AMSI VACATION RESEARCH SCHOLARSHIPS 2018-2019

Modelling Wind Farm Power Output Using Hidden Markov Models and its Implications for South Australian Wind Farms

William Abbott
Supervised by Giang Nguyen
University of Adelaide

February 28, 2019

Vacation Research Scholarships are funded jointly by the Department of Education and Training and the Australian Mathematical Sciences Institute.

AMSI AUSTRALIAN MATHEMATICAL SCIENCES INSTITUTE

Australian Government
Department of Education and Training

Contents

Abstract

Electricity is one of the most relevant commodities in this era, and the increasing penetration of renewable energy raises questions about South Australia's grid stability. With wind energy comprising a large portion of the total energy produced in South Australia, accurately modelling its behaviour is becoming increasingly important. Hidden Markov Models are a type of mathematical model which describe randomly-varying systems in which there is an unknown entity. The environmental conditions affecting wind farm generation are considered the unknown entity, with the generation amount being the observation. We show that Hidden Markov Models are a viable option to model wind farms; however, higher computational power is needed in order to increase accuracy.

1 Introduction

Electricity is one of the worlds most important commodities, and so the stability of the power network is also as important. This is particularly relevant to South Australia, which has been hit recently with power outages. With renewable energy contributing 47.2% to the total electricity generation in 2018 [1], one of the highest rates of renewable energy penetration in the world, with wind-generated electricity consisting 39%, its evolution is especially critical. Due to the intermittent nature of renewable energy, accurate modelling needs to be implemented in order to forecast when extra generation could be needed.

The generators on the South Australian grid are registered with the Australian Energy Market Operator (AEMO). Each generator sends bids to AEMO on the amount of electricity they can generate and the attatched price. AEMO collects these bids and, using an estimate of the electricity demand obtained by measuring the total current usage at specific points in the South Australia grid, chooses which generators to use to minimise the total cost. Then, this total cost determines the spot price. Even over a small timeframe of two days, the demand and price of electricity has many peaks and troughs seen in Figure 1, demonstrating how volatile the market is. In extreme weather events such as heatwaves, demand is substantially higher than normal; however, this is also when non-renewable energy generators find their capacity minimised as the high temperature negatively affects the control systems in the generating plants. This means that the whole of the grid has to rely on intermittent renewable energy. However, with an accurate model, AEMO would plan ahead based on weather and other events in order to ensure that supply and demand match. If it becomes imbalanced, this could result in large price spikes and power outages.

There are currently two main methods of modelling wind farm output, either *direct* or *indirect*. The indirect approach first models the wind speed in a given area then uses this to calculate the power output; however, direct modelling of the power output is going to be the focus of this paper, as described in [2].

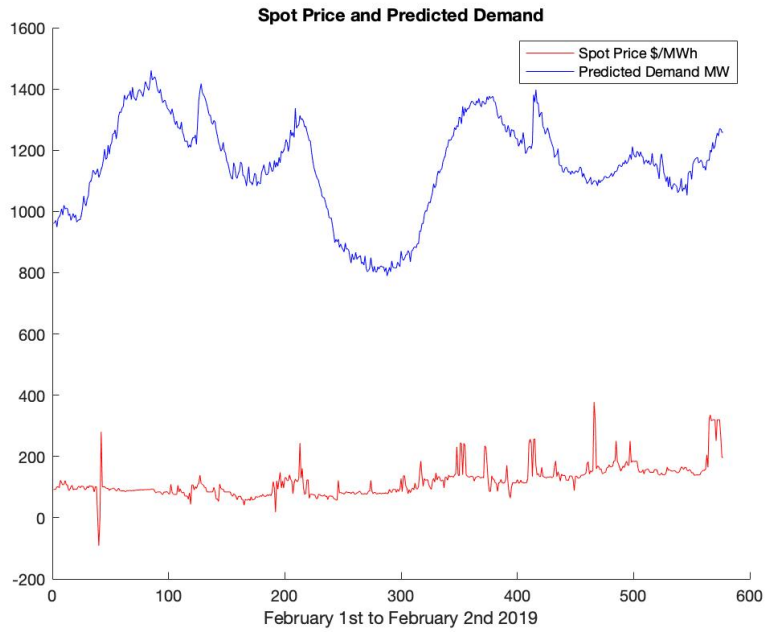


Figure 1: Spot Price and Predicted Demand from 1/2/19 to 2/2/19 in South Australia

2 Preliminaries

2.1 Hidden Markov Model

A Hidden Markov Model $\{X(t), Y(t)\}_{t\in\mathbb{N}}$ is a model that consists of a state sequence $\{X(t)\}$ and an observation sequence $\{Y(t)\}$. We assume the state space of the underlying hidden process is $S_X = \{1, \ldots, M\}$ and the state space of the observations is $S_Y = \{1, \ldots, N\}$. Let the vector $\boldsymbol{X_T} = (X_1, \ldots, X_T)$ be the sequence of hidden states the process takes where X_t is the state of the process at time $t \in \{1, \ldots, T\}$ with $X_t \in S_X$. Similarly, let $\boldsymbol{Y_T} = (Y_1, \ldots, Y_T)$ be the sequence of observations determined by the process, where Y_t is the observation at time $t \in \{1, \ldots, T\}$ with $Y_t \in S_Y$. A Hidden Markov Model is defined by a vector of parameters $\boldsymbol{\theta}$ consisting of P, δ and π, where P is the state transition matrix of the model, δ is the observation-state probability matrix and $\boldsymbol{\pi}$ is the initial state

distibution vector,

$$\boldsymbol{\pi} = [\pi_i]_{i\in\{1,\dots,M\}}, \qquad \pi_i = \mathbb{P}(X_1 = i \mid \boldsymbol{\theta}),$$
$$P = [P_{i,j}]_{i,j\in\{1,\dots,M\}}, \qquad P_{i,j} = \mathbb{P}(X_{t+1} = j \mid X_t = i, \boldsymbol{\theta}),$$
$$\delta = [\delta_i(j)]_{i,j\in\{1,\dots,M\}}, \qquad \delta_i(j) = \mathbb{P}(Y_t = j \mid X_t = i, \boldsymbol{\theta}).$$

A Hidden Markov Model describes a system in which there is a hidden Markovian state sequence determined by a transition matrix and, at each transition, there is an observation that is determined by an observation-state probability matrix. There are two types of Hidden Markov Models: dependent and independent.

A dependent Hidden Markov Model increases the complexity of the normal system by allowing each observation $\{Y(t)\}$ to be dependent also on the previous observation $\{Y(t-1)\}$. Because of the additional dependence, the $\delta_i(j)$ parameter is replaced by $\delta_{i,k}(j) = \mathbb{P}(Y_t = j \mid Y_{t-1} = k, X_t = i)$ and $\Delta_{i,j} = \mathbb{P}(Y_1 = j \mid X_1 = i)$. $\Delta_{i,j}$ is then the same as $\delta_i(j)$ for $t = 1$ and $\delta_{i,k}(j)$ takes into account that, after the first observation, there is an observation dependence.

An independent Hidden Markov Model erases this dependency so that each observation is conditionally independent on the observation preceding it, if you know the current state. In mathematical terms if we consider the expression

$$\mathbb{P}(Y_t = y_t \mid Y_{t-1} = y_{t-1}, X_t = x_t),$$

then

$$\mathbb{P}(Y_t = y_t \mid Y_{t-1} = y_{t-1}, X_t = x_t) = \mathbb{P}(Y_t = y_t \mid X_t = x_t).$$

Although this simplifies the model, it removes the intuitive dependence of each power output observation on the previous one. However, this information will also be kept in the hidden states as part of the distribution of power output in the independent model, although to a lesser extent.

3 Expectation-Maximisation Algorithm

The method we used to estimate the model parameters from the observations is the Expectation-Maximisation (EM) algorithm [4]. Similar to classical parameter estimation for a model, we maximise the likelihood of seeing the particular observation sequence. However, in this model, half of the observations are hidden, and so the idea of the EM algorithm is to replace the missing data with its expectation. This involves summing over the likelihood of each possible state sequence, and multiplying

by the probability of seeing that state sequence which gives us a weighted likeihood. From this, the likelihood is then maximised over the model parameters.

The EM Algorithm converges to local maximums, however, it usually does so quickly [5]. To increase the chance of finding the global maximum, ten random parameter estimates are used as the initial estimates that are fed into the EM algorithm. Only the final estimates with the highest likelihood are then used. In an ideal situation, 1000 or even 10000 random parameter estimates would be used but due to the reduced computing power, using only ten estimates was feasible.

4 Baum-Welch Algorithm

Named after Leonard Baum and Lloyd Welch [3], when the EM Algorithm is applied to a Hidden Markov Model it reduces to the Baum-Welch Algorithm. The model parameters for the $(n+1)^{st}$ iteration of the EM algorithm applied to the independent model are

$$\begin{aligned}
\pi_i &= \mathbb{P}(x_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n), \\
\delta_i(j) &= \frac{\sum_{t=1}^{T} \mathbb{P}(x_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\mathbb{I}(y_t = j)}{\sum_{t=1}^{T} \mathbb{P}(x_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}, \\
P_{i,j} &= \frac{\sum_{t=2}^{T} \mathbb{P}(X_{t-1} = i, X_t = j \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}{\sum_{t=2}^{T} \mathbb{P}(X_{t-1} = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}.
\end{aligned}$$

(For derivation see Appendix.)

For the dependent model $\delta_i(j)$ is not being estimated, instead, we need to estimate $\delta_{i,k}(j) = \mathbb{P}(y_t = j \mid y_{t-1} = k, X_t = i)$ and $\Delta_{i,j} = \mathbb{P}(y_1 = j \mid X_1 = i)$. The parameter estimates for the $(n+1)^{st}$ iteration of the EM algorithm applied to the dependent model are

$$\begin{aligned}
\pi_i &= \mathbb{P}(x_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n), \\
P_{i,j} &= \frac{\sum_{t=2}^{T} \mathbb{P}(X_{t-1} = i, X_t = j \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}{\sum_{t=2}^{T} \mathbb{P}(X_{t-1} = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}, \\
\delta_{i,k}(j) &= \frac{\sum_{t=2}^{T} \mathbb{P}(X_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\mathbb{I}(y_t = j, y_{t-1} = k)}{\sum_{t=2}^{T} \mathbb{P}(X_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\mathbb{I}(y_{t-1} = k)}, \\
\Delta_{i,j} &= \frac{\mathbb{P}(X_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\mathbb{I}(y_1 = j)}{\mathbb{P}(X_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}.
\end{aligned}$$

(For derivation see Appendix.)

4.1 Numerical Solutions for Updated Parameters

There are four main probabilities used to calculate the updated parameters numerically:

$$\begin{aligned}
\alpha_t(i \mid \boldsymbol{\theta}) &= \mathbb{P}(Y_1 = y_1, \ldots, Y_t = y_t, X_t = i \mid \boldsymbol{\theta}),\\
\beta_t(i \mid \boldsymbol{\theta}) &= \mathbb{P}(Y_{t+1} = y_{t+1} \ldots, Y_T = y_T \mid X_t = i, \boldsymbol{\theta}),\\
\gamma_t(i \mid \boldsymbol{\theta}) &= \mathbb{P}(X_t = i \mid Y_1 = y_1, \ldots, Y_T = y_T, \boldsymbol{\theta}),\\
\zeta_t(i, j) &= \mathbb{P}(X_t = i, X_{t+1} = j \mid \boldsymbol{Y}, \boldsymbol{\theta}).
\end{aligned}$$

Both α and β functions can be calculated recursively, as shown in the Appendix, and easily. When the relevant code is vectorised, the functions can be executed even more efficiently. An important note is that $\mathbb{P}(\boldsymbol{Y} \mid \boldsymbol{\theta}) = \sum_{i=1}^{N} \alpha_T(i \mid \boldsymbol{\theta})$, which can be derived straight from the definition:

$$\begin{aligned}
\sum_{i=1}^{N} \alpha_T(i \mid \boldsymbol{\theta}) &= \sum_{i=1}^{N} \mathbb{P}(Y_1 = y_1, \ldots, Y_T = y_T, X_t = i \mid \boldsymbol{\theta})\\
&= \mathbb{P}(Y_1 = y_1, \ldots, Y_T = y_T \mid \boldsymbol{\theta})\\
&= \mathbb{P}(\boldsymbol{Y} \mid \boldsymbol{\theta}).
\end{aligned}$$

Then,

$$\begin{aligned}
\gamma_t(i \mid \boldsymbol{\theta}) &= \mathbb{P}(X_t = i \mid Y_1 = y_1, \ldots, Y_T = y_T, \boldsymbol{\theta})\\
&= \frac{\mathbb{P}(X_t = i, Y_1 = y_1, \ldots, Y_T = y_T \mid \boldsymbol{\theta})}{\mathbb{P}(Y_1 = y_1, \ldots, Y_T = y_T \mid \boldsymbol{\theta})}\\
&= \frac{\mathbb{P}(Y_{t+1} = y_{t+1}, \ldots, Y_T = y_T \mid X_t = i, Y_1 = y_1, \ldots, Y_t = y_t, \boldsymbol{\theta})\mathbb{P}(X_t = i, Y_1 = y_1, \ldots, Y_t = y_t \mid \boldsymbol{\theta})}{\mathbb{P}(\boldsymbol{Y} \mid \boldsymbol{\theta})}.
\end{aligned}$$

We can then use the conditional independence property of the Hidden Markov Model to reduce the expression to

$$\frac{\mathbb{P}(Y_{t+1} = y_{t+1}, \ldots, Y_T = y_T \mid X_t = i, \boldsymbol{\theta})\mathbb{P}(X_t = i, Y_1 = y_1, \ldots, Y_t = y_t \mid \boldsymbol{\theta})}{\mathbb{P}(\boldsymbol{Y} \mid \boldsymbol{\theta})} = \frac{\beta_t(i)\alpha_t(i)}{\mathbb{P}(\boldsymbol{Y} \mid \boldsymbol{\theta})},$$

which is straightforward to calculate numerically once the α and β functions are found. Finally, to calculate ζ, note that

$$\begin{aligned}
\zeta_t(i,j) &= \mathbb{P}(X_t = i, X_{t+1} = j \mid \boldsymbol{Y}, \boldsymbol{\theta}) \\
&= \frac{\mathbb{P}(X_t = i, X_{t+1} = j, \boldsymbol{Y} \mid \boldsymbol{\theta})}{\mathbb{P}(\boldsymbol{Y} \mid \boldsymbol{\theta})} \\
&= \frac{\mathbb{P}(Y_{t+2}, \ldots, Y_T \mid Y_1, \ldots, Y_{t+1}, X_{t+1} = j, X_t = i, \boldsymbol{\theta}) \mathbb{P}(Y_1, \ldots, Y_{t+1}, X_{t+1} = j, X_t = i \mid \boldsymbol{\theta})}{\mathbb{P}(\boldsymbol{Y} \mid \boldsymbol{\theta})} \\
&= \frac{\beta_{t+1}(j) \mathbb{P}(Y_{t+1} \mid Y_1, \ldots, Y_t, X_{t+1} = j, X_t = i, \boldsymbol{\theta}) \mathbb{P}(Y_1, \ldots, Y_t, X_{t+1} = j, X_t = i \mid \boldsymbol{\theta})}{\mathbb{P}(\boldsymbol{Y} \mid \boldsymbol{\theta})} \\
&= \frac{\beta_{t+1}(j) \mathbb{P}(Y_{t+1} \mid X_{t+1} = j, \boldsymbol{\theta}) \mathbb{P}(X_{t+1} = j \mid Y_1, \ldots, Y_t, X_t = i, \boldsymbol{\theta}) \mathbb{P}(Y_1, \ldots, Y_t, X_t = i \mid \boldsymbol{\theta})}{\mathbb{P}(\boldsymbol{Y} \mid \boldsymbol{\theta})} \\
&= \frac{\beta_{t+1}(j) \mathbb{P}(Y_{t+1} \mid X_{t+1} = j, \boldsymbol{\theta}) P_{i,j} \alpha_t(i)}{\mathbb{P}(\boldsymbol{Y} \mid \boldsymbol{\theta})}.
\end{aligned}$$

Now, the updated parameters can be numerically estimated:

$$\begin{aligned}
\pi_i &= \gamma_0(i), \\
\delta_i(j) &= \frac{\sum_{\substack{t=1 \\ y_t = j}}^{T} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)}, \\
P_{i,j} &= \frac{\sum_{t=1}^{T-1} \zeta_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}.
\end{aligned}$$

Intuitively, each element of the δ matrix is the fraction of time that the system is in a state i and the observed output is j. Similarly, each element in $P_{i,j}$ is the fraction of time when, on leaving state i, the system moves to state j. Furthermore, each element in $\boldsymbol{\pi}$ is the fraction of time that the system is in a state i at the beginning of the sequence.

5 Data

The data used to estimate the parameters for the model is from the Australian Energy Market Operator (AEMO), and comprises of the electricity generated at five-minute intervals from all the registered wind farms in South Australia from the 1/1/2018 to 1/1/2019. Due to the sheer volume of data, the generation values were combined to retain the daily amount. Even though the wind conditions can vary greatly over the course of a day, amalgamating the data in this way will give an indication of the overall daily trend. Clearly, the model would benefit heavily from including the five-minute intervals but due to restraints on computational power only the daily value was calculated. The data was split into two sections: Jan-Jun and Jul-Dec, to train and to test on, respectively.

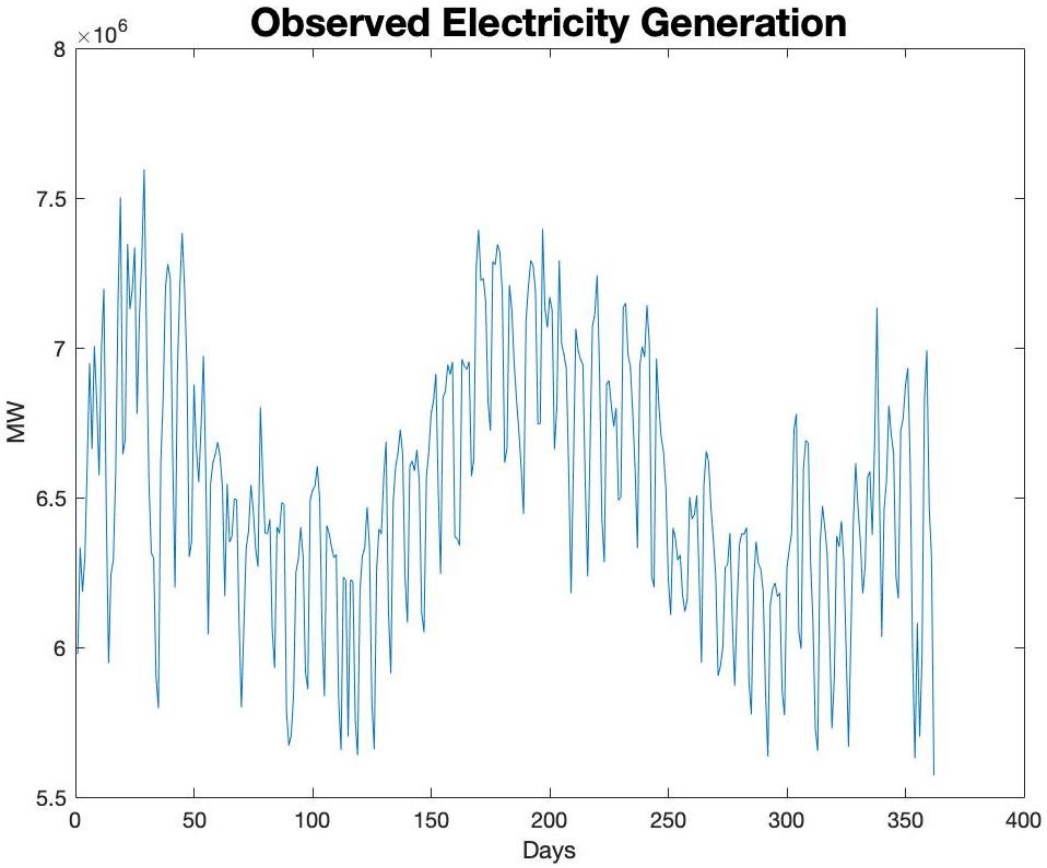
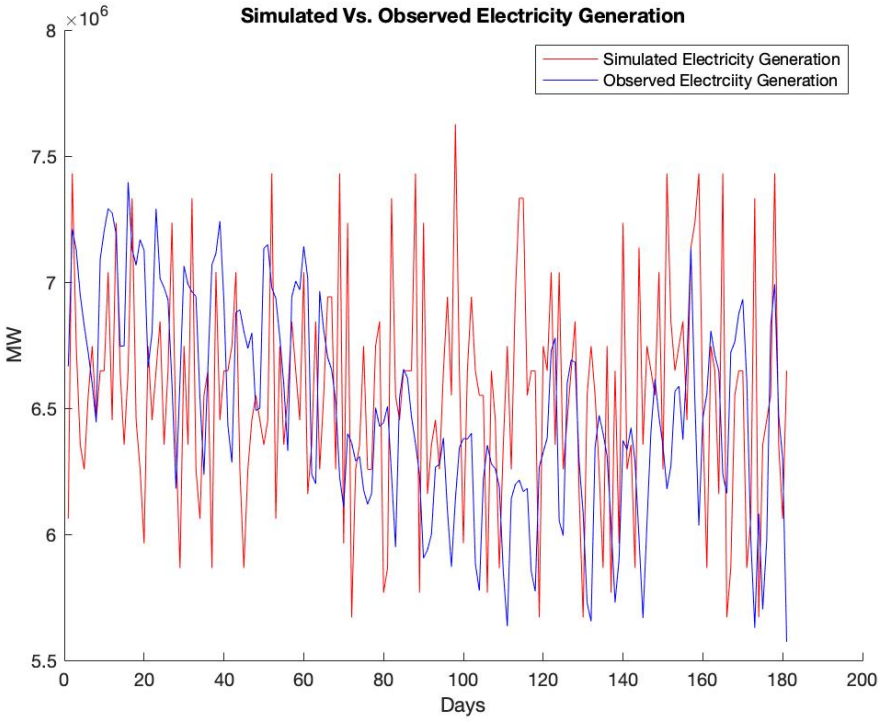


Figure 2: Electricity generation from wind farms in South Australia in 2018 at daily intervals

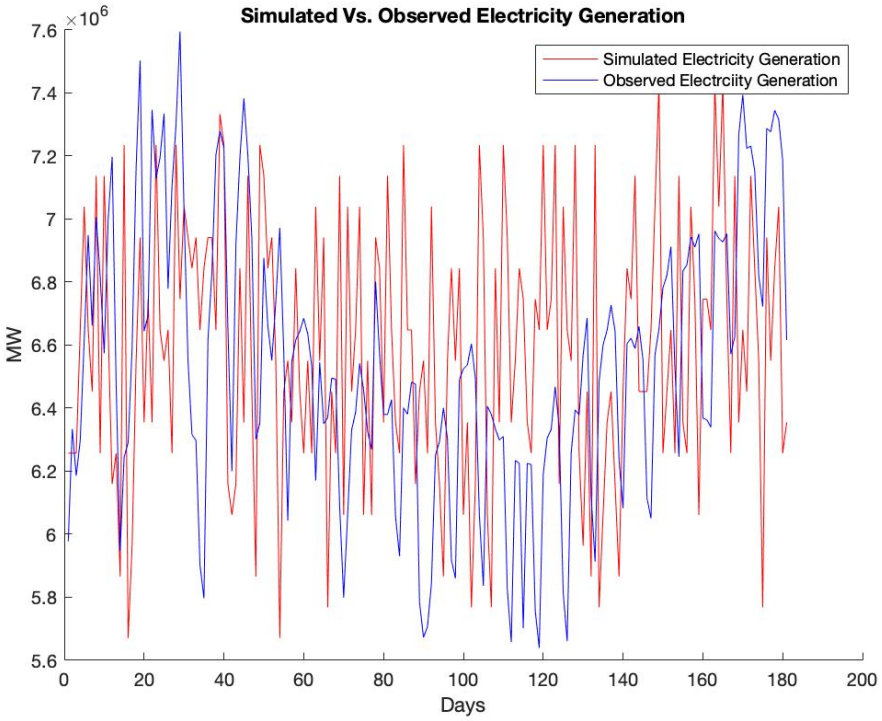
As seen in Figure 2 there is a cyclic behaviour present in the energy generation; however, there is large variation between one day and the next.

6 Results

In order to run the EM Algorithm on the available data, the number of hidden and observed states had to be chosen. In these tests, the number of hidden states chosen were 20 and 60, while the number of observation states tested were 20 and 40. These values were chosen as, if the number of states were larger, more computational power would be needed and if the the number of states was smaller, it would oversimplify the model.



(a) Jan-Jun data with 20 observation states and 20 hidden states



(b) Jul-Dec data with 20 observation states and 20 hidden states

Figure 3: Average of 100 simulations vs. observed wind farm power output in South Australia in 2018

		Number of Hidden States	
		20	60
Number of Observation States	40	7.35%	7.20%
	20	7.37%	6.92%

Table 1: Percentage error of simulation with parameters estimated from Jan-Jun data

		Number of Hidden States	
		20	60
Number of Observation States	40	7.47%	7.39%
	20	7.16%	7.02%

Table 2: Percentage error of simulation with parameters estimated from July-Dec data

7 Discussion

As seen in Figure 3a especially, the individual spikes were not always accurate, however the general trend of the data was, supporting the hypothesis that the model would capture the general trend but would not accurately predict individual values. Overall, the error percent was around 7% however did decrease slightly when the number of hidden states increased. Also of note is, when the number of observation states increased the error percentage also increased, which might be due to the increased spread of probability over the larger amount of states.

In order to provide an accurate model with the sheer amount of data, a large number of hidden and observation states needed to be used. Unfortunately, this led to precision and computer memory issues, and so a more simplified version is used. Another issue is that the EM Algorithm frequently converges to local maximums, which makes it difficult to determine when the global maximum has been reached. To mitigate this error, we fed multiple random initial parameters into the EM algorithm and chose the final parameters with the largest likelihood. There is also another approach in determining the model, in which each individual turbine is modelled separately and then the complete model is the sum of all models. However, this would increase the number of hidden and observation state choices drastically.

8 Conclusion

The simulated data reproduced the observed data fairly well even though the individual peaks were not always taken into account Overall HMMs are a simple way of implicitly modelling the myriad of variables that contribute to the power generation of wind farms. Furthermore, with the use of renewable energy on the rise and wind power being the main contributor, more research needs to be done in this area. From the results of the project, we saw that the distribution was reproduced well but with more states and the ability to handle more data, the results would only increase in accuracy. In South Australia, wind power is already one of the leading producers of electricity; as this will only increase with time, more effective modelling solutions are required to accurately estimate and predict the amount of wind generation.

9 References

[1] Australian Energy Market Operator. Australian Wind Energy Forecasting System. Technical report, 2016. accessed online; `https://www.aemo.com.au/-/media/Files/Electricity/NEM/Security_and_Reliability/Dispatch/Policy_and_Process/2016/Australian-Wind-Energy-Forecasting-System-AWEFS.pdf`.

[2] D. Bhaumik, D. Crommelin, S. Kapodistria, and A. Zwart. Hidden Markov Models for wind farm power output. *IEEE Transactions on Sustainable Energy*, 2018.

[3] A. Churbanov and S. Winters-Hilt. Implementing EM and Viterbi algorithms for Hidden Markov Model in linear memory. Technical report, 2008. accessed online; `https://link.springer.com/article/10.1186/1471-2105-9-224`.

[4] S. Tu. Derivation of Baum-Welch Algorithm for Hidden Markov Models. accessed online; `https://people.eecs.berkeley.edu/~stephentu/writeups/hmm-baum-welch-derivation.pdf`.

[5] L. Xu and M. Jordan. On Convergence Properties of the EM Algorithm for Gaussian Mixtures. accessed online; `http://lasa.epfl.ch/teaching/lectures/ML_Msc/Slides/OnConvergence.pdf`.

[6] J. Yan-Bin. Lagrange Multipliers. Technical report, 2018. accessed online; `http://web.cs.iastate.edu/~cs577/handouts/lagrange-multiplier.pdf`.

10 Appendix

10.1 Derivation of the Baum-Welch Algorithm

This derivation is following the report by Stephen Tu [4] and his arguments are expanded when necessary.

10.1.1 Independent Model

Assume the state space of the underlying hidden process is $S_X = \{1, \dots, M\}$ and the state space of the observations is $S_Y = \{1, \dots, N\}$. Let the vector $\boldsymbol{X} = (x_1, \dots, x_T)$ be the sequence of hidden states the process takes where x_i is the state of the process at time $i \in \{1, \dots, T\}$ with $x_i \in S_X$. Similarly let $\boldsymbol{Y} = (y_1, \dots, y_T)$ be the sequence of observations determined by the process where y_i is the observation inferred by the process at time $i \in \{1, \dots, T\}$ with $y_i \in S_Y$:

$$\begin{aligned}
&\boldsymbol{\theta} = (\boldsymbol{\pi}, P, \delta), \\
&\boldsymbol{\pi} = [\pi_i]_{i \in \{1,\dots,M\}}, && \pi_i = \mathbb{P}(x_1 = i \mid \boldsymbol{\theta}), \\
&P = [P_{i,j}]_{i,j \in \{1,\dots,M\}}, && P_{i,j} = \mathbb{P}(x_{t+1} = j \mid x_t = i, \boldsymbol{\theta}), \\
&\delta = [\delta_i(j)]_{i,j \in \{1,\dots,M\}}, && \delta_i(j) = \mathbb{P}(y_t = j \mid x_t = i, \boldsymbol{\theta}),
\end{aligned}$$

To find the likelihood function $L(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{Y})$,

$$\begin{aligned}
L(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{Y}) &= \mathbb{P}(\boldsymbol{X}, \boldsymbol{Y} \mid \boldsymbol{\theta}) \\
&= \mathbb{P}(x_1, \dots, x_T, y_1, \dots, y_T \mid \boldsymbol{\theta}) \\
&= \mathbb{P}(y_T \mid x_1, \dots, x_T, y_1, \dots, y_{T-1}, \boldsymbol{\theta}) \mathbb{P}(x_1, \dots, x_T, y_1, \dots, y_{T-1} \mid \boldsymbol{\theta}) \\
&= \mathbb{P}(y_T \mid x_1, \dots, x_T, \boldsymbol{\theta}) \mathbb{P}(x_1, \dots, x_T, y_1, \dots, y_{T-1} \mid \boldsymbol{\theta}), \text{ by conditional independence of } y_i, \\
&= \mathbb{P}(y_T \mid x_T, \boldsymbol{\theta}) \mathbb{P}(x_1, \dots, x_T, y_1, \dots, y_{T-1} \mid \boldsymbol{\theta}), \text{ by the Markov property,}
\end{aligned}$$

This can be extended until all y_i have been conditioned out:

$$\begin{aligned}
L(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{Y}) &= \mathbb{P}(y_T \mid x_T, \boldsymbol{\theta}) \dots \mathbb{P}(y_1 \mid x_1, \boldsymbol{\theta}) \mathbb{P}(x_1, \dots, x_T \mid \boldsymbol{\theta}) \\
&= \mathbb{P}(y_T \mid x_T, \boldsymbol{\theta}) \dots \mathbb{P}(y_1 \mid x_1, \boldsymbol{\theta}) \mathbb{P}(x_T \mid x_1, \dots, x_{T-1}, \boldsymbol{\theta}) \mathbb{P}(x_1, \dots, x_{T-1} \mid \boldsymbol{\theta}).
\end{aligned}$$

Then, using the Markov property again, we have

$$L(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{Y}) = \mathbb{P}(y_T \mid x_T, \boldsymbol{\theta}) \dots \mathbb{P}(y_1 \mid x_1, \boldsymbol{\theta}) \mathbb{P}(x_T \mid x_{T-1}, \boldsymbol{\theta}) \mathbb{P}(x_1, \dots, x_{T-1} \mid \boldsymbol{\theta}).$$

This, again, can be extended as follows

$$\begin{aligned}
L(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{Y}) &= \mathbb{P}(y_T \mid x_T, \boldsymbol{\theta}) \dots \mathbb{P}(y_1 \mid x_1, \boldsymbol{\theta}) \mathbb{P}(x_T \mid x_{T-1}, \boldsymbol{\theta}) \dots \mathbb{P}(x_2 \mid x_1, \boldsymbol{\theta}) \mathbb{P}(x_1 \mid \boldsymbol{\theta}) \\
&= \delta_{x_t}(y_t) \dots \delta_{x_1}(y_1) P_{x_{T-1}, x_T} \dots P_{x_1, x_2} \pi_{x_1} \\
&= \pi_{x_1} \delta_{x_1}(y_1) \prod_{t=2}^{T} \delta_{x_t}(y_t) P_{x_{t-1}, x_t}, \\
\ell(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{Y}) &= \log(L(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{Y})) \\
&= \log \left(\pi_{x_1} \delta_{x_1}(y_1) \prod_{t=2}^{T} \delta_{x_t}(y_t) P_{x_{t-1}, x_t} \right) \\
&= \log(\pi_{x_1}) + \sum_{t=1}^{T} \log(\delta_{x_1}(y_1)) + \sum_{t=2}^{T} \log\left(P_{x_{t-1}, x_t}\right), \mathbb{E}_{\boldsymbol{X} \mid \boldsymbol{Y}, \boldsymbol{\theta}^n} \left[\ell(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{Y}) \right] \\
&= \sum_{\boldsymbol{X} \in D} \left[\log(\pi_{x_1}) + \sum_{t=1}^{T} \log(\delta_{x_1}(y_1)) + \sum_{t=2}^{T} \log\left(P_{x_{t-1}, x_t}\right) \right] \mathbb{P}(\boldsymbol{X} \mid \boldsymbol{Y}, \boldsymbol{\theta}^n).
\end{aligned}$$

The constraints placed on the parameters $\boldsymbol{\theta}$ are:

$$\begin{aligned}
\sum_{i=1}^{M} \pi_i - 1 &= 0, \\
\sum_{j=1}^{M} P_{i,j} - 1 &= 0 \ \forall i \in \{1, \dots M\}, \\
\sum_{j=1}^{N} \delta_i(j) - 1 &= 0 \ \forall i \in \{1, \dots M\}.
\end{aligned}$$

Now to maximise this conditional expectation, Lagrange Multipliers will be used [6]:

$$\begin{aligned}
\hat{L} &= \sum_{\boldsymbol{X}\in D}\left[\log\left(\pi_{x_1}\right)+\sum_{t=1}^{T}\log\left(\delta_{x_1}(y_1)\right)+\sum_{t=2}^{T}\log\left(P_{x_{t-1},x_t}\right)\right]\mathbb{P}(\boldsymbol{X}\mid\boldsymbol{Y},\boldsymbol{\theta})-\lambda_\pi\left(\sum_{i=1}^{M}\pi_i-1\right)-\\
&\quad\sum_{i=1}^{M}\lambda_{P_i}\left(\sum_{i=1}^{M}P_{i,j}-1\right)-\sum_{i=1}^{M}\lambda_{\delta_i}\left(\sum_{j=1}^{N}\delta_i(j)-1\right),\\
\frac{\partial\hat{L}}{\partial\pi_i} &= \frac{\partial}{\partial\pi_i}\left[\sum_{\boldsymbol{X}\in D}\log\left(\pi_{x_1}\right)\mathbb{P}(\boldsymbol{X}\mid\boldsymbol{Y},\boldsymbol{\theta}^n)\right]-\lambda_\pi,\\
&= \frac{\partial}{\partial\pi_i}\left[\sum_{\substack{\boldsymbol{X}\in D\\ x_1=i}}\log\left(\pi_i\right)\mathbb{P}(\boldsymbol{X}\mid\boldsymbol{Y},\boldsymbol{\theta}^n)\right]-\lambda_\pi,\ \text{as the derivative of } \log\left(\pi_{x_1}\right) \text{ is only non-zero when } x_1=i,\\
&= \frac{\partial}{\partial\pi_i}\left[\log\left(\pi_i\right)\sum_{\substack{\boldsymbol{X}\in D\\ x_1=i}}\mathbb{P}(x_1=i,x_2=x_2,\dots x_T=x_T\mid\boldsymbol{Y},\boldsymbol{\theta}^n)\right]-\lambda_\pi\\
&= \frac{\partial}{\partial\pi_i}\left[\log\left(\pi_i\right)\mathbb{P}(x_1=i\mid\boldsymbol{Y},\boldsymbol{\theta}^n)\right]-\lambda_\pi\\
&= \frac{\mathbb{P}(x_1=i\mid\boldsymbol{Y},\boldsymbol{\theta}^n)}{\pi_i}-\lambda_\pi.
\end{aligned}$$

Now to maximise the conditional expectation, the partial derivative of the Lagrangian must equal zero:

$$\frac{\mathbb{P}(x_1=i\mid\boldsymbol{Y},\boldsymbol{\theta}^n)}{\pi_i}-\lambda_\pi=0,$$

Which leads to

$$\pi_i=\frac{\mathbb{P}(x_1=i\mid\boldsymbol{Y},\boldsymbol{\theta}^n)}{\lambda_\pi}. \tag{1}$$

Now looking at the partial of $\hat{L}$ with respect to λ_π, equating this to zero and substituting in for π_i gives $\sum_{i=1}^{M}\pi_i-1=0$, which implies

$$\sum_{i=1}^{M}\pi_i=1. \tag{2}$$

Then, by (1) and (2),

$$\begin{aligned}
\sum_{i=1}^{M} \frac{\mathbb{P}(x_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}{\lambda_\pi} &= 1 \\
\Leftrightarrow \sum_{i=1}^{M} \frac{\mathbb{P}(x_1 = i, \boldsymbol{Y} \mid \boldsymbol{\theta})}{\mathbb{P}(\boldsymbol{Y} \mid \boldsymbol{\theta}^n)} &= \lambda_\pi \\
\Leftrightarrow \frac{1}{\mathbb{P}(\boldsymbol{Y} \mid \boldsymbol{\theta})} \sum_{i=1}^{M} \mathbb{P}(x_1 = i, \boldsymbol{Y} \mid \boldsymbol{\theta}^n) &= \lambda_\pi \\
\Leftrightarrow \frac{\mathbb{P}(\boldsymbol{Y} \mid \boldsymbol{\theta}^n)}{\mathbb{P}(\boldsymbol{Y} \mid \boldsymbol{\theta}^n)} &= \lambda_\pi \\
\Rightarrow \lambda_\pi &= 1.
\end{aligned}$$

Therefore,

$$\pi_i = \frac{\mathbb{P}(x_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}{\lambda_\pi} = \mathbb{P}(x_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n).$$

Now looking at $\delta_i(j)$,

$$\frac{\partial \hat{L}}{\partial \delta_i(j)} = \frac{\partial}{\partial \delta_i(j)} \left[\sum_{\boldsymbol{X} \in D} \sum_{t=1}^{T} \log\left(\delta_{x_t}(y_t)\right) \mathbb{P}(\boldsymbol{X} \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \right] - \lambda_{\delta_i}.$$

The partial derivative is only non-zero if $x_t = i$, which reduces the sum to

$$\begin{aligned}
\frac{\partial \hat{L}}{\partial \delta_i(j)} &= \frac{\partial}{\partial \delta_i(j)} \left[\sum_{t=1}^{T} \sum_{\substack{\boldsymbol{X} \in D \\ x_t = i}} \log\left(\delta_i(y_t)\right) \mathbb{P}(x_1 = x_1, \ldots, x_t = i, \ldots, x_T = x_T \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \right] - \lambda_{\delta_i} \\
&= \frac{\partial}{\partial \delta_i(j)} \left[\sum_{t=1}^{T} \log\left(\delta_i(y_t)\right) \sum_{\substack{\boldsymbol{X} \in D \\ x_t = i}} \mathbb{P}(x_1 = x_1, \ldots, x_t = i, \ldots, x_T = x_T \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \right] - \lambda_{\delta_i} \\
&= \frac{\partial}{\partial \delta_i(j)} \left[\sum_{t=1}^{T} \log\left(\delta_i(y_t)\right) \mathbb{P}(x_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \right] - \lambda_{\delta_i} \\
&= \sum_{t=1}^{T} \frac{\mathbb{P}(x_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) I(y_t = j)}{\delta_i(j)} - \lambda_{\delta_i}.
\end{aligned}$$

For a maximum to occur the partial derivative needs to equal zero, that is,

$$\begin{aligned}
\sum_{t=1}^{T} \frac{\mathbb{P}(x_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) I(y_t = j)}{\delta_i(j)} - \lambda_{\delta_i} &= 0 \\
\Rightarrow \delta_i(j) &= \sum_{t=1}^{T} \frac{\mathbb{P}(x_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) I(y_t = j)}{\lambda_{\delta_i}}.
\end{aligned}$$

$$\frac{\partial \hat{L}}{\partial \lambda_{\delta_i}} = -\left(\sum_{j=1}^{N} \delta_i(j) - 1\right) = 0$$

$$\Leftrightarrow \sum_{j=1}^{N} \delta_i(j) = 1$$

$$\Leftrightarrow \sum_{j=1}^{N}\sum_{t=1}^{T} \frac{\mathbb{P}(x_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) I(y_t = j)}{\lambda_{\delta_i}} = 1$$

$$\Leftrightarrow \sum_{t=1}^{T}\sum_{j=1}^{N} \frac{\mathbb{P}(x_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) I(y_t = j)}{\lambda_{\delta_i}} = 1$$

$$\Leftrightarrow \sum_{t=1}^{T} \frac{\mathbb{P}(x_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}{\lambda_{\delta_i}} = 1$$

$$\Rightarrow \lambda_{\delta_i} = \sum_{t=1}^{T} \mathbb{P}(x_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n).$$

Therefore,

$$\delta_i(j) = \sum_{t=1}^{T} \frac{\mathbb{P}(x_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) I(y_t = j)}{\lambda_{\delta_i}},$$

$$\delta_i(j) = \frac{\sum_{t=1}^{T} \mathbb{P}(x_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) I(y_t = j)}{\sum_{t=1}^{T} \mathbb{P}(x_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}.$$

Taking the partial derivative with respect to the state transition probabilities gives

$$\frac{\partial \hat{L}}{\partial P_{i,j}} = \frac{\partial}{\partial P_{i,j}} \left[\sum_{\boldsymbol{X} \in D} \sum_{t=2}^{T} \log\left(P_{x_{t-1}, x_t}\right) \mathbb{P}(\boldsymbol{X} \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\right] - \lambda_{P_i}.$$

After swapping the summations, since the partial derivative of the expression is only non-zero when $x_{t-1} = i$ and $x_t = j$, all other terms can be dropped from the summation, giving rise to

$$\frac{\partial \hat{L}}{\partial P_{i,j}} = \frac{\partial}{\partial P_{i,j}} \left[\sum_{t=2}^{T} \sum_{\substack{\boldsymbol{X} \in D \\ x_{t-1}=i \\ x_t = j}} \log\left(P_{i,j}\right) \mathbb{P}(X_1 = x_1, \ldots, X_{t-1} = i, X_t = j, \ldots, X_T = x_T \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\right] - \lambda_{P_i}.$$

Collapsing the sum using the Law of Total Probability gives

$$\frac{\partial \hat{L}}{\partial P_{i,j}} = \frac{\partial}{\partial P_{i,j}} \left[\sum_{t=2}^{T} \log\left(P_{i,j}\right) \mathbb{P}(X_{t-1} = i, X_t = j \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\right] - \lambda_{P_i}$$

$$= \sum_{t=2}^{T} \frac{\mathbb{P}(X_{t-1} = i, X_t = j \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}{P_{i,j}} - \lambda_{P_i}.$$

