Identification of translational regulation elements in mRNA

Jiayuan Zhu

15/6/2021

Abstract

In contrast to our knowledge of transcriptional regulation, much less is known about translational regulation. A particularly intriguing model of translational regulation involves upstream open reading frames (uORFs). This project develops a statistical model to identify the locations of translated uORFs and main coding regions within the messenger RNA (mRNA). It would further facilitate systematic investigations of uORFs regulator functions.

After studying the characteristics of ribosome profiling data, a hidden Markov model (HMM) is developed for the comprehensive identification of uORFs and main coding regions within the mRNA. Then it is implemented in Python. The estimation of model parameters involves the Expectation-Maximization (EM) algorithm. Identification of the locations of translated regions is conducted by the Viterbi algorithm. Besides, thirty simulation datasets that each contains one hundred mRNA sequences are constructed to verify model correctness.

1 Introduction

1.1 Background

Understanding how gene expression is regulated is fundamental to modern molecular biology research. As multiple proteins can be traced to a single gene [8], when, where, and how much of a protein is expressed is important to explain the underlying biological processes. Proteins are made of large numbers of amino acid. Although there are only 20 types of amino acids, different concatenating combinations to form amino acid chains and diverse folding up ways to construct three-dimensional molecules with complex shapes result in various types of proteins.

According to the central dogma 1, protein synthesis consists of two steps which are transcription and translation. During transcription, genetic instructions are transferred from DNA to messenger RNA (mRNA). Then ribosome moves along the mRNA sequence during the translation to produce protein. Much of the research on gene regulation has been focusing on transcription. However, transcript levels do not always reflect the level of protein expression. Therefore, it is necessary to investigate more about translation to gain a full picture.

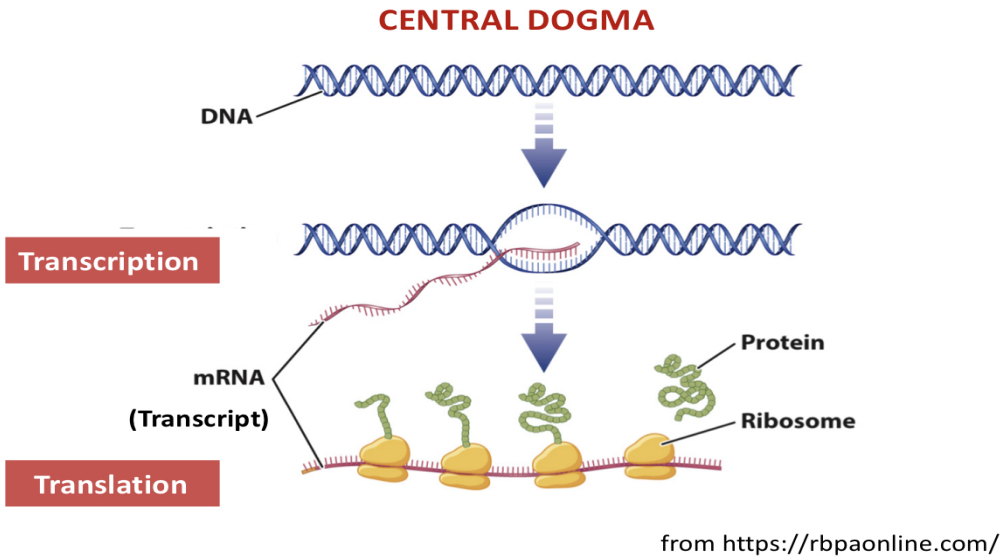

Figure 1: Central Dogma for protein synthesis

One major determinant of protein expression level is the rate of protein translation. A particularly intriguing model of translational regulation involves upstream open reading frames (uORFs). Although translation at uORFs has been known to affect translation of main coding regions during important biological processes [1][7], limited knowledge of translated uORFs impedes the investigation of uORFs regulatory functions.

Therefore, this project develops a statistical model to identify the locations of translated uORFs and main coding regions within the mRNA.

In the following sections, some necessary technical biology terms will be explained. It includes translation mechanism, ribosome profiling data and upstream open reading frames (uORFs).

1.2 Translation Mechanism

During translation, ribosomes read the information carried in mRNA. It is illustrated in 2 that a small ribosome subunit (named as 40S) is attached to the mRNA sequence first. Then it scans through the mRNA to find the Kozak consensus sequence which indicates the start site (e.g. AUG). When it finds a start codon, it assembles with a large ribosome subunit (named as 60S) to form a ribosome. In the elongation process, the ribosome moves 3 base pairs (a codon) each time. One corresponding amino acid is brought by transport RNA (tRNA) when a codon is decoded. It combines amino acid subunits with peptide bonds to form an amino acid chain. As soon as the ribosome meets a stop codon (e.g UAA), the ribosome separates into two subunits and is then released. Meanwhile, the amino acid chain is released and after some transformation, it will form a protein. It is noticeable that each mRNA can have multiple ribosomes to translate at the same time.

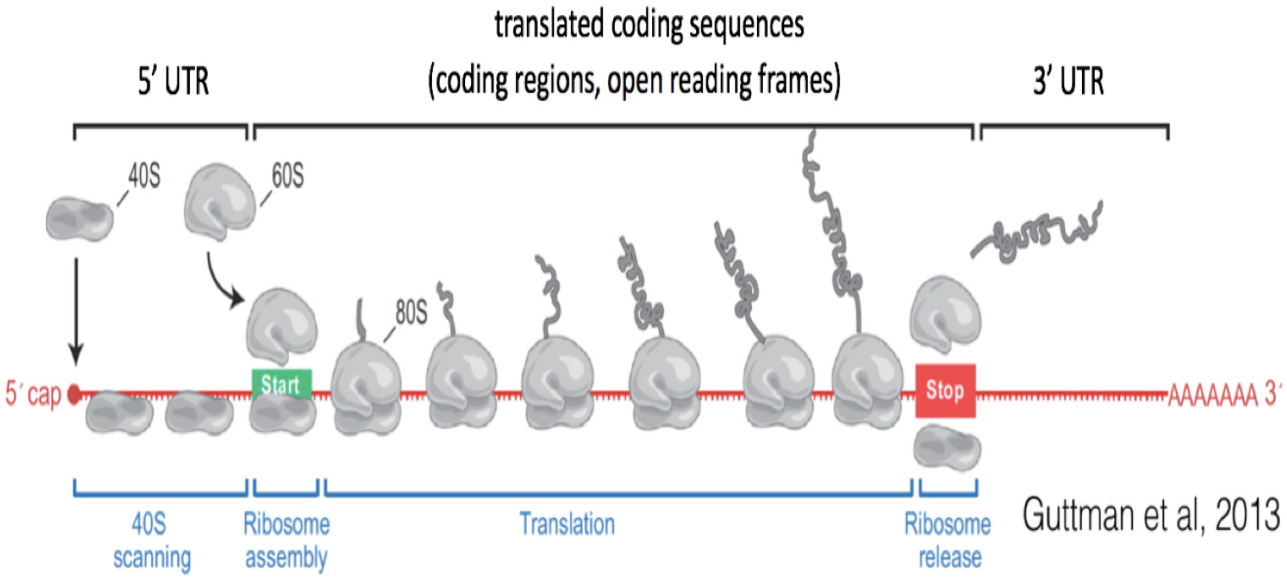

Figure 2: Translation Mechanism

The region where relates to amino acid chain formation is called the coding region. The regions before and after it are usually untranslated, named as 5' untranslated region (5'UTR) and 3' untranslated region (3'UTR) respectively.

Although GENCODE provides some annotations for protein coding regions, it assumes that coding genes have only one long and conserved coding region and annotates others as non-coding. However, according to Pelechano V. et al. [6], multiple coding regions in mRNA are possible in yeast. Besides, Lauressergues D. et al. [4] states that functional short regions can exist in primary transcripts of microRNAs (pri-miRNA) which also contradicts GENCODE's annotation criteria. Therefore, a more advanced approach to identify relatively unbiased coding regions is necessary.

1.3 Ribosome Profiling Data

It is unable to identify the coding regions with mRNA sequences alone as they only contain different bases. However, multiple mRNA can be achieved from DNA after transcription and several ribosomes can translate at mRNA simultaneously. Hence, ribosome occupancy would provide extra information from mRNA sequences.

Ribosome profiling which is a deep-sequencing-based-tool to facilitate the detailed measurement of translation [3] is utilized to identify coding regions. First, ribosome-protected fragments (ribosome footprints) are sequenced to indicate positions of active ribosomes by nucleases [3]. Afterwards, these fragments are isolated and converted to DNA molecular library for deep sequencing [5]. Finally, ribosome profiling data can be obtained by measuring the density of ribosome footprints and mapping them back to the original mRNA sequence. It is a sequence of data representing counts for each base in mRNA.

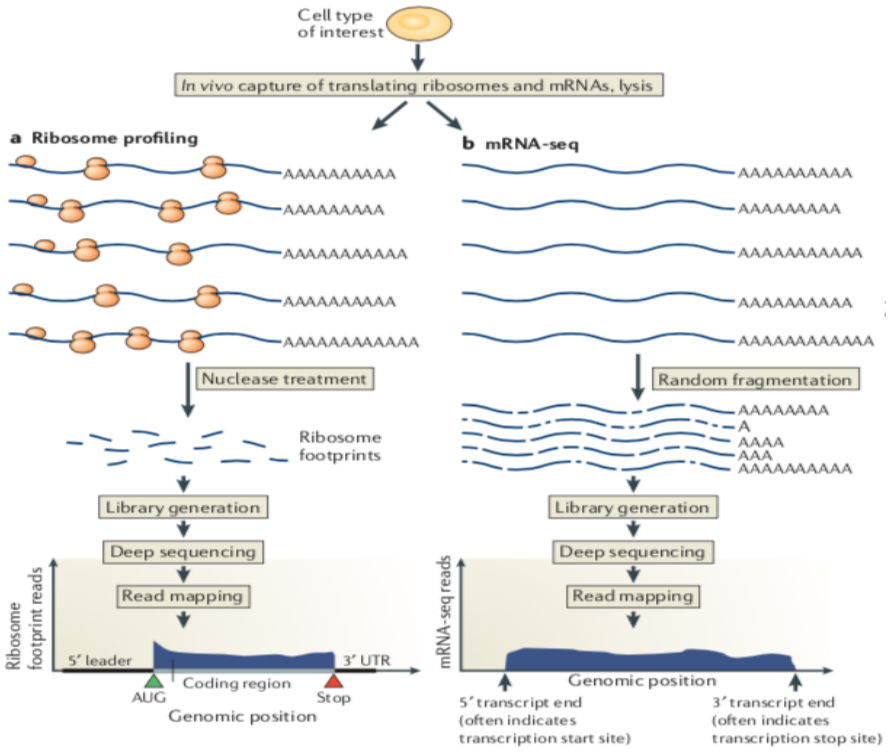

Figure 3: Ribosome Profiling Process. From [3]

The figure 4 below illustrates an example of ribosome profiling data. The top subgraph is related to translated region's start site and the translation starts from position 0. The subgraph at the bottom corresponds to stop site's translated region and the translation terminates at position 0. It is obvious that within the translated region, the proportion of ribosome footprints is higher. The proportion is even greater for start and stop positions. Furthermore, it follows the three-base periodicity in translated region.

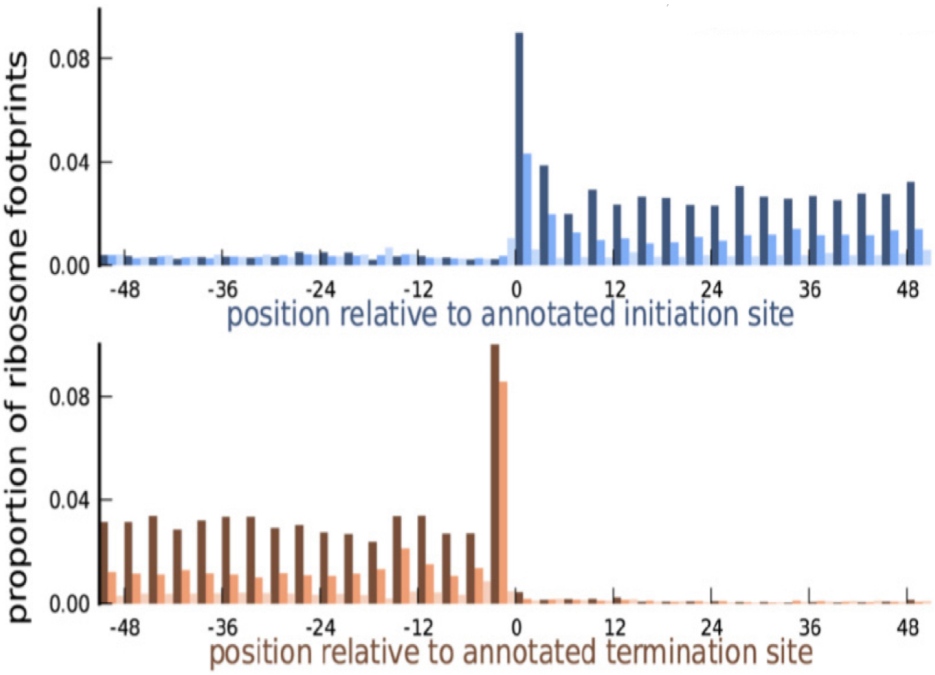

Figure 4: Ribosome profiling data characteristics

1.4 Upstream Open Reading Frames

It is mentioned in the Translation Mechanism 1.2 that 5'UTR is located before the coding region and is normally untranslated. Nevertheless, pairs of start and stop codons can exist in 5'UTR which generate upstream open reading frames (uORFs) in that region.

