs, obs_map = parse_observations(text)
hmm8 = unsupervised_HMM(obs, 10, 100)
```

Part G: Visualization of the sparsities of A and O

We can visualize the sparsities of the A and O matrices by treating the matrix entries as intensity values and showing them as images. What patterns do you notice?

In [4]:
```
visualize_sparsities(hmm8, O_max_cols=50)
```



Generating a sample sentence

As you have already seen, an HMM can be used to generate sample sequences based on the given dataset. Run the cell below to show a sample sentence based on the Constitution.

In [5]:
```
print('Sample Sentence:\n====================')
print(sample_sentence(hmm8, obs_map, n_words=25))
```

```
Sample Sentence:
====================
Them to quorum captures or our shall not any which accept states the ma
ke protect the congress section by representative on the own states con
gress...
```

Part H: Using varying numbers of hidden states

Using different numbers of hidden states can lead to different behaviours in the HMMs. Below, we train several HMMs with 1, 2, 4, and 16 hidden states, respectively. What do you notice about their emissions? How do these emissions compare to the emission above?

In [6]:
```
hmm1 = unsupervised_HMM(obs, 1, 100)
print('\nSample Sentence:\n====================')
print(sample_sentence(hmm1, obs_map, n_words=25))
```

```
Sample Sentence:
====================
Organizing all be coin or between one of and forfeiture state seven hou
se be the states likewise in several blood in erected the of in...
```

In [7]:
```
hmm2 = unsupervised_HMM(obs, 2, 100)
print('\nSample Sentence:\n====================')
print(sample_sentence(hmm2, obs_map, n_words=25))
```

```
Sample Sentence:
====================
And to be general the officers be and courts shall constitute any excep
t suffrage six notwithstanding of to the manner prejudice one nominate
the and...
```

In [8]:
```
hmm4 = unsupervised_HMM(obs, 4, 100)
print('\nSample Sentence:\n====================')
print(sample_sentence(hmm4, obs_map, n_words=25))
```