In order to maximise the parameters the partial derivative is equated to zero:

$$0 = \sum_{t=2}^{T} \frac{\mathbb{P}(X_{t-1} = i, X_t = j \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}{P_{i,j}} - \lambda_{P_i}$$
$$\Rightarrow P_{i,j} = \sum_{t=2}^{T} \frac{\mathbb{P}(X_{t-1} = i, X_t = j \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}{\lambda_{P_i}}.$$

Now using the constraints:

$$\sum_{j=1}^{N} \sum_{t=2}^{T} \frac{\mathbb{P}(X_{t-1} = i, X_t = j \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}{\lambda_{P_i}} = 1$$
$$\Leftrightarrow \sum_{t=2}^{T} \sum_{j=1}^{N} \mathbb{P}(X_{t-1} = i, X_t = j \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) = \lambda_{P_i}$$
$$\Leftrightarrow \sum_{t=2}^{T} \mathbb{P}(X_{t-1} = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) = \lambda_{P_i}.$$

So therefore the final expression is

$$P_{i,j} = \frac{\sum_{t=2}^{T} \mathbb{P}(X_{t-1} = i, X_t = j \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}{\sum_{t=2}^{T} \mathbb{P}(X_{t-1} = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}.$$

10.1.2 Dependent Model

Again, we are going to assume the state space of the underlying hidden process is $S_X = \{1, \ldots, M\}$ and the state space of the observations is $S_Y = \{1, \ldots, N\}$. Furthermore, the random observation vector is given by $\boldsymbol{Y} = (Y_1, \ldots, Y_T)$ where $\boldsymbol{y} = (y_1, \ldots, y_T)$ is a particular realisation with $y_i \in S_Y$ for all $i \in \{1, \ldots, T\}$. Similarly, the state sequence is a random vector given by $\boldsymbol{X} = (X_1, \ldots, X_T)$ where $\boldsymbol{x} = (x_1, \ldots, x_T)$ is a particular realisation with $x_i \in S_X$ for all $i \in \{1, \ldots, T\}$. In this model the only change in assumption is that each observation is not conditionally independent of the preceding one. Therefore, there is an extra parameter that needs to be accounted for. Let $\delta_{i,k}(j) = \mathbb{P}(y_t = j \mid y_{t-1} = k, X_t = i)$ and $\Delta_{i,j} = \mathbb{P}(y_1 = j \mid X_1 = i)$. Then the dependent model is described completely by $\boldsymbol{\theta} = (\boldsymbol{\pi}, P_{i,j}, \Delta_{i,j}, \delta_{i,k}(j))$. In order to approximate these parameters, we are going to derive the log-likelihood function of the state and observation sequence

$$L(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{Y}) = \mathbb{P}(\boldsymbol{X}, \boldsymbol{Y} \mid \boldsymbol{\theta}),$$
$$= \mathbb{P}(X_1 = x_1, \ldots, X_T = x_T, Y_1 = y_1, \ldots, Y_T = y_T \mid \boldsymbol{\theta}),$$

After applying the definition of conditional probability, we get

$$\mathbb{P}(Y_T = y_T \mid X_1 = x_1, \ldots, X_T = x_T, Y_1 = y_1, \ldots, Y_{T-1} = y_{t-1}, \boldsymbol{\theta}) \mathbb{P}(X_1 = x_1, \ldots, X_T = x_T,$$
$$Y_1 = y_1, \ldots, Y_{T-1} = y_{t-1} \mid \boldsymbol{\theta}).$$

Then, using conditional independence on observations more than one time step before T and states before time T , we can reduce the expression to get

$$\mathbb{P}(Y_T = y_T \mid X_T = x_T, Y_{T-1} = y_{t-1}, \boldsymbol{\theta})\mathbb{P}(X_1 = x_1, \ldots, X_T = x_T, Y_1 = y_1, \ldots, Y_{T-1} = y_{t-1} \mid \boldsymbol{\theta}).$$

We can extend the procedure until the expression becomes

$$\begin{aligned}
&\prod_{t=2}^{T} \mathbb{P}(Y_T = y_T \mid X_T = x_T, Y_{T-1} = y_{t-1}, \boldsymbol{\theta})\mathbb{P}(Y_1 = y_1 \mid X_1 = x_1, \ldots, X_T = x_T, \boldsymbol{\theta}) \times \\
&\mathbb{P}(X_1 = x_1, \ldots, X_T = x_T \mid \boldsymbol{\theta}) \\
=&\prod_{t=2}^{T} \mathbb{P}(Y_T = y_T \mid X_T = x_T, Y_{T-1} = y_{t-1}, \boldsymbol{\theta})\mathbb{P}(Y_1 = y_1 \mid X_1 = x_1, \boldsymbol{\theta})\mathbb{P}(X_1 = x_1, \ldots, X_T = x_T \mid \boldsymbol{\theta}).
\end{aligned}$$

Similarly, we can expand the last term in the expression using conditional independence and remove variables through the Markov property to obtain

$$\mathbb{P}(Y_1 = y_1 \mid X_1 = x_1, \boldsymbol{\theta})\mathbb{P}(X_1 = x_1 \mid \boldsymbol{\theta}) \prod_{t=2}^{T} \mathbb{P}(Y_T = y_T \mid X_T = x_T, Y_{T-1} = y_{t-1}, \boldsymbol{\theta})\mathbb{P}(X_t = x_t, \ldots, X_{t-1} = x_{t-1} \mid \boldsymbol{\theta}).$$

We can see that these probabilities are the same as the parameters defined earlier, and so we can substitute them in for clarity to get

$$L(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{Y}) = \Delta_{x_1,y_1} \pi(x_1) \prod_{t=2}^{T} \delta_{x_t,y_{t-1}}(y_t) P_{x_{t-1},x_t}.$$

Therefore the log-likelihood can be written as

$$\ell(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{Y}) = \log(\Delta_{x_1,y_1}) + \log(\pi(x_1)) + \sum_{t=2}^{T} \log(\delta_{x_t,y_{t-1}}(y_t)) + \log(P_{x_{t-1},x_t}),$$

and after taking the conditional expectation with respect to $\boldsymbol{X} \mid \boldsymbol{Y}, \boldsymbol{\theta}^n$ the expression becomes

$$\mathbb{E}_{\boldsymbol{X}|\boldsymbol{Y},\boldsymbol{\theta}^n}\left[\ell(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{Y})\right] = \sum_{\boldsymbol{X} \in D} \left[\log(\Delta_{x_1,y_1}) + \log(\pi(x_1)) + \sum_{t=2}^{T} \log(\delta_{x_t,y_{t-1}}(y_t)) + \log(P_{x_{t-1},x_t})\right] \mathbb{P}(\boldsymbol{X} \mid \boldsymbol{Y}, \boldsymbol{\theta}^n).$$

Now looking at the constraints placed on the parameters

$$\begin{aligned}
\sum_{i=1}^{M} \pi_i - 1 &= 0, \\
\sum_{j=1}^{M} P_{i,j} - 1 &= 0 \ \forall i \in \{1, \ldots M\}, \\
\sum_{j=1}^{N} \Delta_{i,j} - 1 &= 0 \ \forall i \in \{1, \ldots M\}, \\
\sum_{j=1}^{N} \delta_{i,k}(j) - 1 &= 0 \ \forall i \in \{1, \ldots M\}.
\end{aligned}$$

Let

$$\begin{aligned}\hat{L}() = &\sum_{\boldsymbol{X}\in D}\left[\log(\Delta_{y_1,x_1}) + \log(\pi(x_1)) + \sum_{t=2}^{T}\log(\delta_{x_t,y_{t-1}}(y_t)) + \log(P_{x_{t-1},x_t})\right]\mathbb{P}(\boldsymbol{X}\mid\boldsymbol{Y},\boldsymbol{\theta}^n)\\ &-\lambda_\pi\left(\sum_{i=1}^{M}\pi_i - 1\right) - \sum_{i=1}^{M}\lambda_{P_i}\left(\sum_{j=1}^{M}P_{i,j} - 1\right) - \sum_{i=1}^{M}\lambda_{\Delta_i}\left(\sum_{j=1}^{N}\Delta_{i,j} - 1\right)\\ &-\sum_{i=1}^{M}\sum_{k=1}^{N}\lambda_{\delta_{i,k}}\left(\sum_{j=1}^{N}\delta_{i,k}(j) - 1\right)\end{aligned}$$

be the Lagrangian and we will use this to obtain parameters that maximise the conditional expectation by first calculating each partial derivative. We will first look at π_i and therefore all terms not including this parameter can be removed, resulting in

$$\frac{\partial\hat{L}}{\partial\pi_i} = \frac{\partial}{\partial\pi_i}\left[\sum_{\boldsymbol{X}\in D}\log(\pi_{x_1})\,\mathbb{P}(\boldsymbol{X}\mid\boldsymbol{Y},\boldsymbol{\theta}^n) - \lambda_\pi\left(\sum_{i=1}^{M}\pi_i - 1\right)\right].$$

Now, we know that the derivative of $\sum_{i=1}^{M}\pi_i$ is non-zero only when the dummy variable $i = i$, which reduces the expression to

$$\frac{\partial}{\partial\pi_i}\left[\sum_{\boldsymbol{X}\in D}\log(\pi_{x_1})\,\mathbb{P}(\boldsymbol{X}\mid\boldsymbol{Y},\boldsymbol{\theta}^n)\right] - \lambda_\pi.$$

We can then reduce the expression even further by noticing that any term of the sum that does not include $X_1 = x_1$ will have a derivative of zero, and so we get

$$\frac{\partial}{\partial\pi_i}\left[\sum_{\substack{\boldsymbol{X}\in D\\ x_1=i}}\log(\pi_i)\,\mathbb{P}(X_1 = i, X_2 = x_2,\ldots,X_T = x_T\mid\boldsymbol{Y},\boldsymbol{\theta}^n)\right] - \lambda_\pi$$

$$= \frac{\partial}{\partial\pi_i}\left[\log(\pi_i)\,\mathbb{P}(X_1 = i\mid\boldsymbol{Y},\boldsymbol{\theta}^n)\right] - \lambda_\pi, \tag{3}$$

$$= \frac{\mathbb{P}(X_1 = i\mid\boldsymbol{Y},\boldsymbol{\theta}^n)}{\pi_i} - \lambda_\pi. \tag{4}$$

Next we take the derivative with respect to λ_π, so we get

$$\frac{\partial\hat{L}}{\partial\lambda_\pi} = -\left(\sum_{i=1}^{M}\pi_i - 1\right).$$

Taking the derivative with respect to $P_{i,j}$ gives

$$\frac{\partial\hat{L}}{\partial P_{i,j}} = \frac{\partial}{\partial P_{i,j}}\left[\sum_{\boldsymbol{X}\in D}\sum_{t=2}^{T}\log\left(P_{x_{t-1},x_t}\right)\mathbb{P}(\boldsymbol{X}\mid\boldsymbol{Y},\boldsymbol{\theta}^n)\right] - \lambda_{P_i}$$

and following the thought process from equation (3) the non-zero terms will occur when $x_{t-1} = j$ and $x_t = i$, which reduces the RHS to

$$\frac{\partial}{\partial P_{i,j}} \left[\sum_{\substack{\boldsymbol{X} \in D \\ x_{t-1}=i \\ x_t=j}} \sum_{t=2}^{T} \log\left(P_{i,j}\right) \mathbb{P}(X_1 = x_1, \ldots, X_{t-1} = i, X_t = j, \ldots, X_T = x_t \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \right] - \lambda_{P_i}.$$

Since both summations are finite, they can be switched which then allows us to use the Law of Total Probability to simplify the expression and obtain

$$\begin{aligned} &\frac{\partial}{\partial P_{i,j}} \left[\sum_{t=2}^{T} \log\left(P_{i,j}\right) \mathbb{P}(X_{t-1} = i, X_t = j \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \right] - \lambda_{P_i} \\ &= \sum_{t=2}^{T} \frac{\mathbb{P}(X_{t-1} = i, X_t = j \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}{P_{i,j}} - \lambda_{P_i}. \end{aligned}$$

For λ_{P_i} the partial derivative is

$$\frac{\partial \hat{L}}{\partial \lambda_\pi} = -\left(\sum_{j=1}^{M} P_{i,j} - 1 \right).$$

Next, we look at $\Delta_{i,j}$ which gives

$$\frac{\partial \hat{L}}{\partial \Delta_{i,j}} = \frac{\partial}{\partial \Delta_{i,j}} \left[\sum_{\boldsymbol{X} \in D} \log\left(\Delta_{x_1,y_1}\right) \mathbb{P}(\boldsymbol{X} \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \right] - \lambda_{\Delta_i}.$$

Again, we can reduce the sum by noticing that non-zero terms will occur only if $y_1 = j$ and $x_1 = i$. However, $\boldsymbol{Y}$ is an observed sequence and so we will use the indicator function $\mathbb{I}$ which results in

$$\begin{aligned} &\frac{\partial}{\partial \Delta_{i,j}} \left[\sum_{\substack{\boldsymbol{X} \in D \\ x_1=i}} \log\left(\Delta_{i,j}\right) \mathbb{P}(X_1 = i, X_2 = x_2, \ldots, X_T = x_T \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \mathbb{I}(y_1 = j) \right] - \lambda_{\Delta_i}, \\ &= \frac{\partial}{\partial \Delta_{i,j}} \left[\log\left(\Delta_{i,j}\right) \mathbb{P}(X_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \mathbb{I}(y_1 = j) \right] - \lambda_{\Delta_i}, \\ &= \frac{\mathbb{P}(X_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \mathbb{I}(y_1 = j)}{\Delta_{i,j}} - \lambda_{\Delta_i}. \end{aligned}$$

For λ_{Δ_i} the derivative reduces to

$$\frac{\partial}{\partial \lambda_{\Delta_i}} = -\left(\sum_{j=1}^{N} \Delta_{i,j} - 1 \right).$$

Finally, the partial derivative of $\hat{L}$ with respect to $\delta_{i,k}(j)$ is

$$\frac{\partial \hat{L}}{\partial \delta_{i,k}(j)} = \frac{\partial}{\partial \delta_{i,k}(j)} \left[\sum_{\boldsymbol{X} \in D} \sum_{t=2}^{T} \log(\delta_{x_t,y_{t-1}}(y_t)) \mathbb{P}(\boldsymbol{X} \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \right] - \lambda_{\delta_i(j)}.$$

We can see that non-zero terms will occur only when $x_t = i$, $y_{t-1} = k$ and $y_t = j$. Again, because $\boldsymbol{Y}$ is an observed vector, we will use the indicator function to ensure that the derivative is non-zero, and so substituting in, we get

$$\frac{\partial}{\partial \delta_{i,k}(j)} \left[\sum_{\substack{\boldsymbol{X} \in D \\ x_t = i}} \sum_{t=2}^{T} \log(\delta_{i,k}(j)) \mathbb{P}(X_1 = x_1, \dots X_t = i, \dots, X_T = x_T \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \mathbb{I}(y_{t-1} = k, y_t = j) \right] - \lambda_{\delta_{i,k}}.$$

Since both summations are finite we can swap them and then use the Law of Total Probability to leave the expression as

$$\begin{aligned} &\frac{\partial}{\partial \delta_{i,k}(j)} \left[\sum_{t=2}^{T} \log(\delta_{i,k}(j)) \mathbb{P}(X_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \mathbb{I}(y_{t-1} = k, y_t = j) \right] - \lambda_{\delta_{i,k}} \\ &= \sum_{t=2}^{T} \frac{\mathbb{P}(X_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \mathbb{I}(y_{t-1} = k, y_t = j)}{\delta_{i,k}(j)} - \lambda_{\delta_{i,k}}. \end{aligned}$$

For $\lambda_{\delta_{i,k}}$ the derivative reduces to

$$\frac{\partial}{\partial \lambda_{\delta_{i,k}}} = -\left(\sum_{j=1}^{N} \delta_{i,k}(j) - 1 \right).$$

We then equate each partial derivative to zero which yields the set of equations

$$\begin{aligned} \frac{\mathbb{P}(X_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}{\pi_i} - \lambda_\pi &= 0, \\ -\left(\sum_{i=1}^{M} \pi_i - 1 \right) &= 0, \\ \sum_{t=2}^{T} \frac{\mathbb{P}(X_{t-1} = i, X_t = j \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}{P_{i,j}} - \lambda_{P_i} &= 0, \\ -\left(\sum_{j=1}^{M} P_{i,j} - 1 \right) &= 0, \\ \frac{\mathbb{P}(X_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \mathbb{I}(y_1 = j)}{\Delta_{i,j}} - \lambda_{\Delta_i} &= 0, \\ -\left(\sum_{j=1}^{N} \Delta_{i,j} - 1 \right) &= 0, \\ \sum_{t=2}^{T} \frac{\mathbb{P}(X_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) \mathbb{I}(y_{t-1} = k, y_t = j)}{\delta_{i,k}(j)} - \lambda_{\delta_{i,k}} &= 0, \\ -\left(\sum_{j=1}^{N} \delta_{i,k}(j) - 1 \right) &= 0. \end{aligned}$$

It turns out that $\Delta_{i,j}$ and $\delta_{i,k}(j)$ are the only parameters that have a different expression compared to the independent model. If we look at $\Delta_{i,j}$ we get

$$\frac{\mathbb{P}(X_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\mathbb{I}(y_1 = j)}{\lambda_{\Delta_i}} = \Delta_{i,j}.$$

Now, substituting into the other equation results in

$$\sum_{j=1}^{N} \frac{\mathbb{P}(X_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\mathbb{I}(y_1 = j)}{\lambda_{\Delta_i}} = 1,$$

$$\Leftrightarrow \mathbb{P}(X_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n) = \lambda_{\Delta_i}.$$

Therefore, the final estimate expression is

$$\Delta_{i,j} = \frac{\mathbb{P}(X_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\mathbb{I}(y_1 = j)}{\mathbb{P}(X_1 = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)}.$$

Finally,

$$\delta_{i,k}(j) = \sum_{t=2}^{T} \frac{\mathbb{P}(X_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\mathbb{I}(y_{t-1} = k, y_t = j)}{\lambda_{\delta_{i,k}}}.$$

Now, substituting into the other equation results in

$$\sum_{j=1}^{N}\sum_{t=2}^{T} \frac{\mathbb{P}(X_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\mathbb{I}(y_{t-1} = k, y_t = j)}{\lambda_{\delta_{i,k}}} = 1,$$

$$\sum_{t=2}^{T} \mathbb{P}(X_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\mathbb{I}(y_{t-1} = k) = \lambda_{\delta_{i,k}}.$$

Therefore the final estimate expression is

$$\delta_{i,k}(j) = \frac{\sum_{t=2}^{T} \mathbb{P}(X_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\mathbb{I}(y_{t-1} = k, y_t = j)}{\sum_{t=2}^{T} \mathbb{P}(X_t = i \mid \boldsymbol{Y}, \boldsymbol{\theta}^n)\mathbb{I}(y_{t-1} = k)}.$$

10.2 Matlab Code

10.2.1 Alpha function matrix

```
function [result,ALPHA] = forwardalgorithmvector(t, i, N,initdist,P,delta
    ,y)
%Calculate P( Y_1 = y_1, ..., Y_t = y_t, X_t = i | theta): the
    probability that Y_1 = y_1, ..., Y_t = y_t (the observation sequence
    up to time t) given the
%parameters and X_t = i (the state at time t is i).
```

```
% t = vector of whole numbers from 1 to the final time point
% i = a vector of whole numbers from 1 to the number of states
% N = the number of states
% initdist = the row vector containing the initial distribution
% probabilities across all the states
% P = the state transition matrix
% delta = a matrix where the i,j-th element is the probability of seeing
% observation i given the system is in state j
% y = the observation sequence
%OUTPUT: the matrix alpha(i,t)  where alpha(i,t) is P( Y_1 = y_1, ...,
    Y_t = y_t, X_t = i | theta)
%METHOD: the calculation is done recursively, with each recursion done
    with
%t(1:end-1)
if t(end)==1
    %calculates the starting value in the recursion for all states
     result = (initdist.*delta(y(1),i))';
else
    [X,ALPHA] = forwardalgorithmvector(t(1:end-1),i,N,initdist,P,delta,y)
        ;

    result = (sum(repmat(X,[1,N]).*P).*(delta(y(t(end)),i)))';
    %calculates the column vector of ALPHA(i,t(end)) for all states
end
ALPHA(1:N,length(t)) = result;
end
```

10.2.2 Beta function matrix

```
function [result,BETA] = backwardalgorithmvector(t, i, N,initdist,P,delta
    ,y)
%Calculate P( Y_(t+1) = y_(t+1), ..., Y_T = y_T | X_t = i, theta): the
    probability that Y_(t+1) = y_(t+1), ..., Y_T = y_T (the observation
```

```
    sequence from time t+1 to time T) given the
%parameters and X_t = i (the state at time t is i).
% t = vector of whole numbers from 1 to the final time point
% i = a vector of whole numbers from 1 to the number of states
% N = the number of states
% initdist = the row vector containing the initial distribution
% probabilities across all the states
% P = the state transition matrix
% delta = a matrix where the i,j-th element is the probability of seeing
% observation i given the system is in state j
% y = the observation sequence
%OUTPUT: the matrix BETA(i,t)  where alpha(i,t) is P( Y_1 = y_1, ..., Y_t
    = y_t | X_t = i, theta)
%METHOD: the calculation is done recursively, with each recursion done
    with
%t(2:end)
if t(1)==length(y)
    %initialises the recursion with BETA(i,T) equalling 1 for all states
result = ones(N,1);
else
    [X,BETA] = backwardalgorithmvector(t(2:end),i,N,initdist,P,delta,y);
    result = sum(P.*repmat((delta(y(t(2)),i)),[N,1]).*repmat(X',[N,1]),2)
        ;
    %calculates the column vector of BETA(i,t(end)) for all states
end
BETA(1:N,(length(y) +1 - length(t))) = result;
end
```

10.2.3 Zeta function matrix

```
function [E] = epsilonalgorithmvector(N,P,delta,y,ALPHA,BETA,prob_y_theta
    )
% Calculates the zeta function matrix
```

```
% N = the number of hidden states
% P = the state transition matrix
% delta = a matrix where the i,j-th element is the probability of seeing
% y = the observation sequence
% ALPHA = the alpha function matrix
% BETA = the beta function matrix
% observation i given the system is in state j
% prob_y_theta = the likelihood of the current observation sequence
%OUTPUT: the zeta function matrix
%METHOD: the calculation is done using mainly repmat and sum commands to
%ensure it is vectorised
temp1=1:length(y);

temp2 = repmat(ALPHA(:,1:end-1),[1,N]).*repmat(reshape(delta(y(temp1(2:
    end)),:),[1,N*(length(y)-1)]),[N,1]).*repmat(reshape(BETA(:,2:end)
    ',[1,N*(length(y)-1)]),[N,1]);
E = (reshape(sum((reshape(temp2',[(length(y)-1),N*N]))',2),[N,N])'.*P)./
    prob_y_theta;


end
```

10.2.4 Observation-state probability matrix

```
function [delta] = DeltaApproximation(t, N,GAMMA, E, M, y)
 %Calculates the observation-state probability matrix
% t = vector of whole numbers from 1 to the final time point
% N = the number of hidden states
% GAMMA = the gamma matrix
% E = the zeta function matrix
% M = number of observation states
% y = the observation sequence
%OUTPUT: the observation-state probability matrix
```

```
%METHOD: the calculation is done using mainly repmat and sum commands to
%ensure it is vectorised
temp = repmat(GAMMA(:,t(1:end-1)),[1,M]).*(repmat(repmat(y(1:end-1),[N
    ,1]),[1,M])== reshape(repmat(1:M,[length(y)-1,1]),[1,M*(length(y)-1)]
    );

delta = reshape(sum(reshape(temp',[(length(y)-1)],M*N)',2),[M,N])./repmat
    ((sum(GAMMA(:,t(1:end-1)),2))',[M,1]);
end
```

10.2.5 State transition matrix

```
function [P] = P_ijApproximation(t,N,GAMMA,E)
% Calculate state transition matrix
% t = vector of whole numbers from 1 to the final time point
% N = the number of states
% GAMMA = the gamma matrix
% E = the zeta function matrix
%OUTPUT: the state transition matrix
P = E./(repmat(sum(GAMMA(:,t(1:end-1)),2),[1,N]));
end
```

10.2.6 EM algorithm

```
function [prob_y_theta_2,initdist,P,delta] = EM(y,init,P_0,delta_0,N,M)
format long
% Calculates the model parameters that maximise the likelihood of the
% observation sequence given an intital parameter estimate
% y = the observation sequence
% init = the initial row vector containing the initial distribution
% probabilities across all the states
% P_0 = the initial state transition matrix
% delta_0 = the initial observation-state probability matrix
%OUTPUT: the model parameters that maximise the likelihood of the
% observation sequence given an intital parameter estimate
```

```
12
13 thereallength = 1:length(y);
14 [~,ALPHA] = forwardalgorithmvector(thereallength, 1:N, N,init,P_0,delta_0
       ,y);
15 %finds the alpha function matrix
16
17 [~,BETA] = backwardalgorithmvector(thereallength, 1:N, N,init,P_0,delta_0
       ,y);
18 %finds the beta function matrix
19
20 prob_y_theta=sum(ALPHA(:,length(y)));
21 %finds the current likelihood using the current parameter estimates
22
23 GAMMA= (ALPHA.*BETA)./(prob_y_theta);
24 %calculates the gamma matrix
25
26 E = epsilonalgorithmvector(N,P_0,delta_0,y,ALPHA,BETA,prob_y_theta);
27 %calculates the zeta matrix
28
29 P = P_ijApproximation(thereallength, N, GAMMA, E);
30 %calculates the updated state transition matrix
31
32 initdist =GAMMA(:,1)';
33 %calculates the updated initial distribution
34
35
36 delta = DeltaApproximation(1:length(y), N,GAMMA, E, M, y);
37 %calculates the updated observation-state matrix
38
39
40 [~,ALPHA2] = forwardalgorithmvector(1:length(y), 1:N, N,initdist,P,delta,
       y);
```

```
41 prob_y_theta_2=sum(ALPHA2(:,length(y)));
42 %calculates the new likelihood probability given the updated parameters
43
44 if abs(log(prob_y_theta_2) - log((prob_y_theta))) > 0.000000000000000001
45     EM(y,initdist,P,delta,N,M);
46 end
47 %checks whether the difference in likelihoods is within the set tolerance
48 end
```