As a result, there are two possible translated regions in mRNA which are uORFs and the main coding region. According to Calvo SE. et al. [2], translation at uORFs is known to affect translation at the main coding region. So it is critical to identify the locations of both translated regions within the mRNA.

The following plot 5 shows the four possible cases:

1. Translation only exists in the main coding region
2. Translation only exists in uORFs
3. Translations exist in both uORFs and the main coding region
4. No translation exists in either uORFs or the main coding region

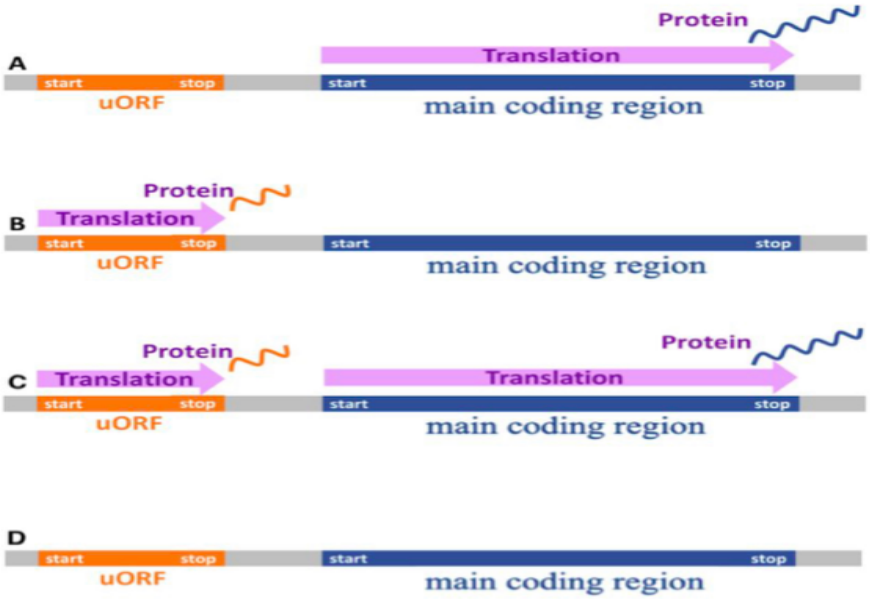

Figure 5: Four possible translated cases in mRNA

2 Hidden Markov Model

Ribosome profiling data is a sequence of counts in mRNA to measure how frequently each base has been translated. According to the characteristics described in Ribosome Profiling Data 1.3, it carries unobserved information about translation. Different ribosome footprint counts are highly relates to the bases. Hence, a hidden Markov model (HMM) is proposed to identify translated uORFs and the main coding region.

2.1 Hidden States

Twenty-one hidden states are decided for this model and they can be separated into two major groups 6. One is related to untranslated regions and it contains three states: 5'U, 5'U2 and 3'U (grey stripes). They represent the untranslated region before translated uORFs, in between translated uORFs and main coding region, after main coding region respectively. Another group is corresponding to translated regions. It can be divided into two subclasses that one for uORFs and the other for the main coding region. The figure 4 demonstrates that initiation and termination sites achieve different counts than elongation parts and they all follow a three-base periodicity pattern. Therefore, for every subclass, 3 states are allocated to initiation, elongation, termination each. Notations for the main coding region are initiation (I1, I2, I3), elongation (E1, E2, E3) and termination (T1, T2, T3). Similarly, translated uORFs are notated with initiation (uI1, uI2, uI3), elongation (uE1, uE2, uE3) and termination (uT1, uT2, uT3).

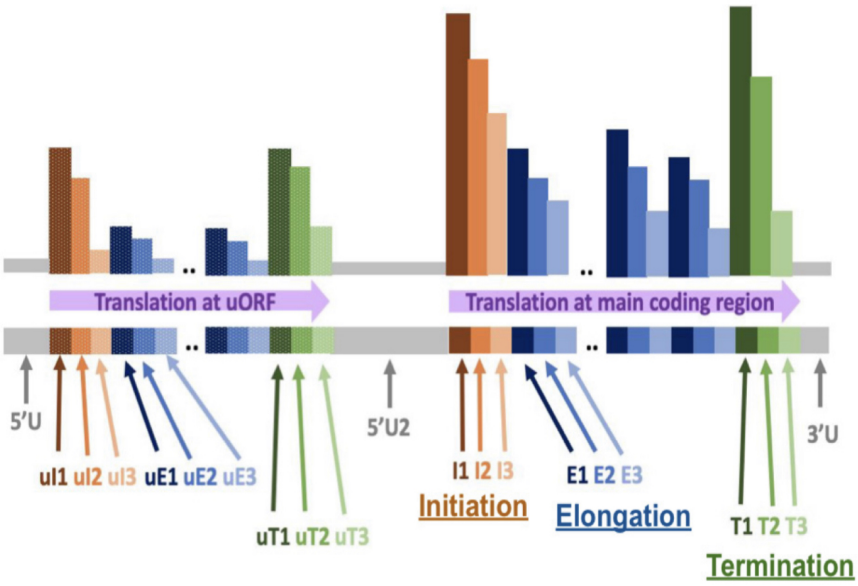

Figure 6: Twenty-one states for HMM

2.2 Model Assumption

It is assumed that every mRNA sequence would start from state 5'U and stay in that state for some time. Besides, the coding region can start at any codon, not restricted to canonical start codon AUG. However, it must terminate at the first in-frame canonical stop codon which includes UAA, UGA and UAG. When the ribosome enters any translated coding region, every movement is based on three bases (one codon). It would experience one codon for initiation, several codons for elongation and one codon for termination. Afterwards, it should stay in the untranslated region for a while. The ribosome from the main coding region will enter state 3'U and the ribosome from uORFs would enter state 5'U2. After staying in 5'U2 for a while, it may enter the main coding region to translate again.

2.3 Transition Probability

To achieve the four scenarios proposed previously 1.4, transition probability is introduced as the following diagram 7. Although there are twenty-one hidden states, most of them are deterministic. All grey arrows in the transition diagram indicate probability 1. For example, if the current state is uI1, the ribosome should move along the sequence with specific order uI2, uI3, uE1, uE2, uE3.

θ_μ and θ are parameters related to the termination of the translation region. If the ribosome has already entered coding region and the next three bases form a canonical stop codon (i.e. UAA, UGA or UAG), then $\theta_\mu = \theta = 1$, otherwise, $\theta_\mu = \theta = 0$.

Because coding region can start at any start codon ci, there are three parameters corresponding to each ci, namely ρ_μ^{ci}, ρ^{ci} and δ^{ci}. For the specific codon ci, the ribosome has probability ρ_μ^{ci} and ρ^{ci} to start uORFs or the main coding region from 5'U respectively. It has likelihood δ^{ci} to enter main coding region from 5'U2.

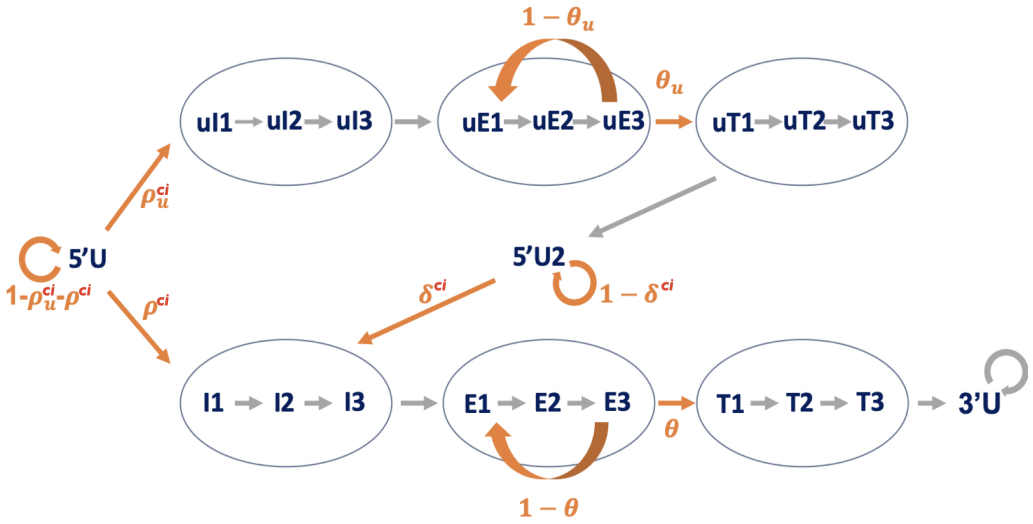

Figure 7: Transition Diagram

2.4 Emission Probability

In the hidden Markov model, the emission probability is the conditional probability of observed data given the hidden state.

In this project, the observed data is ribosome profiling data which contains a sequence of counts and the hidden states have been described in 2.1. With the non-negative counts, a natural choice is to apply the Poisson distribution. Nonetheless, Poisson distribution guarantees the same mean and variance value. It may lead to overdispersion. In order to avoid this issue, the rate parameter in Poisson distribution is randomly generated by Gamma distribution whose parameters are depending on the given hidden state. Then the ribosome profiling data can be further derived from the Poisson distribution.

Furthermore, difference in ribosome footprint abundance across transcripts is corrected according to transcript expression levels by the transcript-specific normalization factor.

3 Model Set Up

Consider a set of N mRNA sequences that are assumed to be independent. The n^{th} sequence has length $T^{(n)}$. Let $\mathbf{X^{(n)}} = (x_1^{(n)}, \ldots, x_{T^{(n)}}^{(n)})$ and $\mathbf{S^{(n)}} = (s_1^{(n)}, \ldots, s_{T^{(n)}}^{(n)})$ where $x_t^{(n)}$ and $s_t^{(n)}$ represent the ribosome footprint count and the base at the t^{th} position in the n^{th} mRNA sequence. Hence, $\mathbf{X} = (\mathbf{X^{(1)}}, \ldots, \mathbf{X^{(N)}})$ and $\mathbf{S} = (\mathbf{S^{(1)}}, \ldots, \mathbf{S^{(N)}})$ store detailed information for N mRNA sequences. $z_t^{(n)}$ represents the hidden state for the t^{th} position in the n^{th} sequence. So notations $\mathbf{Z^{(n)}}$ and $\mathbf{Z}$ can be understood in the similar way.

Furthermore, $\mathbf{E} = (E^{(1)}, \ldots, E^{(N)})$ denotes the gene expression levels for different sequences. To be convenient, 21 states (5'U, uI1, uI2, uI3, uE1, uE2, uE3, uT1, uT2, uT3, 5'U2, I1, I2, I3, E1, E2, E3, T1, T2, T3, 3'U) are labelled by 1 to 21.

3.1 Emission Distribution

The emission distribution involves both Poisson distribution and Gamma distribution 2.4:

$$\begin{aligned} X_{z,t}^{(n)} \mid Z_t^{(n)} &\sim Poi(\mu_{z,t}^{(n)} E^{(n)}) \\ \mu_{z,t}^{(n)} &\sim Gamma(\alpha_z, \beta_z) \end{aligned}$$

The density of Poisson distribution is

$$g(x_{z,t}^{(n)} \mid z_t^{(n)}) = \frac{(\mu_{z,t}^{(n)} \cdot E^{(n)})^{x_{z,t}^{(n)}}}{x_{z,t}^{(n)}!} e^{-\mu_{z,t}^{(n)} E^{(n)}}$$

The density of Gamma distribution is

$$g(\mu_{z,t}^{(n)}) = \frac{\beta_z^{\alpha_z}}{\Gamma(\alpha_z)} \cdot \mu_{z,t}^{(n)(\alpha_z - 1)} \cdot e^{-\beta_z \cdot \mu_{z,t}^{(n)}}$$

Then it can be proved that $X_{z,t}^{(n)}$ follows a Negative Binomial distribution 6.1.

$$X_{z,t}^{(n)} \sim NB(\alpha_z, \frac{\beta_z}{E^{(n)} + \beta_z})$$

3.2 Log Likelihood

Log likelihood is useful when estimating model parameters. Recall that ρ_μ^{ci}, ρ^{ci} are the probabilities transferring from state 1 to state 2 or state 12 according to specific codon ci. δ^{ci} represents the probability from state 11 to state 12. Ω provides the set of all possible start codons. Note that θ is current model's parameters. The likelihood for a single sequence is derived first and ore derivation details is in Appendix1 6.2.

$$\begin{aligned} logP(\mathbf{x_{1:T}}, \mathbf{z_{1:T}} \mid \theta, \mathbf{s_{1:T}}) \propto & \sum_{t=1}^{T} \sum_{m=1}^{M} I(z_t = m) \cdot logP(x_t \mid \alpha_m, \beta_m, z_t = m) \\ & + \sum_{m=1}^{M} I(z_1 = m) \cdot logP(z_1 \mid \theta) \\ & + \sum_{t=2}^{T} \sum_{ci \in \Omega} [I(z_{t-1} = 1, z_t = 1, s_{s:s+2} = ci) \cdot log(1 - \rho_\mu^{ci} - \rho^{ci}) \\ & \quad + [I(z_{t-1} = 1, z_t = 2, s_{s:s+2} = ci) \cdot log\rho_\mu^{ci} \\ & \quad + [I(z_{t-1} = 1, z_t = 12, s_{s:s+2} = ci) \cdot log\rho^{ci} \\ & \quad + [I(z_{t-1} = 11, z_t = 11, s_{s:s+2} = ci) \cdot log(1 - \delta^{ci}) \\ & \quad + [I(z_{t-1} = 11, z_t = 12, s_{s:s+2} = ci) \cdot log\delta^{ci}] \end{aligned}$$

As mRNA sequences are assumed to be independent, so the log likelihood for multiple mRNA sequences can be easily extended as follows:

$$\begin{aligned}\sum_{n=1}^{N} logP(\mathbf{x_{1:T}}, \mathbf{z_{1:T}} \,|\, \theta, \mathbf{s_{1:T}}) \propto & \sum_{n=1}^{N}\sum_{t=1}^{T}\sum_{m=1}^{M} I(z_t^{(n)} = m) \cdot logP(x_t^{(n)} \,|\, \alpha_m, \beta_m, z_t^{(n)} = m) \\ & + \sum_{m=1}^{M} I(z_1^{(n)} = m) \cdot logP(z_1^{(n)} \,|\, \theta) \\ & + \sum_{n=1}^{N}\sum_{t=2}^{T}\sum_{ci\in\Omega} [I(z_{t-1}^{(n)} = 1, z_t^{(n)} = 1, s_{s:s+2}^{(n)} = ci) \cdot log(1 - \rho_\mu^{ci} - \rho^{ci}) \\ & \quad + [I(z_{t-1}^{(n)} = 1, z_t^{(n)} = 2, s_{s:s+2}^{(n)} = ci) \cdot log\rho_\mu^{ci} \\ & \quad + [I(z_{t-1}^{(n)} = 1, z_t^{(n)} = 12, s_{s:s+2}^{(n)} = ci) \cdot log\rho^{ci} \\ & \quad + [I(z_{t-1}^{(n)} = 11, z_t^{(n)} = 11, s_{s:s+2}^{(n)} = ci) \cdot log(1 - \delta^{ci}) \\ & \quad + [I(z_{t-1}^{(n)} = 11, z_t^{(n)} = 12, s_{s:s+2}^{(n)} = ci) \cdot log\delta^{ci}]\end{aligned}$$

4 Expectation-Maximization Algorithm

Expectation–Maximization (EM) algorithm is an iterative approach to find model's local maximum likelihood and thus to estimate parameters. It contains two steps which are Expectation and Maximization. The first step aims to average hidden variables $\mathbf{Z}$ out of the log likelihood to achieve the Q function. Then Q function is optimized in the Maximization step to update parameters.

4.1 Expectation Step

Q function for multiple sequences is constructed based on the log likelihood (3.2), by averaging out hidden variables $\mathbf{Z}$. Note that θ and θ' are current and previous estimated model parameters.

$$\begin{aligned}Q(\theta \,|\, \theta') \propto & \sum_{n=1}^{N}\sum_{t=1}^{T}\sum_{m=1}^{M} P(z_t^{(n)} = m) \cdot logP(x_t^{(n)} \,|\, \alpha_m, \beta_m, z_t^{(n)} = m) \\ & + \sum_{m=1}^{M} P(z_1^{(n)} = m) \cdot logP(z_1^{(n)} \,|\, \theta) \\ & + \sum_{n=1}^{N}\sum_{t=2}^{T}\sum_{ci\in\Omega} [P(z_{t-1}^{(n)} = 1, z_t^{(n)} = 1, s_{s:s+2}^{(n)} = ci) \cdot log(1 - \rho_\mu^{ci} - \rho^{ci}) \\ & \quad + P(z_{t-1}^{(n)} = 1, z_t^{(n)} = 2, s_{s:s+2}^{(n)} = ci) \cdot log\rho_\mu^{ci} \\ & \quad + P(z_{t-1}^{(n)} = 1, z_t^{(n)} = 12, s_{s:s+2}^{(n)} = ci) \cdot log\rho^{ci} \\ & \quad + P(z_{t-1}^{(n)} = 11, z_t^{(n)} = 11, s_{s:s+2}^{(n)} = ci) \cdot log(1 - \delta^{ci}) \\ & \quad + P(z_{t-1}^{(n)} = 11, z_t^{(n)} = 12, s_{s:s+2}^{(n)} = ci) \cdot log\delta^{ci}]\end{aligned}$$

4.2 Hidden Markov Model Algorithm

In order to calculate value such as $P(z_{t-1}^{(n)} = 1, z_t^{(n)} = 2, s_{s:s+2}^{(n)} = ci)$, the following Hidden Markov Model related algorithms are used. m represents the current state and j represents the next state. Details is shown in Appendix1:

1. Forward Algorithm 6.3:

$$\begin{aligned}
F_m(t) &= P(\mathbf{X_{1:t}}, z_t = m \,|\, \theta) \\
F_j(t+1) &= (\sum_{m=1}^{M} F_m(t) \cdot P(z_{t+1} = j \,|\, z_t = m)) \cdot P(x_{t+1} \,|\, z_{t+1} = j) \\
F_m(1) &= F_1(1) = g(x_1 \,|\, \alpha_1, \beta_1) \quad \text{Assume to start from state 1}
\end{aligned}$$

2. Backward Algorithm 6.4:

$$\begin{aligned}
B_m(t) &= P(\mathbf{x_{t+1:T}} \,|\, z_t = m, \theta) \\
&= \sum_{j=1}^{M} P(x_{t+1} \,|\, z_{t+1} = j, \theta) \cdot B_j(t+1) \cdot P(z_{t+1} = j \,|\, z_t = m, \theta) \\
B_m(T) &= 1
\end{aligned}$$

3. One State Probability 6.5:

$$\begin{aligned}
L_m(t) &= P(z_t = m \,|\, \mathbf{x_{1:T}}, \theta) \\
&= \frac{F_m(t) \cdot B_m(t)}{\sum_{j=1}^{M} F_j(t) \cdot B_j(t)}
\end{aligned}$$

4. Two States Probability 6.6:

$$\begin{aligned}
H_{m,j}(t) &= P(z_t = m, z_{t+1} = j \,|\, \mathbf{x_{1:T}}, \theta) \\
&= \frac{F_m(t) \cdot P(z_{t+1} = j \,|\, z_t = m, \theta) \cdot P(x_{t+1} \,|\, z_{t+1} = j, \theta) \cdot B_j(t+1)}{\sum_{m=1}^{M} \sum_{j=1}^{M} F_m(t) \cdot P(z_{t+1} = j \,|\, z_t = m, \theta) \cdot P(x_{t+1} \,|\, z_{t+1} = j, \theta) \cdot B_j(t+1)}
\end{aligned}$$

   Because the model is codon specific, so it has to check if the next codon can be start codon. If not, the probabily would be assigned to 0.

$$H_{m,j}(t, ci) = \begin{cases} H_{m,j}(t) & \text{if } S_{t+1:t+3} = ci \\ 0 & \text{if } S_{t+1:t+3} \neq ci \end{cases}$$

4.3 Maximization step

In the Maximization step, Q function will be used to optimize transition and emission probability. Each sum part in Q function is independent, so emission probability only locates at the first sum part and transition probability only exists at the third part. Thus, it is possible to maximize them independently.

4.3.1 Transition Probability

To maximize transition probability, $Q(\theta \,|\, \theta')$ is derived with respect to transition probability first and then the derivatives are set to be 0. It is actually the derivation for Q function's third sum part.

There are close form solutions for ρ_{μ}^{ci}, ρ^{ci} and δ^{ci}. Full derivation is given in Appendix1 6.76.8:

$$\rho_{\mu}^{ci} = \frac{\sum_{n=1}^{N}\sum_{t=2}^{T} H_{1,2}(t-1, ci)}{\sum_{n=1}^{N}\sum_{t=2}^{T} P(s_{s:s+2}^{(n)} = ci \,|\, \mathbf{x_{1:T}})}$$

$$\rho^{ci} = \frac{\sum_{n=1}^{N}\sum_{t=2}^{T} H_{1,12}(t-1, ci)}{\sum_{n=1}^{N}\sum_{t=2}^{T} P(s_{s:s+2}^{(n)} = ci \,|\, \mathbf{x_{1:T}})}$$

$$\delta^{ci} = \frac{\sum_{n=1}^{N}\sum_{t=2}^{T} H_{11,12}(t-1, ci)}{\sum_{n=1}^{N}\sum_{t=2}^{T} P(s_{s:s+2}^{(n)} = ci \,|\, \mathbf{x_{1:T}})}$$

4.3.2 Emission Probability

To maximize emission probability, $Q(\theta \,|\, \theta')$ is derived with respect to emission probability. It is equivalent to compute the derivation for Q function's first sum part.

There is no close form solution for α_m and β_m when setting the derivatives to 0. So optimize function in Python would help to solve the issue by providing the formulas below. Detailed derivation information is given in Appendix1.

$$\frac{\partial Q(\theta \,|\, \theta')}{\partial \alpha_m} = \sum_{n=1}^{N}\sum_{t=1}^{T} L_m(t)[\sum_{s=1}^{x_t^{(n)}} \frac{1}{(\alpha_m + x_t^{(n)} - s)} + log\frac{\beta_m}{E^{(n)} + \beta_m}] \qquad 6.9$$

$$\frac{\partial Q(\theta \,|\, \theta')}{\partial \beta_m} = \sum_{n=1}^{N}\sum_{t=1}^{T} L_m(t)[\frac{\alpha_m \cdot E^{(n)}}{\beta_m \cdot (E^{(n)} + \beta_m)} - \frac{x_t^{(n)}}{E^{(n)} + \beta_m}] \qquad 6.10$$

4.4 Viterbi Algorithm

The aim of the project is to find translated uORFs and main coding regions. Therefore, after updating all parameters, it is possible to estimate the most likely state for each base by Viterbi Algorithm. Then translated region can be further identified by comparing states.

Viterbi Algorithm first calculates the log likelihood 3.2 for each possible sequence of states and then pick the one with the highest likelihood as the optimal path. By back tracking states from the end to the start, the best sequence of states can be found.

Here is how it works:

1. Initialisation: t = 1, state m = 1, ..., M

$$\phi_m(1) = \begin{cases} 1 & \text{if } m = 1 \\ 0 & \text{if } m \neq 1 \end{cases}, \qquad \psi_m(1) = 0$$

2. Induction: t = 1, ..., T-1, state m, j = 1, ..., M

$$\phi_j(t+1) = \max_{1 \leq m \leq M} [\phi_m(t) \cdot P(z_{t+1} = j \mid z_t = m, \theta) \cdot P(x_{t+1} \mid z_{t+1} = j, \theta)]$$
$$\psi_j(t+1) = \arg\max_{1 \leq m \leq M} \phi_m(t) \cdot P(z_{t+1} = j \mid z_t = m, \theta)$$

3. Termination

$$Path_{best} = \max_{1 \leq m \leq M} \phi_m(T)$$
$$q_T^* = \arg\max_{1 \leq m \leq M} \phi_m(T)$$

4. Backtrack states to establish best path: t = T-1, ..., 1

$$q_t^* = \psi_{q_{t+1}^*}(t+1)$$

5 Results

5.1 Simulation Data

Thirty datasets that each has one hundred mRNA sequences are simulated to test the correctness of the model. For simplicity, the only possible start codon is AUG and only three stop codons are covered. After generating 2700 single sequences with Algorithm 1 7.1, Algorithm 2 7.2 gathers them together. Besides, in Algorithm 2, some sequences contain both uORFs and the main coding region are changed to only uORFs cases. Furthermore, Algorithm 3 7.3 provides the no translation scenarios which are combined with previous sequences to construct the complete dataset later on.

5.2 Procedure

Inside each EM iteration, the forward algorithm (1) and backward algorithm (2) are computed and stored first. Then log likelihood 3.2, one state probability (3) and two state probability (4) can be calculated based on them. Afterwards, transition probability are updated according to the close form solution 4.3.1. Emission probability are estimated by scipy.optimize package in Python with the derivation functions 4.3.2. Instead of updating by fixed order, they get optimized randomly to avoid bias. After parameter updating, the log likelihood is recalcuated to check if there is an increase. Iterate these steps until the increase of log likelihood is so tiny or it has reached max iteration times. Moreover, with the optimized parameters, hidden states can be revealed by the Viterbi algorithm.

5.3 Analysis

5.3.1 Update All Transition Probability and Emission Probability

Following the steps described above 5.2, a set of refined parameters for both transition probability and emission probability can be obtained. The boxplots for transition probability 10, $alpha_m$ 8 and $beta_m$ 9 are shown below. The blue lines represent the true value when simulating data. However, $alpha_m$ and $beta_m$ both tend to have higher value than expected. As they are the parameters in Gamma distribution, the dependent relationship may cause this issue.

To check the correctness of the model, the number of unmatched states before and after the EM algorithm is compared 11. Twenty-seven out of thirty datasets performs quite well. But the remaining three datasets have poor results as their parameters are not updated 10 8 9 due to non-increase log likelihood. They may be stuck in the local maximum due to poor initial values. It is the limitation of the EM algorithm.

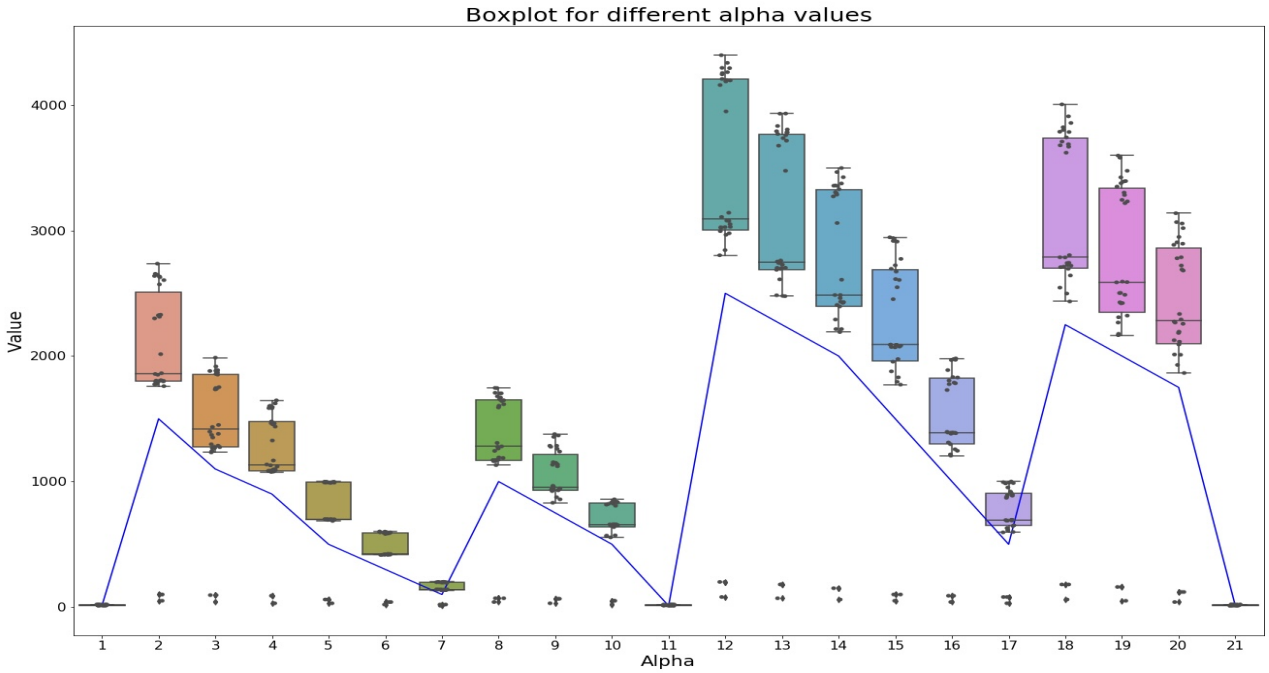

Figure 8:

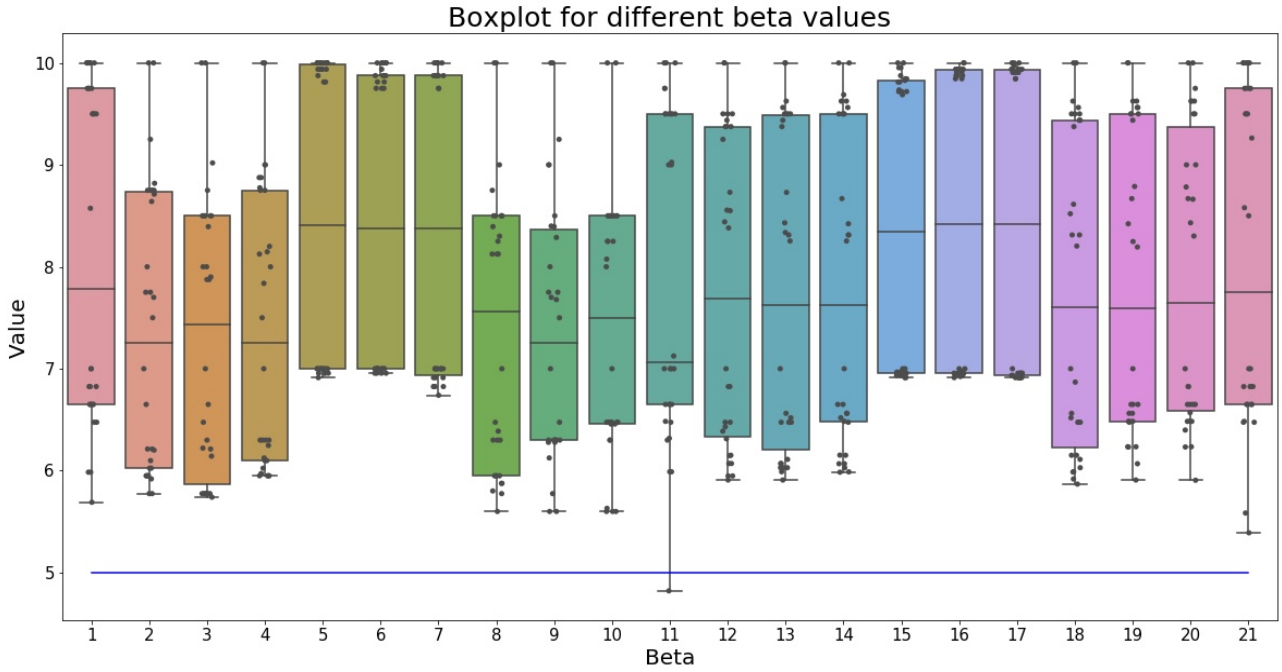

Figure 9:

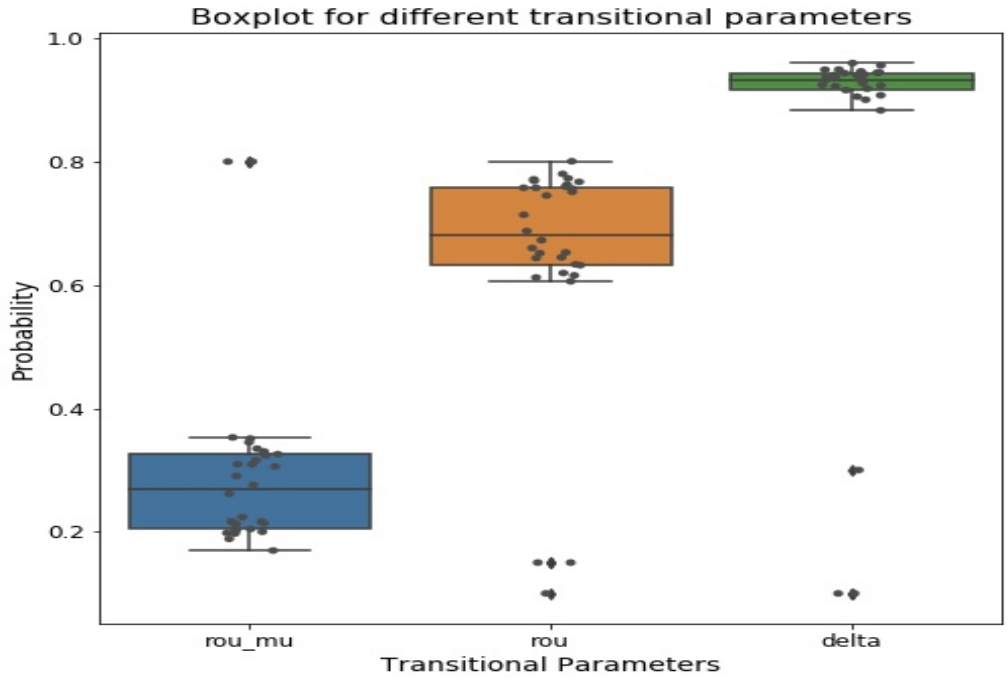

Figure 10:

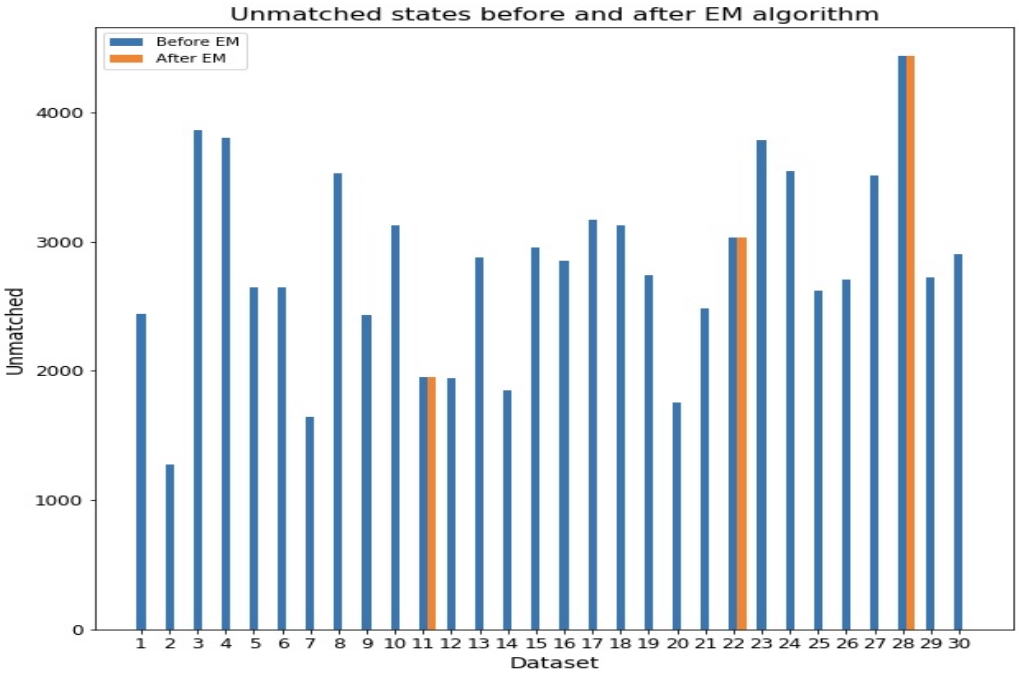

Figure 11:

5.3.2 Update All Transition Probability and $alpha_m$ With $beta_m$ Fixed

As proposed above, the relationship between $alpha_m$ and $beta_m$ may weaken the model performance. As a result, in order to verify this hypothesis, EM algorithm is applied to all the other parameters, except fixed true value $beta_m$. Similar graphs are plotted 12 13 that $alpha_m$ is more close to the correct value and there is no big difference between transition probability. This is because $alpha_m$ and $beta_m$ are dependent. To the contrast, transition probability and $beta_m$ are independent. Furthermore, twenty-nine out of thirty datasets 14 could back track the accurate hidden states.

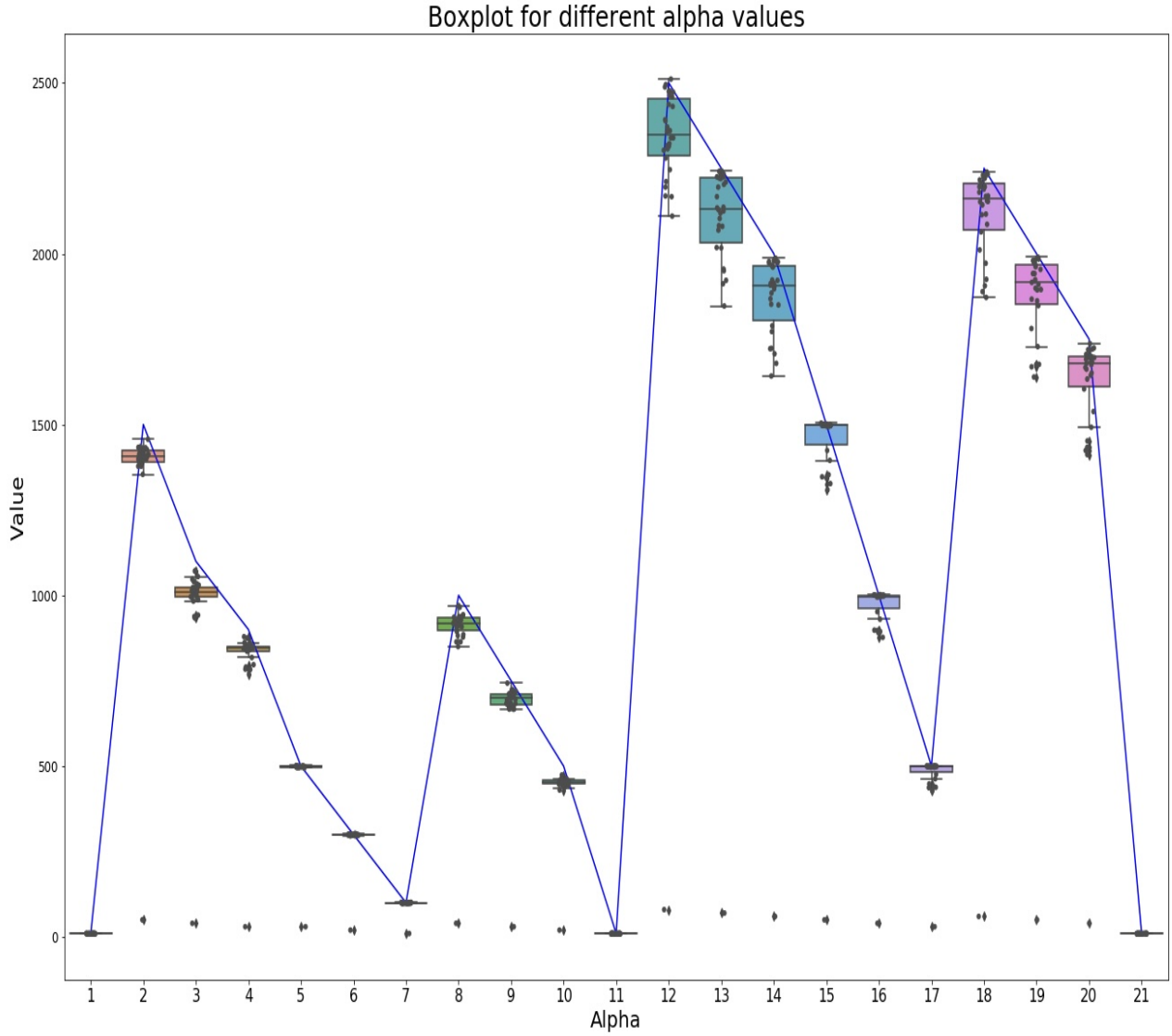

Figure 12:

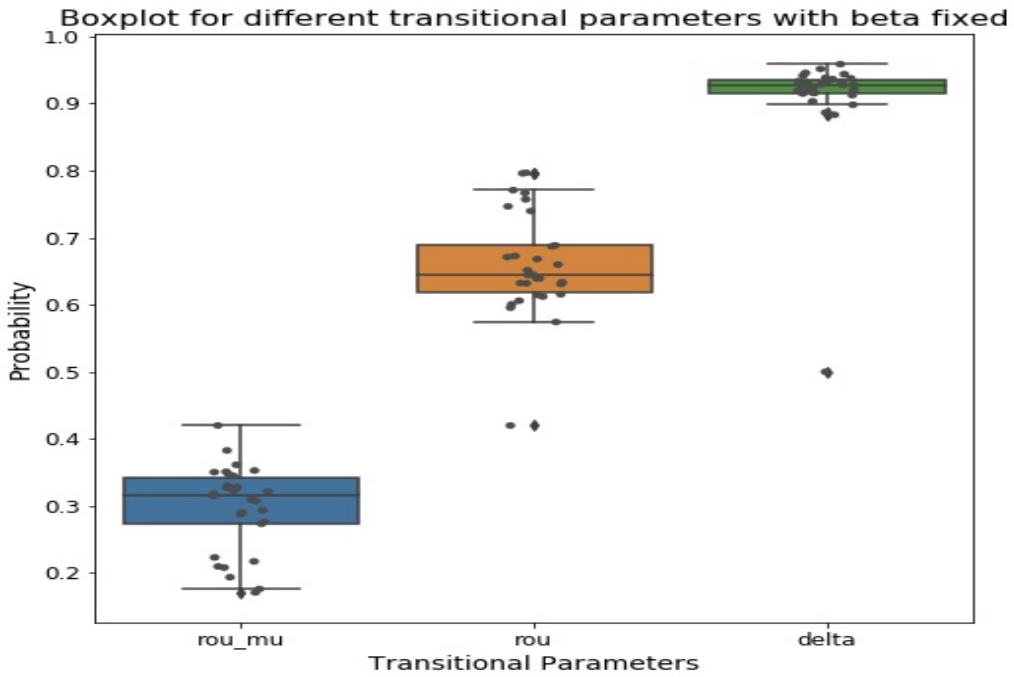

Figure 13:

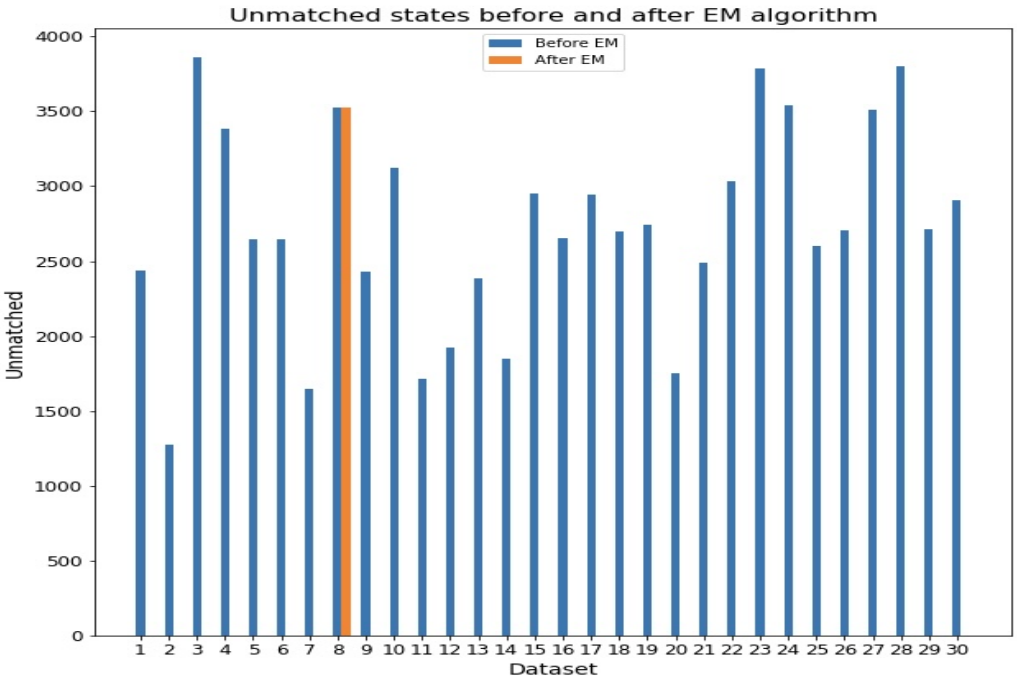

Figure 14:

References

[1] Millevoi S. Prats H. Touriol C. Arcondeguy T., Lacazette E. Vegf-a mrna processing, stability and translation: a paradigm for intricate regulation of gene expression at the post-transcriptional level. *Nucleic acids research*, 41(17):p7997–p8010, 2013.

[2] Mootha VK. Calvo SE., Pagliarini DJ. Upstream open reading frames cause widespread reduction of protein expression and are polymorphic among humans. *Proceedings of the National Academy of Sciences of the United States of America*, 106(18):p7507–p7512, 2009.

[3] Jonathan SW. Gloria AB. Ribosome profiling reveals the what, when, where and how of protein synthesis. *Nature Reviews Molecular Cell Biology*, 16:p651–p664, 2015.

[4] Clemente HS. Martinez Y. Dunand C. Bécard G. Combier JP Lauressergues D., Couzigou JM. Primary transcripts of microrna encode regulatory peptides. *Nature*, 520(7545):p90–p93, 2015.

[5] John RSN. Jonathan SW. Nicholas TI., Sina G. Genome-wide analysis in vivo of translation with nucleotide resolution using ribosome profiling. *Science*, 324(5924):p218–p223, 2009.

[6] Steinmetz LM Pelechano V., Wei W. Extensive transcriptional heterogeneity revealed by isoform profiling. *Nature*, 497(7447):p127–p131, 2013.

[7] Chen K. Shodiya M. Wang L. Yahiro K. Martins GM. Shastri N. Walter P. Starck SR., Tsai JC. Translation from the 5 untranslated region shapes the integrated stress response. *Nucleic acids research*, 351(6272):p7997–p8010, 2016.

[8] Héctor H. Valdivia. One gene, many proteins. *Proceedings of the National Academy of Sciences of the United States of America*, 100:p761–p763, 2007.

6 Appendix1

6.1 Negative Binomial Proof

$$
\begin{aligned}
P(x_{z,t}^{(n)} \mid \alpha_z, \beta_z) &= \int_0^\infty P(x_{z,t}^{(n)}, \mu_{z,t}^{(n)} E^{(n)} \mid \alpha_z, \beta_z) d\mu_{z,t}^{(n)} E^{(n)} \\
&= \int_0^\infty P(x_{z,t}^{(n)} \mid \mu_{z,t}^{(n)} E^{(n)}, \alpha_z, \beta_z) \cdot P(\mu_{z,t}^{(n)} E^{(n)} | \alpha_z, \beta_z) d\mu_{z,t}^{(n)} E^{(n)} \\
&= \int_0^\infty \frac{(\mu_{z,t}^{(n)} E^{(n)})^{x_{z,t}^{(n)}}}{x_{z,t}^{(n)}!} \cdot e^{-\mu_{z,t}^{(n)} E^{(n)}} \cdot \frac{(\frac{\beta_z}{E^{(n)}})^{\alpha_z}}{\Gamma(\alpha_z)} \cdot (\mu_{z,t}^{(n)} E^{(n)})^{(\alpha_z - 1)} \\
&\quad \cdot e^{-\frac{\beta_z}{E^{(n)}} \mu_{z,t}^{(n)} E^{(n)}} d\mu_{z,t}^{(n)} E^{(n)} \\
&= \frac{1}{x_{z,t}^{(n)}!} \cdot \frac{(\frac{\beta_z}{E^{(n)}})^{\alpha_z}}{\Gamma(\alpha_z)} \cdot \frac{\Gamma(x_{z,t}^{(n)} + \alpha_z)}{(1 + \frac{\beta_z}{E^{(n)}})^{x_{z,t}^{(n)} + \alpha_z}} \\
&\quad \cdot \int_0^\infty \frac{(1 + \frac{\beta_z}{E^{(n)}})^{(x_{z,t}^{(n)} + \alpha_z)}}{\Gamma(x_{z,t}^{(n)} + \alpha_z)} \cdot (\mu_{z,t}^{(n)} E^{(n)})^{x_{z,t}^{(n)} + \alpha_z - 1} \cdot e^{-(1 + \frac{\beta_z}{E^{(n)}}) \cdot \mu_{z,t}^{(n)} E^{(n)}} d\mu_{z,t}^{(n)} E^{(n)} \\
&= \frac{1}{x_{z,t}^{(n)}!} \cdot \frac{(\frac{\beta_z}{E^{(n)}})^{\alpha_z}}{\Gamma(\alpha_z)} \cdot \frac{\Gamma(x_{z,t}^{(n)} + \alpha_z)}{(1 + \frac{\beta_z}{E^{(n)}})^{x_{z,t}^{(n)} + \alpha_z}} \\
&= \frac{\Gamma(x_{z,t}^{(n)} + \alpha_z)}{x_{z,t}^{(n)}! \cdot \Gamma(\alpha_z)} \cdot \frac{(\frac{\beta_z}{E^{(n)}})^{\alpha_z}}{(1 + \frac{\beta_z}{E^{(n)}})^{x_{z,t}^{(n)}} \cdot (1 + \frac{\beta_z}{E^{(n)}})^{\alpha_z}} \\
&= \frac{\Gamma(x_{z,t}^{(n)} + \alpha_z)}{x_{z,t}^{(n)}! \cdot \Gamma(\alpha_z)} \cdot (\frac{\frac{\beta_z}{E^{(n)}}}{1 + \frac{\beta_z}{E^{(n)}}})^{\alpha_z} \cdot (\frac{1}{1 + \frac{\beta_z}{E^{(n)}}})^{x_{z,t}^{(n)}} \\
&\quad \Gamma(n) = (n-1)! \\
&\quad C_r^n = \binom{n}{r} = \frac{n!}{r! \cdot (n-r)!} \\
&= \frac{(x_{z,t}^{(n)} + \alpha_z - 1)!}{x_{z,t}^{(n)}! \cdot (\alpha_z - 1)!} \cdot (\frac{\beta_z}{E^{(n)} + \beta_z})^{\alpha_z} \cdot (\frac{E^{(n)}}{E^{(n)} + \beta_z})^{x_{z,t}^{(n)}} \\
&= \binom{x_{z,t}^{(n)} + \alpha_z - 1}{x_{z,t}^{(n)}} \cdot (\frac{\beta_z}{E^{(n)} + \beta_z})^{\alpha_z} \cdot (1 - \frac{\beta_z}{E^{(n)} + \beta_z})^{x_{z,t}^{(n)}}
\end{aligned}
$$

$$X_{z,t}^{(n)} \sim NB(\alpha_z, \frac{\beta_z}{E^{(n)} + \beta_z}),$$
$$g(x \mid r, p) = \binom{x + r - 1}{x} \cdot p^r \cdot (1 - p)^x$$

More distribution information here:

$$\mu_{z,t}^{(n)} E^{(n)} \sim Gamma(\alpha_z, \frac{\beta_z}{E^{(n)}})$$
$$g(\mu_{z,t}^{(n)} E^{(n)}) = \frac{(\frac{\beta_z}{E^{(n)}})^{\alpha_z}}{\Gamma(\alpha_z)} \cdot (\mu_{z,t}^{(n)} E^{(n)})^{(\alpha_z - 1)} \cdot e^{-\frac{\beta_z}{E^{(n)}} \mu_{z,t}^{(n)} E^{(n)}}$$

6.2 Log Likelihood for single mRNA sequence

$$
\begin{aligned}
log P(\mathbf{x_{1:T}}, \mathbf{z_{1:T}} \mid \theta, \mathbf{s_{1:T}}) &= log[P(\mathbf{x_{1:T}} \mid \mathbf{z_{1:T}}, \mathbf{s_{1:T}}, \theta) \cdot P(\mathbf{z_{1:T}} \mid \mathbf{s_{1:T}}, \theta)] \\
&= log[\prod_{t=1}^{T} P(x_t \mid z_t, \theta) \cdot P(z_1 \mid \theta) \cdot \prod_{t=2}^{T} P(z_t \mid z_{t-1}, s_{t:t+2}, \theta)] \\
&= log[\prod_{t=1}^{T} \prod_{m=1}^{M} P(x_t \mid \alpha_m, \beta_m, z_t = m)^{I(z_t=m)} \cdot \prod_{m=1}^{M} P(z_1 \mid \theta)^{I(z_1=m)} \\
&\quad \cdot \prod_{t=2}^{T} \prod_{m=1}^{M} \prod_{m'=1}^{M} P(z_t = m' \mid z_{t-1} = m, s_{t:t+2}, \theta)^{I(z_{t-1}=m, z_t=m')}]
\end{aligned}
$$

$$
\begin{aligned}
&\prod_{t=2}^{T} \prod_{m=1}^{M} \prod_{m'=1}^{M} P(z_t = m' \mid z_{t-1} = m, s_{t:t+2}, \theta)^{I(z_{t-1}=m, z_t=m')} \\
&\quad \propto \prod_{t=2}^{T} [P(z_t = 1 \mid z_{t-1} = 1, s_{t:t+2}, \theta)^{I(z_{t-1}=1, z_t=1)} \\
&\qquad \cdot P(z_t = 2 \mid z_{t-1} = 1, s_{t:t+2}, \theta)^{I(z_{t-1}=1, z_t=2)} \\
&\qquad \cdot P(z_t = 12 \mid z_{t-1} = 1, s_{t:t+2}, \theta)^{I(z_{t-1}=1, z_t=12)} \\
&\qquad \cdot P(z_t = 11 \mid z_{t-1} = 11, s_{t:t+2}, \theta)^{I(z_{t-1}=11, z_t=11)} \\
&\qquad \cdot P(z_t = 12 \mid z_{t-1} = 11, s_{t:t+2}, \theta)^{I(z_{t-1}=11, z_t=12)}] \\
&\quad = \prod_{t=2}^{T} \prod_{ci \in \Omega} [1 - \rho_\mu^{ci} - \rho^{ci}]^{I(z_{t-1}=1, z_t=1, s_{s:s+2}=ci)} \cdot [\rho_\mu^{ci}]^{I(z_{t-1}=1, z_t=2, s_{s:s+2}=ci)} \\
&\qquad \cdot [\rho^{ci}]^{I(z_{t-1}=1, z_t=12, s_{s:s+2}=ci)} \cdot [1 - \delta^{ci}]^{I(z_{t-1}=11, z_t=11, s_{s:s+2}=ci)} \\
&\qquad \cdot [\delta^{ci}]^{I(z_{t-1}=11, z_t=12, s_{s:s+2}=ci)} \qquad \Omega \text{ is the set of possible start codons}
\end{aligned}
$$

$$
\begin{aligned}
logP(\mathbf{x_{1:T}}, \mathbf{z_{1:T}} \,|\, \theta, \mathbf{s_{1:T}}) \propto & \sum_{t=1}^{T} \sum_{m=1}^{M} I(z_t = m) \cdot logP(x_t \,|\, \alpha_m, \beta_m, z_t = m) \\
& + \sum_{m=1}^{M} I(z_1 = m) \cdot logP(z_1 \,|\, \theta) \\
& + \sum_{t=2}^{T} \sum_{m=1}^{M} \sum_{m'=1}^{M} I(z_{t-1} = m, z_t = m') \cdot logP(z_t = m' \,|\, z_{t-1} = m, s_{t:t+2}, \theta) \\
= & \sum_{t=1}^{T} \sum_{m=1}^{M} I(z_t = m) \cdot logP(x_t \,|\, \alpha_m, \beta_m, z_t = m) \\
& + \sum_{m=1}^{M} I(z_1 = m) \cdot logP(z_1 \,|\, \theta) \\
& + \sum_{t=2}^{T} \sum_{ci \in \Omega} [I(z_{t-1} = 1, z_t = 1, s_{s:s+2} = ci) \cdot log(1 - \rho_{\mu}^{ci} - \rho^{ci}) \\
& \quad + I(z_{t-1} = 1, z_t = 2, s_{s:s+2} = ci) \cdot log\rho_{\mu}^{ci} \\
& \quad + I(z_{t-1} = 1, z_t = 12, s_{s:s+2} = ci) \cdot log\rho^{ci} \\
& \quad + I(z_{t-1} = 11, z_t = 11, s_{s:s+2} = ci) \cdot log(1 - \delta^{ci}) \\
& \quad + I(z_{t-1} = 11, z_t = 12, s_{s:s+2} = ci) \cdot log\delta^{ci}]
\end{aligned}
$$

6.3 Forward Algorithm

$$
\begin{aligned}
&F_m(t) = P(\mathbf{X_{1:t}}, z_t = m \,|\, \theta) \\
&F_m(1) = F_1(1) = g(x_1 \,|\, \alpha_1, \beta_1) \quad \text{Assume to start from state 1} \\
&P(\mathbf{X_{1:T}} \,|\, \theta) = \sum_{m=1}^{M} F_m(T)
\end{aligned}
$$

$$
\begin{aligned}
F_j(t+1) &= P(\mathbf{x_{1:t+1}}, z_{t+1} = j \,|\, \theta) \\
&= \sum_{m=1}^{M} P(\mathbf{x_{1:t+1}}, z_{t+1} = j, z_t = m \,|\, \theta) \\
&= \sum_{m=1}^{M} P(x_{t+1} \,|\, \mathbf{x_{1:t}}, z_{t+1} = j, z_t = m, \theta) \\
&\qquad \cdot P(\mathbf{x_{1:t}}, z_{t+1} = j, z_t = m \,|\, \theta) \\
&= \sum_{m=1}^{M} P(x_{t+1} \,|\, z_{t+1} = j) \cdot P(z_{t+1} = j \,|\, \mathbf{x_{1:t}}, z_t = m, \theta) \\
&\qquad \cdot P(\mathbf{x_{1:t}}, z_t = m \,|\, \theta) \\
&= \sum_{m=1}^{M} P(x_{t+1} \,|\, z_{t+1} = j) \cdot P(z_{t+1} = j \,|\, z_t = m) \cdot F_m(t) \\
&= (\sum_{m=1}^{M} F_m(t) \cdot P(z_{t+1} = j \,|\, z_t = m)) \cdot P(x_{t+1} \,|\, z_{t+1} = j)
\end{aligned}
$$

6.4 Backward Algorithm

$$
\begin{aligned}
&B_m(t) = P(\mathbf{x_{t+1:T}} \,|\, z_t = m, \theta) \\
&B_m(T) = 1 \\
&P(\mathbf{x_{1:T}} \,|\, \theta) = \sum_{m=1}^{M} P(\mathbf{x_{1:T}}, z_1 = m \,|\, \theta) \\
&\qquad = \sum_{m=1}^{M} P(\mathbf{x_{1:T}} \,|\, z_1 = m, \theta) \cdot P(z_1 = m \,|\, \theta) \\
&\qquad = \sum_{m=1}^{M} P(x_1 \,|\, \mathbf{x_{2:T}}, z_1 = m, \theta) \cdot P(\mathbf{x_{2:T}} \,|\, z_1 = m, \theta) \cdot P(z_1 = m \,|\, \theta) \\
&\qquad = \sum_{m=1}^{M} P(x_1 \,|\, z_1 = m, \theta) \cdot B_m(1) \cdot P(z_1 = m \,|\, \theta) \\
&\qquad = P(x_1 \,|\, z_1 = 1, \theta) \cdot B_1(1) \cdot P(z_1 = 1 \,|\, \theta) \quad \text{Assume to start from state 1}
\end{aligned}
$$

$$
\begin{aligned}
B_m(t) &= P(\mathbf{x_{t+1:T}} \mid z_t = m, \theta) \\
&= \sum_{j=1}^{M} P(\mathbf{x_{t+1:T}}, z_{t+1} = j \mid z_t = m, \theta) \\
&= \sum_{j=1}^{M} P(\mathbf{x_{t+1:T}} \mid z_t = m, z_{t+1} = j, \theta) \cdot P(z_{t+1} = j \mid z_t = m, \theta) \\
&= \sum_{j=1}^{M} P(\mathbf{x_{t+1:T}} \mid z_{t+1} = j, \theta) \cdot P(z_{t+1} = j \mid z_t = m, \theta) \\
&= \sum_{j=1}^{M} P(x_{t+1} \mid \mathbf{x_{t+2:T}}, z_{t+1} = j, \theta) \cdot P(\mathbf{x_{t+2:T}} \mid z_{t+1} = j, \theta) \cdot P(z_{t+1} = j \mid z_t = m, \theta) \\
&= \sum_{j=1}^{M} P(x_{t+1} \mid z_{t+1} = j, \theta) \cdot B_j(t+1) \cdot P(z_{t+1} = j \mid z_t = m, \theta)
\end{aligned}
$$

6.5 One State Probability

$$
\begin{aligned}
P(z_t = m, \mathbf{x_{1:T}}, \theta) &= P(\mathbf{x_{1:t}}, z_t = m, \theta) \cdot P(\mathbf{x_{1:T}} \mid \mathbf{x_{1:t}}, z_t = m, \theta) \\
&= F_m(t) \cdot P(\mathbf{x_{t+1:T}} \mid z_t = m, \theta) \\
&= F_m(t) \cdot B_m(t)
\end{aligned}
$$

$$
\begin{aligned}
L_m(t) &= P(z_t = m \mid \mathbf{x_{1:T}}, \theta) \\
&= \frac{P(z_t = m, \mathbf{x_{1:T}}, \theta)}{P(\mathbf{x_{1:T}}, \theta)} \\
&= \frac{P(z_t = m, \mathbf{x_{1:T}}, \theta)}{\sum_{j=1}^{M} P(z_t = j, \mathbf{x_{1:T}}, \theta)} \\
&= \frac{F_m(t) \cdot B_m(t)}{\sum_{j=1}^{M} F_j(t) \cdot B_j(t)}
\end{aligned}
$$

6.6 Two States Probability

$$
\begin{aligned}
P(\mathbf{x_{1:T}}, z_t = m, z_{t+1} = j, \theta) &= P(\mathbf{x_{1:t}}, z_t = m, \theta) \cdot P(\mathbf{x_{1:T}}, z_{t+1} = j \mid z_t = m, \mathbf{x_{1:t}}, \theta) \\
&= F_m(t) \cdot P(\mathbf{x_{1:T}}, z_{t+1} = j \mid z_t = m, \mathbf{x_{1:t}}, \theta) \\
&= F_m(t) \cdot \frac{P(\mathbf{x_{1:T}}, z_{t+1} = j, z_t = m, \mathbf{x_{1:t}}, \theta)}{P(z_t = m, \mathbf{x_{1:t}}, \theta)} \\
&= F_m(t) \cdot \frac{P(\mathbf{x_{1:T}} \mid z_{t+1} = j, z_t = m, \mathbf{x_{1:t}}, \theta)}{P(z_t = m, \mathbf{x_{1:t}}, \theta)} \\
&\qquad \cdot P(z_{t+1} = j, z_t = m, \mathbf{x_{1:t}}, \theta) \\
&= F_m(t) \cdot P(z_{t+1} = j \mid z_t = m, \mathbf{x_{1:t}}, \theta) \cdot P(z_t = m, \mathbf{x_{1:t}}, \theta) \\
&\qquad \cdot \frac{P(\mathbf{x_{1:T}} \mid z_{t+1} = j, z_t = m, \mathbf{x_{1:t}}, \theta)}{P(z_t = m, \mathbf{x_{1:t}}, \theta)} \\
&= F_m(t) \cdot P(z_{t+1} = j \mid z_t = m, \mathbf{x_{1:t}}, \theta) \\
&\qquad \cdot P(\mathbf{x_{1:T}} \mid z_t = m, z_{t+1} = j, \mathbf{x_{1:t}}, \theta) \\
&= F_m(t) \cdot P(z_{t+1} = j \mid z_t = m, \theta) \cdot P(\mathbf{x_{t+1:T}} \mid z_{t+1} =, \theta) \\
&= F_m(t) \cdot P(z_{t+1} = j \mid z_t = m, \theta) \cdot P(x_{t+1} \mid z_{t+1} = j, \theta) \\
&\qquad \cdot P(\mathbf{x_{t+2:T}} \mid z_{t+1} = j, \theta) \\
&= F_m(t) \cdot P(z_{t+1} = j \mid z_t = m, \theta) \cdot P(x_{t+1} \mid z_{t+1} = j, \theta) \cdot B_j(t+1)
\end{aligned}
$$

$$
\begin{aligned}
H_m j(t) &= P(z_t = m, z_{t+1} = j \mid \mathbf{x_{1:T}}, \theta) \\
&= \frac{P(\mathbf{x_{1:T}}, z_t = m, z_{t+1} = j, \theta)}{P(\mathbf{x_{1:T}}, \theta)} \\
&= \frac{P(\mathbf{x_{1:T}}, z_t = m, z_{t+1} = j, \theta)}{\sum_{m=1}^{M} \sum_{j=1}^{M} P(\mathbf{x_{1:T}}, z_t = m, z_{t+1} = j, \theta)} \\
&= \frac{F_m(t) \cdot P(z_{t+1} = j \mid z_t = m, \theta) \cdot P(x_{t+1} \mid z_{t+1} = j, \theta) \cdot B_j(t+1)}{\sum_{m=1}^{M} \sum_{j=1}^{M} F_m(t) \cdot P(z_{t+1} = j \mid z_t = m, \theta) \cdot P(x_{t+1} \mid z_{t+1} = j, \theta) \cdot B_j(t+1)}
\end{aligned}
$$

$$
H_m j(t, ci) = \begin{cases} H_m j(t) & \text{if } S_{t+1:t+3} = ci \\ 0 & \text{if } S_{t+1:t+3} \neq ci \end{cases}
$$

6.7 ρ_μ^{ci} and ρ^{ci} Formula

$$\frac{\partial Q(\theta \mid \theta')}{\partial \rho_\mu^{ci}} = \sum_{n=1}^{N}\sum_{t=2}^{T}[-\frac{P(z_{t-1}^{(n)}=1, z_t^{(n)}=1, s_{s:s+2}^{(n)}=ci)}{1-\rho_\mu^{ci}-\rho^{ci}} + \frac{P(z_{t-1}^{(n)}=1, z_t^{(n)}=2, s_{s:s+2}^{(n)}=ci)}{\rho_\mu^{ci}}] = 0$$

$$\Rightarrow \frac{(1-\rho_\mu^{ci}-\rho^{ci})\cdot\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1, z_t^{(n)}=2, s_{s:s+2}^{(n)}=ci)}{(\rho_\mu^{ci})\cdot(1-\rho_\mu^{ci}-\rho^{ci})}$$
$$-\frac{\rho_\mu^{ci}\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1, z_t^{(n)}=1, s_{s:s+2}^{(n)}=ci)}{(\rho_\mu^{ci})\cdot(1-\rho_\mu^{ci}-\rho^{ci})} = 0$$

$$\begin{aligned}\Rightarrow (1-\rho_\mu^{ci}-\rho^{ci})\cdot\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1, z_t^{(n)}=2, s_{s:s+2}^{(n)}=ci)\\ = \rho_\mu^{ci}\cdot\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1, z_t^{(n)}=1, s_{s:s+2}^{(n)}=ci)\end{aligned} \tag{1}$$

$$\frac{\partial Q(\theta \mid \theta')}{\partial \rho^{ci}} = \sum_{n=1}^{N}\sum_{t=2}^{T}[-\frac{P(z_{t-1}^{(n)}=1, z_t^{(n)}=1, s_{s:s+2}^{(n)}=ci)}{1-\rho_\mu^{ci}-\rho^{ci}} + \frac{P(z_{t-1}^{(n)}=1, z_t^{(n)}=12, s_{s:s+2}^{(n)}=ci)}{\rho^{ci}}] = 0$$

$$\Rightarrow \frac{(1-\rho_\mu^{ci}-\rho^{ci})\cdot\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1, z_t^{(n)}=12, s_{s:s+2}^{(n)}=ci)}{(\rho^{ci})\cdot(1-\rho_\mu^{ci}-\rho^{ci})}$$
$$-\frac{\rho^{ci}\cdot\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1, z_t^{(n)}=1, s_{s:s+2}^{(n)}=ci)}{(\rho^{ci})\cdot(1-\rho_\mu^{ci}-\rho^{ci})} = 0$$

$$\begin{aligned}\Rightarrow (1-\rho_\mu^{ci}-\rho^{ci})\cdot\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1, z_t^{(n)}=12, s_{s:s+2}^{(n)}=ci)\\ = \rho^{ci}\cdot\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1, z_t^{(n)}=1, s_{s:s+2}^{(n)}=ci)\end{aligned} \tag{2}$$

According to (1) + (2):

$$(1-\rho_\mu^{ci}-\rho^{ci})\{\sum_{n=1}^{N}\sum_{t=2}^{T}[P(z_{t-1}^{(n)}=1, z_t^{(n)}=2, s_{s:s+2}^{(n)}=ci) + P(z_{t-1}^{(n)}=1, z_t^{(n)}=12, s_{s:s+2}^{(n)}=ci)]\}$$
$$= (\rho_\mu^{ci}+\rho^{ci})\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1, z_t^{(n)}=1, s_{s:s+2}^{(n)}=ci)$$

$$\begin{aligned}\Rightarrow (1-\rho_\mu^{ci}-\rho^{ci})\{\sum_{n=1}^{N}\sum_{t=2}^{T}[P(z_{t-1}^{(n)}=1, z_t^{(n)}=2, s_{s:s+2}^{(n)}=ci) + P(z_{t-1}^{(n)}=1, z_t^{(n)}=12, s_{s:s+2}^{(n)}=ci)\\ + P(z_{t-1}^{(n)}=1, z_t^{(n)}=1, s_{s:s+2}^{(n)}=ci)]\} = \sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1, z_t^{(n)}=1, s_{s:s+2}^{(n)}=ci)\end{aligned}$$

$$\Rightarrow (1-\rho_\mu^{ci}-\rho^{ci})\sum_{n=1}^{N}\sum_{t=2}^{T}P(s_{s:s+2}^{(n)}=ci\,|\,\mathbf{x_{1:T}})=\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1,z_t^{(n)}=1,s_{s:s+2}^{(n)}=ci)$$

$$\begin{aligned}\Rightarrow 1-\rho_\mu^{ci}-\rho^{ci}&=\frac{\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1,z_t^{(n)}=1,s_{s:s+2}^{(n)}=ci)}{\sum_{n=1}^{N}\sum_{t=2}^{T}P(s_{s:s+2}^{(n)}=ci\,|\,\mathbf{x_{1:T}})}\\&=\frac{\sum_{n=1}^{N}\sum_{t=2}^{T}H_{1,1}(t-1,ci)}{\sum_{n=1}^{N}\sum_{t=2}^{T}P(s_{s:s+2}^{(n)}=ci\,|\,\mathbf{x_{1:T}})}\end{aligned}\tag{3}$$

Plug (3) to (1):

$$\begin{aligned}\rho_\mu^{ci}&=\frac{(1-\rho_\mu^{ci}-\rho^{ci})\cdot\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1,z_t^{(n)}=2,s_{s:s+2}^{(n)}=ci)}{\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1,z_t^{(n)}=1,s_{s:s+2}^{(n)}=ci)}\\&=\frac{\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1,z_t^{(n)}=1,s_{s:s+2}^{(n)}=ci)}{\sum_{n=1}^{N}\sum_{t=2}^{T}P(s_{s:s+2}^{(n)}=ci\,|\,\mathbf{x_{1:T}})}\cdot\frac{\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1,z_t^{(n)}=2,s_{s:s+2}^{(n)}=ci)}{\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1,z_t^{(n)}=1,s_{s:s+2}^{(n)}=ci)}\\&=\frac{\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1,z_t^{(n)}=2,s_{s:s+2}^{(n)}=ci)}{\sum_{n=1}^{N}\sum_{t=2}^{T}P(s_{s:s+2}^{(n)}=ci\,|\,\mathbf{x_{1:T}})}\\&=\frac{\sum_{n=1}^{N}\sum_{t=2}^{T}H_{1,2}(t-1,ci)}{\sum_{n=1}^{N}\sum_{t=2}^{T}P(s_{s:s+2}^{(n)}=ci\,|\,\mathbf{x_{1:T}})}\end{aligned}$$

Plug (3) to (2):

$$\begin{aligned}\rho^{ci}&=\frac{(1-\rho_\mu^{ci}-\rho^{ci})\cdot\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1,z_t^{(n)}=12,s_{s:s+2}^{(n)}=ci)}{\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1,z_t^{(n)}=1,s_{s:s+2}^{(n)}=ci)}\\&=\frac{\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1,z_t^{(n)}=1,s_{s:s+2}^{(n)}=ci}{\sum_{n=1}^{N}\sum_{t=2}^{T}P(s_{s:s+2}^{(n)}=ci)\,|\,\mathbf{x_{1:T}})}\cdot\frac{\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1,z_t^{(n)}=12,s_{s:s+2}^{(n)}=ci)}{\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1,z_t^{(n)}=1,s_{s:s+2}^{(n)}=ci)}\\&=\frac{\sum_{n=1}^{N}\sum_{t=2}^{T}P(z_{t-1}^{(n)}=1,z_t^{(n)}=12,s_{s:s+2}^{(n)}=ci)}{\sum_{n=1}^{N}\sum_{t=2}^{T}P(s_{s:s+2}^{(n)}=ci\,|\,\mathbf{x_{1:T}})}\\&=\frac{\sum_{n=1}^{N}\sum_{t=2}^{T}H_{1,12}(t-1,ci)}{\sum_{n=1}^{N}\sum_{t=2}^{T}P(s_{s:s+2}^{(n)}=ci\,|\,\mathbf{x_{1:T}})}\end{aligned}$$

6.8 δ^{ci} Formula

$$\frac{\partial Q(\theta \,|\, \theta')}{\partial \delta^{ci}} = \sum_{n=1}^{N} \sum_{t=2}^{T} [\frac{P(z_{t-1}^{(n)} = 11, z_t^{(n)} = 12, s_{s:s+2}^{(n)} = ci)}{\delta^{ci}} - \frac{P(z_{t-1}^{(n)} = 11, z_t^{(n)} = 11, s_{s:s+2}^{(n)} = ci)}{1 - \delta^{ci}}] = 0$$

$$\Rightarrow \frac{(1-\delta^{ci}) \cdot \sum_{n=1}^{N} \sum_{t=2}^{T} P(z_{t-1}^{(n)} = 11, z_t^{(n)} = 12, s_{s:s+2}^{(n)} = ci)}{(\delta^{ci}) \cdot (1 - \delta^{ci})} - \frac{\delta^{ci} \cdot \sum_{n=1}^{N} \sum_{t=2}^{T} P(z_{t-1}^{(n)} = 11, z_t^{(n)} = 11, s_{s:s+2}^{(n)} = ci)}{(\delta^{ci}) \cdot (1 - \delta^{ci})} = 0$$

$$\Rightarrow (1-\delta^{ci}) \cdot \sum_{n=1}^{N} \sum_{t=2}^{T} P(z_{t-1}^{(n)} = 11, z_t^{(n)} = 12, s_{s:s+2}^{(n)} = ci) = \delta^{ci} \cdot \sum_{n=1}^{N} \sum_{t=2}^{T} P(z_{t-1}^{(n)} = 11, z_t^{(n)} = 11, s_{s:s+2}^{(n)} = ci)$$

$$\Rightarrow \sum_{n=1}^{N} \sum_{t=2}^{T} P(z_{t-1}^{(n)} = 11, z_t^{(n)} = 12, s_{s:s+2}^{(n)} = ci) = \delta^{ci} \cdot [\sum_{n=1}^{N} \sum_{t=2}^{T} P(z_{t-1}^{(n)} = 11, z_t^{(n)} = 12, s_{s:s+2}^{(n)} = ci) + P(z_{t-1}^{(n)} = 11, z_t^{(n)} = 11, s_{s:s+2}^{(n)} = ci)] = \delta^{ci} \cdot \sum_{n=1}^{N} \sum_{t=2}^{T} P(s_{s:s+2}^{(n)} = ci \,|\, \mathbf{x_{1:T}})$$

$$\Rightarrow \delta^{ci} = \frac{\sum_{n=1}^{N} \sum_{t=2}^{T} P(z_{t-1}^{(n)} = 11, z_t^{(n)} = 12, s_{s:s+2}^{(n)} = ci}{\sum_{n=1}^{N} \sum_{t=2}^{T} P(s_{s:s+2}^{(n)} = ci) \,|\, \mathbf{x_{1:T}})} = \frac{\sum_{n=1}^{N} \sum_{t=2}^{T} H_{11,12}(t-1, ci)}{\sum_{n=1}^{N} \sum_{t=2}^{T} P(s_{s:s+2}^{(n)} = ci \,|\, \mathbf{x_{1:T}})}$$

6.9 Alpha

$$
\begin{aligned}
\frac{\partial Q(\theta \mid \theta')}{\partial \alpha_m} &= \frac{\partial}{\partial \alpha_m} \sum_{n=1}^{N} \sum_{t=1}^{T} P(z_t^{(n)} = m) log P(x_t^{(n)} \mid \alpha_m, \beta_m, z_t^{(n)} = m) \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} P(z_t^{(n)} = m) \frac{\partial}{\partial \alpha_m} log P(x_t^{(n)} \mid \alpha_m, \beta_m, z_t^{(n)} = m) \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} P(z_t^{(n)} = m) \frac{\partial}{\partial \alpha_m} [log[\binom{x_t^{(n)} + \alpha_m - 1}{x_t^{(n)}} \cdot (1 - \frac{E^{(n)}}{E^{(n)} + \beta_m})^{\alpha_m} \\
&\quad \cdot (\frac{E^{(n)}}{E^{(n)} + \beta_m})^{x_t^{(n)}}]] \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} P(z_t^{(n)} = m) \frac{\partial}{\partial \alpha_m} [log \binom{x_t^{(n)} + \alpha_m - 1}{x_t^{(n)}} + \alpha_m \cdot log(1 - \frac{E^{(n)}}{E^{(n)} + \beta_m}) \\
&\quad + x_t^{(n)} \cdot log(\frac{E^{(n)}}{E^{(n)} + \beta_m})] \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} P(z_t^{(n)} = m) \frac{\partial}{\partial \alpha_m} [log \prod_{s=1}^{x_t^{(n)}} (\alpha_m + x_t^{(n)} - s) + \alpha_m \cdot log(1 - \frac{E^{(n)}}{E^{(n)} + \beta_m}) \\
&\quad + x_t^{(n)} \cdot log(\frac{E^{(n)}}{E^{(n)} + \beta_m})] \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} P(z_t^{(n)} = m) [\sum_{s=1}^{x_t^{(n)}} \frac{\partial}{\partial \alpha_m} log(\alpha_m + x_t^{(n)} - s) + log(1 - \frac{E^{(n)}}{E^{(n)} + \beta_m})] \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} P(z_t^{(n)} = m) [\sum_{s=1}^{x_t^{(n)}} \frac{1}{(\alpha_m + x_t^{(n)} - s)} + log \frac{\beta_m}{E^{(n)} + \beta_m}] \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} L_m(t) [\sum_{s=1}^{x_t^{(n)}} \frac{1}{(\alpha_m + x_t^{(n)} - s)} + log \frac{\beta_m}{E^{(n)} + \beta_m}]
\end{aligned}
$$

6.10 Beta

$$
\begin{aligned}
\frac{\partial Q(\theta \mid \theta')}{\partial \beta_m} &= \frac{\partial}{\partial \beta_m} \sum_{n=1}^{N} \sum_{t=1}^{T} P(z_t^{(n)} = m) log P(x_t^{(n)} \mid \alpha_m, \beta_m, z_t^{(n)} = m) \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} P(z_t^{(n)} = m) \frac{\partial}{\partial \beta_m} log P(x_t^{(n)} \mid \alpha_m, \beta_m, z_t^{(n)} = m) \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} P(z_t^{(n)} = m) \frac{\partial}{\partial \beta_m} [log[\binom{x_t^{(n)} + \alpha_m - 1}{x_t^{(n)}} \cdot (1 - \frac{E^{(n)}}{E^{(n)} + \beta_m})^{\alpha_m} \\
&\quad \cdot (\frac{E^{(n)}}{E^{(n)} + \beta_m})^{x_t^{(n)}}]] \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} P(z_t^{(n)} = m) \frac{\partial}{\partial \beta_m} [log \binom{x_t^{(n)} + \alpha_m - 1}{x_t^{(n)}} + \alpha_m \cdot log(1 - \frac{E^{(n)}}{E^{(n)} + \beta_m}) \\
&\quad + x_t^{(n)} \cdot log(\frac{E^{(n)}}{E^{(n)} + \beta_m})] \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} P(z_t^{(n)} = m) \frac{\partial}{\partial \beta_m} [\alpha_m \cdot log \frac{\beta_m}{E^{(n)} + \beta_m} + x_t^{(n)} \cdot log(\frac{E^{(n)}}{E^{(n)} + \beta_m})] \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} P(z_t^{(n)} = m) [\alpha_m \cdot \frac{E^{(n)} + \beta_m}{\beta_m} \cdot \frac{E^{(n)} + \beta_m - \beta_m}{(E^{(n)} + \beta_m)^2} \\
&\quad + x_t^{(n)} \cdot \frac{E^{(n)} + \beta_m}{E^{(n)}} \cdot \frac{0 - E^{(n)}}{(E^{(n)} + \beta_m)^2}] \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} P(z_t^{(n)} = m) [\frac{\alpha_m \cdot E^{(n)}}{\beta_m \cdot (E^{(n)} + \beta_m)} - \frac{x_t^{(n)}}{E^{(n)} + \beta_m}] \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} L_m(t) [\frac{\alpha_m \cdot E^{(n)}}{\beta_m \cdot (E^{(n)} + \beta_m)} - \frac{x_t^{(n)}}{E^{(n)} + \beta_m}]
\end{aligned}
$$

7 Appendix2

7.1 Simulate one sequence

Algorithm 1: Simulate one sequence

Output: one RNA sequence with one observed data
Randomly generate five elements from ['A', 'C', 'G', 'U'] for curr_RNA;
if *'AUG' in curr_RNA* **then**
  Change one letter in 'AUG' to avoid start codon;
end
Randomly generate five elements from NB(state = 1) for curr_observed;
Randomly generate True or False for start;
while *start = False* **do**
  curr_RNA appends one from ['A', 'C', 'G', 'U'];
  curr_observed appends one from NB(state = 1);
  update start (boolean) randomly;
  if *last three letters in curr_RNA is 'AUG'* **then**
    Discard last three elements in curr_RNA;
    Discard last three elements in curr_observed;
    Set start = True;
  end
  if *length of curr_RNA exceeds 100* **then**
    return curr_RNA, curr_observed
  end
end
curr_RNA appends ['A', 'U', 'G'];
Generate one state from [1, 2, 12] with probability [0.2, 0.5, 0.3] ;
while *state = 1* **do**
  curr_observed appends three from NB(state = 1);
  update start (boolean) randomly;
  while *start = False* **do**
    curr_RNA appends one from ['A', 'C', 'G', 'U'];
    curr_observed appends one from NB(state = 1);
    update start (boolean) randomly;
    if *last three letters in curr_RNA is 'AUG'* **then**
      Discard last three elements in curr_RNA;
      Discard last three elements in curr_observed;
      Set start = True;
    end
    if *length of curr_RNA exceeds 100* **then**
      return curr_RNA, curr_observed
    end
  end
  curr_RNA appends ['A', 'U', 'G'];
  Generate one state from [1, 2, 12] with probability [0.2, 0.5, 0.3];
end

```
if state = 2 then
    curr_observed appends [NB(state = 2), NB(state = 3), NB(state = 4)];
    repeat = Randomly generate repeat times (between 20 to 30);
    curr_RNA appends codons repeat times (allow 'AUG');
    curr_observed appends [NB(state = 5), NB(state = 6), NB(state = 7)] repeat times;
    curr_RNA appends one stop codon;
    curr_observed appends [NB(state = 8), NB(state = 9), NB(state = 10)];
    curr_RNA appends five elements from ['A', 'C', 'G', 'U'];
    curr_observed appends five elements from NB(state = 11);
    state = 11;
    while state = 11 do
        update start (boolean) randomly;
        while start = False do
            curr_RNA appends one from ['A', 'C', 'G', 'U'];
            curr_observed appends one from NB(state = 11);
            update start (boolean) randomly;
            if last three letters in curr_RNA is 'AUG' then
                Discard last three elements in curr_RNA;
                Discard last three elements in curr_observed;
                Set start = True;
            end
            if length of curr_RNA exceeds 100 then
                return curr_RNA, curr_observed
            end
        end
        curr_RNA appends ['A', 'U', 'G'];
        Generate one state from [11, 12] with probability [0.2, 0.8];
        if state = 11 then
            curr_observed appends [NB(state = 11), NB(state = 11), NB(state = 11)];
        end
    end
end
if state = 12 then
    curr_observed appends [NB(state = 12), NB(state = 13), NB(state = 14)];
    repeat = Randomly generate repeat times (between 30 to 50);
    curr_RNA appends codons repeat times (allow 'AUG');
    curr_observed appends [NB(state = 15), NB(state = 16), NB(state = 17)] repeat
     times;
    curr_RNA appends one stop codon;
    curr_observed appends [NB(state = 18), NB(state = 19), NB(state = 20)];
    repeat = Randomly generate repeat times (between 20 to 30);
    curr_RNA appends element from ['A', 'C', 'G', 'U'] repeat times;
    curr_observed appends NB(state = 21) repeat times;
end
return curr_RNA, curr_observed
```

7.2 Simulate uORF + main ORF + both

Algorithm 2: Simulate uORF + main ORF + both

Input: length: number of RNA sequences to simulate
Output: RNA sequence, observed data
for *i in range(length)* **do**
    curr_RNA, curr_observed = Algorithm1();
    if *sequence contains uORF and main ORF* **then**
        for *every three of this sequence* **do**
            repeat = Randomly generate repeat times (between 40 to 80);
            Replace curr_RNA after state 10 by appending element from ['A', 'C', 'G', 'U'] repeat times;
            Replace curr_observed after state 10 by appending NB(state = 11) repeat times;
            if *'AUG' in replaced part* **then**
                Change one letter in 'AUG' to avoid start codon;
            end
        end
    end
    Add curr_RNA to RNA_sequence;
    Add curr_observed to observed_data;
end
return RNA_sequence, observed_data;

7.3 Simulate neither

Algorithm 3: Simulate neither

Input: length: number of RNA sequences to simulate
Output: RNA sequence, observed data
for *i in range(length)* **do**
    repeat = Randomly generate repeat times (between 40 to 80);
    Generate from ['A', 'C', 'G', 'U'] repeat times for curr_RNA;
    Generate NB(state = 1) repeat times for curr_observed;
    if *'AUG' in curr_RNA* **then**
        Change one letter in 'AUG' to avoid start codon;
    end
    Add curr_RNA to RNA_sequence;
    Add curr_observed to observed_data;
end
return RNA_sequence, observed_data;