```
Sample Sentence:
====================
Legislature more on his grant regulate crimes excises coin office and b
y law on which manner as the useful during post diminished by meet t
o...
```

In [9]:
```
hmm16 = unsupervised_HMM(obs, 16, 100)
print('\nSample Sentence:\n====================')
print(sample_sentence(hmm16, obs_map, n_words=25))
```

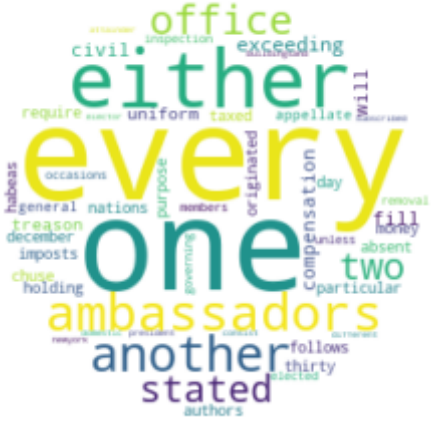
```
Sample Sentence:
====================
Disapproved establish the direct days or the make discoveries a require
either of the section or to the excepting or of treason the first hous
e...
```

Part I: Visualizing the wordcloud of each state

Below, we visualize each state as a wordcloud by sampling a large emission from the state:

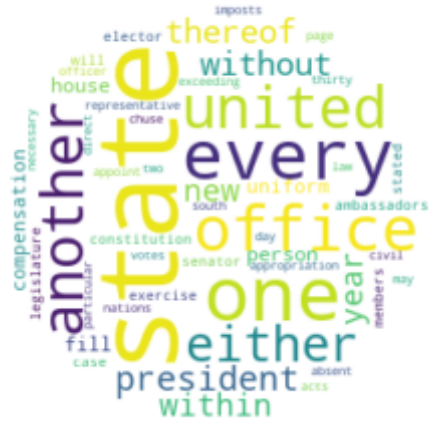
In [10]:
```
wordclouds = states_to_wordclouds(hmm8, obs_map)
```

State 0



State 1

State 2



State 3

State 4

State 5

State 6

State 7

State 8

Visualizing the process of an HMM generating an emission

The visualization below shows how an HMM generates an emission. Each state is shown as a wordcloud on the plot, and transition probabilities between the states are shown as arrows. The darker an arrow, the higher the transition probability.

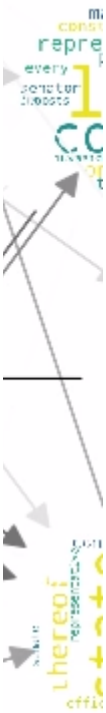
At every frame, a transition is taken and an observation is emitted from the new state. A red arrow indicates that the transition was just taken. If a transition stays at the same state, it is represented as an arrowhead on top of that state.

Use fullscreen for a better view of the process.

In [11]:
```
anim = animate_emission(hmm8, obs_map, M=8)
HTML(anim.to_html5_video())
```

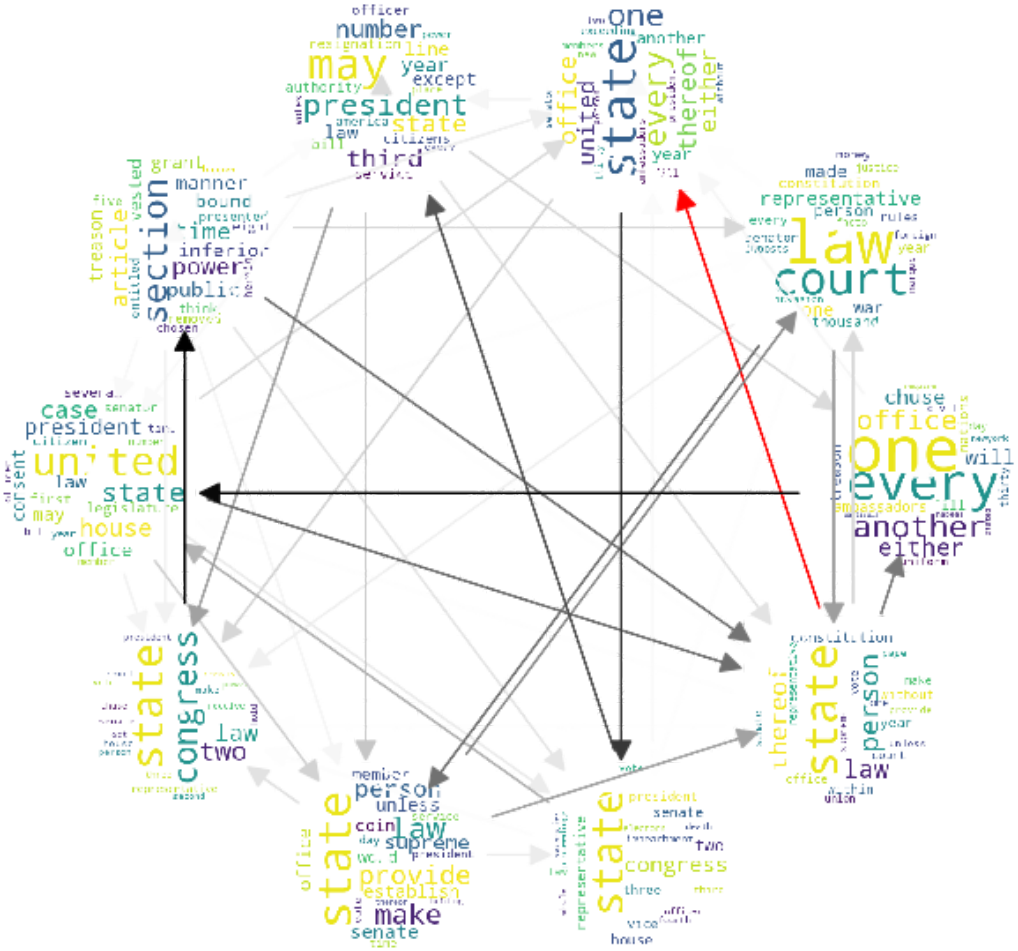
```
Animating...
```

Out[11]:



0:00 / 0:09

Be not bound as three and of the



In [ ]:

In [ ]: